UCUENCA

Universidad de Cuenca

Facultad de Ingeniería

Carrera de Ingeniería de Sistemas

Diseño e Implementación de un Asistente Virtual para facilitar el acceso a geoinformación a través de una Infraestructura de Datos Espaciales (IDE)

Trabajo de titulación previo a la obtención del título de Ingeniero de Sistemas

Autor:

Juan Francisco Bustamante Avila

Director:

Carlos Villie Morocho Zurita

ORCID: 0000-0002-8196-2644

Cuenca, Ecuador

2023-09-22

Resumen

El presente trabajo de titulación “Diseño e Implementación de un Asistente Virtual para facilitar el acceso a geoinformación a través de una Infraestructura de Datos Espaciales (IDE)” se enfoca en desarrollar un Asistente Virtual que mejore el acceso y uso de geoinformación relacionada con el ordenamiento territorial en el cantón Cuenca, Azuay, Ecuador, a través de la IDE UCuenca. La investigación se basa en el contexto de la IDE UCuenca, que reúne datos geoespaciales desde diversas fuentes de geoinformación para usuarios con diferentes niveles de conocimiento. Se consideraron usuarios técnicos en temas de ordenamiento territorial, pero también ciudadanos comunes que carecen de dicho entendimiento. Se empleó el framework Rasa Open Source, utilizando técnicas de procesamiento de lenguaje natural y aprendizaje automático, se construyó un nuevo esquema de conocimiento de geoinformación para entrenar el Asistente Virtual de dominio específico del cantón Cuenca, incluyendo términos técnicos y jerga local. El diseño de la interfaz se centró en la facilidad de uso y se consideraron diferentes métodos de comunicación, como texto y la voz. El objetivo es brindar una experiencia interactiva y eficiente, facilitando el acceso a la información geoespacial y contribuyendo a los objetivos del proyecto principal “Los Asistentes Virtuales como aporte en los procesos de participación ciudadana de los planes de desarrollo y ordenamiento territorial” (IDE AV-PPGIS), con el propósito de proporcionar información organizada y personalizada sobre los planes de ordenamiento territorial a los ciudadanos, mejorando la gobernanza y democratizando la información del territorio.

Palabras clave: Asistente Virtual(AV), Infraestructura de Datos Espaciales(IDE), geoservicios, Rasa Open Source

Abstract

The present degree project ”Design and Implementation of a Virtual Assistant to facilitate access to geoinformation through a Spatial Data Infrastructure (SDI)” focuses on developing a Virtual Assistant that enhances access and usage of geoinformation related to territorial planning in Cuenca, Azuay, Ecuador, through the UCuenca SDI. The research is based on the context of the UCuenca SDI, which gathers geospatial data from various sources for users with different levels of expertise, ranging from technical knowledge in territorial planning to ordinary citizens lacking such understanding.

The Rasa Open Source framework was employed, using natural language processing and machine learning techniques to create a new geoinformation knowledge schema for training the Virtual Assistant with specific data from Cuenca, including technical terms and local jargon. The interface design focused on user-friendliness and considered various communication methods, such as text and voice. The objective is to provide an interactive and efficient experience, facilitating access to geospatial information and contributing to the goals of the main project ”Virtual Assistants as a contribution to citizen participation processes in development and territorial planning plans” (IDE AV-PPGIS). This aims to provide organized and personalized information about territorial planning to citizens, improving governance and democratizing territorial information.

Keywords: Virtual Asistant(AV), Spatial Date Infrastructure(IDE) geo service, Rasa Open Source

Índice de contenido

Índice de figuras

Índice de tablas

Agradecimientos

Quiero expresar mi más sincero agradecimiento a todas las personas que me han apoyado en mi camino para convertirme en Ingeniero en Sistemas. Su amor, apoyo y aliento han sido fundamentales en este logro y me siento profundamente agradecido por su presencia en mi vida.

En primer lugar, quiero agradecer a mi querida mamá, Eulalia Avila, por su amor incondicional y por ser mi fuente constante de inspiración. Gracias por creer en mí y por brindarme todo el apoyo emocional y económico necesario para alcanzar mis metas. Tu amor y sacrificio son invaluables y estoy eternamente agradecido.

A mi padre, Jhonny Bustamante, quiero expresar mi gratitud por estar siempre a mi lado, brindándome tu apoyo incondicional. Tus palabras de aliento y tu presencia han sido un verdadero regalo en mi vida. Gracias por ser mi compañero en este viaje y por creer en mí.

A mi amada hermana, Emilia Bustamante, gracias por su constante apoyo y aliento en cada paso de mi carrera. Tu sabiduría, experiencia y consejos han sido invaluables en mi formación como profesional. Gracias por ser mi ejemplo de dedicación y perseverancia.

Quiero agradecer también a mi familia y amigos cercanos que han estado a mi lado a lo largo de mis estudios. Su constante apoyo, ánimo y comprensión han sido fundamentales en los momentos de dificultad. Gracias por creer en mí y por ser mi red de apoyo inquebrantable.

A mi director, Ing. Vilie Morocho, y asesora Ing. Rosario Achig, les estoy profundamente agradecido por la oportunidad de realizar esta tesis y por su invaluable guía y conocimiento. También quiero agradecer al proyecto de investigación/vinculación “Los Asistentes Virtuales como aporte en los procesos de participación ciudadana de los planes de desarrollo y ordenamiento territorial” (AVPPGIS), al Vicerrectorado de Investigación de la Universidad de Cuenca, que brindaron el financiamiento para la publicación de los artículos que forman parte de esta tesis. Al Programa Iberoamericano de Ciencia y Tecnología para el Desarrollo (CYTED) - Red IDEAIS por concederme pasantías internacionales en México. A todos gracias por abrirme las puertas a nuevas oportunidades académicas y por su compromiso en mi desarrollo como profesional.

A cada una de las personas mencionadas y a todas aquellas que de alguna manera han formado parte de mi camino académico, quiero expresar mi más sincero agradecimiento. Sin su apoyo, aliento y confianza, no habría logrado llegar hasta aquí. Estoy profundamente agradecido y espero poder corresponderles con éxito y felicidad en la siguiente etapa de la vida.

¡Gracias de todo corazón!

Juan Francisco Bustamante Avila

Abreviaciones y Acrónimos

AV - Asistente Virtual

IDE - Infraestructura de Datos Espaciales

GIS - Sistemas de Información Geográfica

AV-PPGIS - Los Asistentes Virtuales como aporte en los procesos de participación ciudadana de los planes de desarrollo y ordenamiento territorial

DIET - Dual Intent and Entity Transformer

NLP - Procesamiento de Lenguaje Natural

NLG - Generación de Lenguaje Natural

NLU - Compresión de Lenguaje Natutal

IA - Inteligencia Artificial

API - Interfaz de programación de Aplicaciones

GPT - Generative Pre-trained Transformer

IGM - Instituto Geográfico Militar

INEC - Instituto Nacional de Estadística y Censos

SIGTIERRAS - Sistema Nacional de Información de Tierras

MAG - Ministerio de Agricultura y Ganadería

MIDUVI - Ministerio de Desarrollo Urbano y Vivienda

IIGE - Instituto de Investigación Geológico y Energético

INOCAR - Instituto Oceanográfico de la Armada

IEE - Instituto Espacial Ecuatoriano

Centro Sur - Empresa Eléctrica Regional Centro Sur

MPS - Ministerio de Salud Pública

CNEL - Corporación Nacional de Electricidad

Glosario

Bot: Programa de software que puede ejecutar comandos, responder a mensajes o realizar tareas de rutina, como búsquedas en línea, ya sea automáticamente o con una mínima intervención humana.

NLP: Esta sigla significa Natural Language Processing, en español procesamiento de lenguaje natural. Las computadoras usan algoritmos de inteligencia artificial para entender, interpretar y manipular el lenguaje humano.

NLU: Esta sigla significan Natural Language Understanding, en español Comprensión del Lenguaje Natural, es una rama de la inteligencia artificial que utiliza software de computadora para comprender la entrada en forma de oraciones usando texto o habla.

NGL: Esta sigla significa Natural Language Generation, en español: Generación de Lenguaje Natural, y se trata del proceso contrario al NLU, es decir, cuando una máquina escribe o habla en lenguaje humano.

Intent: En español, intención, dentro del contexto, con un conjunto de expresiones de lenguaje dichas por el usuario que expresan una acción concreta.

Entity: En español entidad, dentro del contexto, hace referencia a información que es extraída de lo que nos dice un usuario, por ejemplo, “Me llamo nombre”, donde nombre es la entidad.

Action: Una action o acción es la tarea que se espera que el bot realice. En la gran mayoría de los casos se tendrá que consultar una API externa, con parámetros introducidos por el usuario.

Pipelines: En la lista de componentes NLU que define librerías de análisis para procesar uno por uno los mensajes de entrada de usuario, antes de devolver la salida estructurada final.

Rasa CORE: El motor de diálogo que decide qué hacer a continuación en una conversación en función del contexto.

Rasa NLU: Un elemento Rasa NLU que procesa los mensajes entrantes. Los componentes realizan tareas que van desde la extracción de entidades hasta lograr la clasificación de intenciones y el preprocesamiento.

Action Server: es un componente esencial de la plataforma Rasa que permite la ejecución de acciones personalizadas durante las interacciones con el asistente virtual.

Dominio Abierto: Los asistentes de dominio abierto pueden hablar sobre temas generales y responder en consecuencia.

Dominio Específico: Los asistentes virtuales de dominio específico se han definido como bots que se especializan en un área de conocimiento.

1. Introducción

1.1. Motivación y Contexto

La gestión del territorio es un aspecto fundamental para el desarrollo sostenible de las ciudades y regiones. La información geoespacial se ha convertido en una herramienta clave para la toma de decisiones. Dentro de este ámbito, la participación ciudadana en la elaboración de los planes de ordenamiento territorial es una pieza fundamental para enriquecer la calidad del mismo. Al involucrar a los ciudadanos, se aprovecha su conocimiento y experiencia local, lo que puede llevar a soluciones más informadas y contextualmente relevantes. Además, la participación ciudadana ayuda a identificar problemas y desafíos que pueden no haber sido considerados inicialmente, lo que contribuye a la formulación de políticas más efectivas y a la captación de geoinformación de los territorios.

Las Tecnologías de la Información y Comunicación (TIC) y la geoinformación desempeñan un papel importante en los procesos de planificación de ordenamiento territorial al proporcionar herramientas y recursos que facilitan la recopilación, análisis, gestión, y visualización de datos espaciales. Este último, pueden ser los visores de mapas junto a las Infraestructuras de Datos Espaciales (IDE) que permiten almacenar y gestionar grandes volúmenes de datos. Los sistemas de información geográfica y las bases de datos espaciales permiten organizar y administrar la geoinformación de manera eficiente, lo que facilita el acceso y la recuperación de datos para su uso en el proceso de planificación.

El caso de estudio de este proyecto tiene como finalidad la implementación de un Asistente Virtual por medio de Procesamiento de Lenguaje Natural (NLP) en el campo de la Inteligencia Artificial (IA). La geoinformación de los GADs municipales y diferentes instituciones públicas y privadas como IGM (Instituto Geográfico Militar), MAG (Ministerio de Agricultura y Ganadería), MIDUVI (Ministerio de Desarrollo Urbano y vivienda), Universidad del Azuay, entre otros relevantes han servidoro de base para este proyecto. Con este Asistente Virtual la ciudadanía del cantón Cuenca contará acceso y precisión a esta geoinformación dentro de los planes de ordenamiento territorial.

En la actualidad, el campo de la geoinformación es uno menos explorados dentro de las tec-

nologías de Procesamiento de Lenguaje Natural. Lo que motiva a desarrollar e implementar un Asistente Virtual para el uso de geoinformación dentro del cantón Cuenca. Al diseñar este AV se proveerá de una herramienta eficiente que permita el acceso ágil, eficiente y una búsqueda en servidores de mapas a través de una IDE. El aporte del Asistente Virtual será ayudar a los ciudadanos que no cuenten con conocimientos técnicos sobre el manejo de datos espaciales a obtener y visualizar geoinformación de manera rápida, relevante y actualizada de los territorios de interés.

1.2. Planteamiento del Problema

La información y la planificación territorial con la que cuenta actualmente el Gobierno Autónomo Descentralizado de Cuenca muestra deficiencia en cuanto al acceso público a la distribución de suelos, vías, planificación urbana, entre otras (Vivanco Cruz et al., 2020). La falta de participación ciudadana en la elaboración de los planes de ordenamiento territorial genera dificultad para manejar y entender la información geoespacial que provee el GAD Municipal.

De acuerdo a la investigación de Cruz (Cruz et al., 2018), indica que las herramientas relacionadas con la información geoespacial que maneja el GAD municipal son difíciles de manejar y entender para algunos ciudadanos, especialmente a los que habitan en el sector rural, quienes aparentemente cuentan con un bajo nivel de conocimientos técnicos e informáticos. Por lo tanto, los ciudadanos se ven obligados a realizar trámites de manera presencial, aplicando solicitudes que pueden resultar complicadas y tomar mucho tiempo, cuando actualmente la vía más eficaz sería optar por el uso de la tecnología(Vivanco Cruz et al., 2019).

Otro desafío radica en la dispersión de la información geoespacial en diversas instituciones públicas y privadas, sin una adecuada clasificación o regulación. Esta falta de estandarización dificulta la búsqueda y el acceso a los datos necesarios para llevar a cabo actividades relacionadas con la participación ciudadana y el ordenamiento territorial (Granell et al., 2021).

Además, la información geoespacial se encuentra almacenada en diferentes tecnologías, como los servicios web geoespaciales (WFS, WMS) y herramientas como ArcGIS. Esta diversidad tecnológica dificulta la integración y la interoperabilidad de los datos. Lo cual provoca que la información deba ser obtenida en diferentes entidades públicas y privadas, desagilizando el proceso(Granell et al., 2021).

Estos desafíos presentan obstáculos significativos para la gestión eficiente de la geoinformación. Es fundamental abordar estas problemáticas para facilitar el acceso a la información geoespacial, promover su clasificación y regulación, así como fomentar la adopción de estándares tecnológicos que permitan una integración fluida de los datos. De esta manera, se

podrán impulsar procesos de ordenamiento territorial más eficaces y participativos.

1.3. Solución Planteada

Para facilitar el acceso a la información, existen propuestas de mejoramiento de la usabilidad de sistemas o plataformas de geoinformación (Cruz et al., 2018). Una solución integral a los desafíos planteados puede abordarse mediante la implementación de un Asistente Virtual que facilite el acceso a la geoinformación y su visualización en un visor de mapas. Esta solución brindará una interfaz intuitiva y amigable para los usuarios, incluyendo aquellos con bajos conocimientos técnicos e informáticos.

Para organizar y clasificar la geoinformación dispersa en las diferentes entidades generadoras, se propone utilizar la guía del SENPLADES, que para el cantón Cuenca consta de los 6 ejes de planificación para el ordenamiento territorial: biofísico, sociocultural, cultural y patrimonial, económico, asentamientos humanos y canales de relación, uso y ocupación del suelo (SENPLADES, 2021). Este marco proporciona una estructura unificada para categorizar y etiquetar los datos geoespaciales. Además, estos datos clasificados se utilizan para el entrenamiento del Asistente Virtual, lo que mejora su capacidad de comprender las consultas y brindar respuestas relevantes.

En cuanto a la diversidad tecnológica de los geoservicios, se buscará solventar este problema con la interacción del Asistente Virtual. El AV está diseñado para interactuar con diferentes tecnologías, como WFS, WMS y herramientas ArcGIS, ocultando estas complejidades técnicas al usuario. Esto permitirá una experiencia fluida y unificada, donde el usuario podrá realizar consultas y obtener resultados independientemente de la tecnología subyacente.

Con esta solución, se logrará una mayor accesibilidad a la geoinformación, una clasificación efectiva de los datos y una experiencia de usuario simplificada. Esto impulsará la participación ciudadana en el proceso de ordenamiento territorial, facilitando el acceso a la geoinformación y contribuyendo a un desarrollo sostenible y planificado de los territorios. Esta tesis forma parte del proyecto de investigación/vinculación, titulado “Los Asistentes Virtuales como aporte en los procesos de participación ciudadana de los planes de desarrollo y ordenamiento territorial” (IDE AV-PPGIS) que se desarrolla en el grupo de investigación en IDEs, por lo que el AV sé desplegó en la plataforma IDE UCuenca.

1.4. Objetivos

1.4.1. Objetivo General

Implementar un Asistente Virtual por medio de texto y voz para mejorar la usabilidad de las plataformas de infraestructura de datos espaciales, tomando como caso de estudio la mejora de la participación pública en proceso de planificación territorial.

1.4.2. Objetivos Específicos

1. Organizar, filtrar, clasificar y depurar la geoinformación para el entrenamiento del AV.
2. Diseñar e implementar un AV de texto y voz preferentemente en código abierto, capaz de integrarse a la IDE UCuenca.
3. Integrar en la IDE UCuenca con el AV con la capacidad diseñada para experimentación en el caso de estudio.
4. Diseño de escenarios de prueba para el uso del AV con la IDE UCuenca.

2. Marco Teórico

2.1. Estado del Arte

Un Asistente Virtual es un software de procesamiento de lenguaje natural que incluye reconocimiento de texto, comprensión, generación de lenguaje natural y transformación de texto a voz para mejorar la experiencia de interacción con los usuarios a través de preguntas y respuestas(Adamopoulou and Moussiades, 2020). En la revisión de literatura, la relación directa entre la geoinformación y los AV es un campo relativamente nuevo (Granell et al., 2021).

Según la accesibilidad de los asistentes de conocimiento, se pueden dividir en 2 categorías principales: *Dominio abierto* y *Dominio específico*. Los AV de dominio abierto pueden hablar sobre temas generales y responder en consecuencia. Los AV de dominio específico se han definido como bots que se especializan en un área de conocimiento (Pavel, 2021). En caso de esta tesis, es de *dominio específico*, ya que el conocimiento que maneja es exclusivamente de geoinformación relacionada con el ordenamiento territorial del cantón Cuenca en Azuay, Ecuador.

En la publicación ”Integrating a chatbot with a GIS-MCDM system” de Frei (Frei, 2018) se comparan tipos de AV, donde se proporciona una clasificación primaria de estos como son Atención al Cliente, Social Networks/Marketing, E-commerce, entre otros, y se expone este nuevo paradigma de AV con una integración e interacción a sistemas de geoinformación (GIS). A estos AV el autor Frei los categoriza como “Geobots”, aunque en dicho artículo no describe implementación ni desarrollo.

En el caso de los Asistentes Virtuales más utilizados y conocidos en el medio, tales como Google Assistant, Siri, Alexa y PAVAL (Massai et al., 2019), estos utilizan *skills* de Google, Yandex y Apple, entre otros. Para calcular distancias, encontrar caminos, y reseñas de lugares, estos utilizan servicios geográficos como GPS (posición) y *Geographic Information Retrieval* (GIR).

El caso más común de uso de estos asistentes es para tareas genéricas como las búsquedas de sitios de interés o direcciones específicas. A diferencia de lo propuesto en esta tesis, donde principalmente se accede a información no común, y que además se encuentra en servidores

de instituciones oficiales que las herramientas de Google no acceden.

Es importante mencionar las tecnologías de Inteligencia Artificial con NLP emergentes como ChatGTP, Jasper Chat, Bing Chat, YouChat, Elicit, Socratic, entre otras. Estos implementan un tipo de red neuronal conocida como transformador (*transformer networks*). Estos transformadores son un tipo de algoritmo de aprendizaje profundo para procesar y generar texto como respuesta a las interacciones del usuario(Mathew, 2023). Específicamente se refiere a la implementación de la arquitectura de modelo de lenguaje *Generative Pre-trained Transformer (GPT)*. Los cuales tienen distintos tipos de entrenamiento y aplicaciones. Se hace una comparación entre los resultados obtenidos de este trabajo y estas nuevas tecnologías en la sección de discusión y conclusiones.

2.2. Conceptos de Asistente Virtual

2.2.1. Redes Neuronales

Una red neuronal es un modelo computacional inspirado en el funcionamiento del cerebro humano. Está compuesta por un conjunto interconectado de unidades llamadas neuronas artificiales o nodos, que trabajan en conjunto para procesar información y realizar tareas específicas. Estas neuronas están organizadas en capas, con una capa de entrada que recibe los datos de entrada, una o más capas ocultas que procesan la información y una capa de salida que produce los resultados finales (Williams and Zweig, 2016).

El proceso de programación de una red neuronal se basa en dos etapas principales: la etapa de entrenamiento y la etapa de inferencia. En la etapa de entrenamiento, se alimenta a la red neuronal con un conjunto de datos de entrenamiento, que consta de entradas y salidas esperadas. Durante el entrenamiento, la red ajusta iterativamente los pesos sinápticos de sus conexiones para minimizar la diferencia entre las salidas producidas por la red y las salidas esperadas (Merrill and Sabharwal, 2022). Esto se logra mediante algoritmos de aprendizaje que utilizan técnicas como la retropropagación del error, que propaga los errores de las salidas hacia atrás a través de la red para ajustar los pesos de manera gradual. En la actualidad, existen diversas bibliotecas y frameworks de programación de redes neuronales que facilitan su implementación, como TensorFlow, Keras y PyTorch, entre otros (Barricelli and Fogli, 2021). Estas herramientas proporcionan una interfaz de programación de aplicaciones (API) que simplifica el proceso de configuración y entrenamiento de las redes neuronales, así como el despliegue y uso en diferentes aplicaciones.

Entrenamiento Supervisado: Se proporciona a la red neuronal un conjunto de datos de entrada junto con las correspondientes etiquetas o salidas deseadas. La red aprende a partir de estos ejemplos etiquetados, ajustando sus pesos y conexiones internas para mapear de manera óptima los datos de entrada a las salidas esperadas.

Entrenamiento No Supervisado: Implica exponer a la red neuronal a un conjunto de datos ”abiertos” sin etiquetas o conjunto de información de salida explícita. En lugar de tener ejemplos etiquetados, la red encuentra patrones y estructuras ocultas en los datos de entrada por sí misma.

2.2.2. Inteligencia Artificial y Asistente Virtual

La inteligencia artificial (IA) es una disciplina de la informática que busca desarrollar sistemas y programas capaces de emular ciertos aspectos de la inteligencia humana (Ridley, 2022). Consiste en la creación de algoritmos y modelos matemáticos que permiten a las máquinas aprender, razonar, tomar decisiones y comprender el lenguaje natural. Los sistemas de IA pueden analizar grandes cantidades de datos, detectar patrones, realizar inferencias y adaptarse a nuevas situaciones. Una de las aplicaciones más destacadas de la inteligencia artificial es la creación de asistentes virtuales, también conocidos como chatbots o agentes conversacionales (Arrieta et al., 2019). Estos asistentes virtuales son programas diseñados para interactuar con los usuarios de manera natural, a través de texto o voz, y proporcionar información, responder preguntas, realizar tareas específicas y brindar soporte en tiempo real. Los asistentes virtuales utilizan técnicas de procesamiento de lenguaje natural (NLP) y aprendizaje automático para entender y responder a las consultas de los usuarios de manera inteligente (Bustamante et al., 2022).

Los asistentes virtuales tienen aplicaciones en una amplia gama de industrias y sectores. En el ámbito empresarial. En el campo de la salud, los asistentes virtuales pueden proporcionar información sobre enfermedades, medicamentos y citas médicas. En la educación, pueden ayudar a los estudiantes a acceder a recursos y responder preguntas sobre diversos temas. En la industria del comercio electrónico, los AV pueden guiar a los usuarios a través de procesos de compra y proporcionar recomendaciones personalizadas (Janssen et al., 2022). Estos son solo algunos ejemplos, y las aplicaciones de los asistentes virtuales son cada vez más diversas y sofisticadas.

Una de las principales ventajas de los asistentes virtuales es su disponibilidad las 24 horas del día, los 7 días de la semana. Los usuarios pueden acceder a ellos en cualquier momento

y obtener respuestas instantáneas a sus preguntas o asistencia en tiempo real. Esto mejora la satisfacción del cliente y la eficiencia operativa al reducir los tiempos de espera y agilizar los procesos de atención al cliente. Además, los asistentes virtuales pueden manejar múltiples consultas simultáneamente, lo que los hace altamente escalables y rentables para las organizaciones (Krishnan et al., 2019).

Otra ventaja de los asistentes virtuales es su capacidad para aprender y mejorar con el tiempo. Utilizando técnicas de aprendizaje automático, los asistentes virtuales pueden analizar datos de interacciones anteriores y ajustar sus respuestas y comportamiento en consecuencia. Esto les permite adaptarse a los patrones de uso, personalizar las respuestas y ofrecer recomendaciones más precisas a medida que acumulan más experiencia(García-Reina, 2018). Además, los asistentes virtuales pueden integrarse con otros sistemas y fuentes de datos, lo que les permite acceder a información actualizada y brindar respuestas más completas y precisas. Sin embargo, aunque los asistentes virtuales ofrecen muchas ventajas, también presentan desafíos. Uno de los desafíos es lograr un procesamiento de lenguaje natural preciso y comprensivo. Entender y responder de manera precisa y contextualmente relevante.

2.2.3. Procesamiento de Lenguaje Natural

El Procesamiento de Lenguaje Natural (NLP, por sus siglas en inglés) es una rama de la inteligencia artificial que se centra en la interacción entre las computadoras y el lenguaje humano. El objetivo principal del NLP es permitir que las máquinas comprendan, interpreten y generen lenguaje humano de manera efectiva (Lalwani et al., 2018).

El Procesamiento de Lenguaje Natural abarca diversas áreas y técnicas, entre las cuales se encuentran la generación de lenguaje natural (NLG) y la comprensión de lenguaje natural (NLU). La generación de lenguaje natural (NLG) se refiere a la capacidad de las máquinas para producir texto o discurso en lenguaje humano de manera coherente y comprensible. Implica la creación de textos que pueden ser utilizados para responder preguntas, proporcionar información, generar informes, redactar noticias, entre otros usos. Para lograrlo, se utilizan técnicas como el modelado de lenguaje, la planificación del discurso y la generación de texto basada en plantillas o reglas, así como también enfoques más avanzados basados en redes neuronales y aprendizaje automático(Lam et al., 2020). Por otro lado, la comprensión de lenguaje natural se refiere a la capacidad de las máquinas para entender y extraer significado del lenguaje humano. Implica el procesamiento de texto o discurso para identificar palabras clave, estructuras gramaticales, intenciones, emociones y entidades mencionadas, entre otros elementos.

La comprensión de lenguaje natural se basa en técnicas como el análisis sintáctico, el reconocimiento de entidades nombradas y la clasificación de intenciones (Cutinha et al., 2021). Las ventajas del Procesamiento de Lenguaje Natural son numerosas. Permite a las máquinas interactuar de manera más natural y accesible con los usuarios, facilitando la comunicación y mejorando la experiencia del usuario. Además, el NLP tiene el potencial de automatizar tareas que requerirían un esfuerzo significativo por parte de los humanos, como la clasificación y análisis de grandes volúmenes de texto (Bustamante et al., 2022).

2.2.4. Asistente Virtual: Rasa Open Source

Rasa Open Source es un conjunto de herramientas y bibliotecas de software de código abierto diseñadas para facilitar el desarrollo de chatbots y asistentes virtuales con capacidades avanzadas de procesamiento de lenguaje natural y aprendizaje automático. Rasa Open Source proporciona una plataforma completa para la creación de sistemas de diálogo conversacionales basados en texto.

En su núcleo, Rasa Open Source utiliza técnicas de aprendizaje automático y procesamiento de lenguaje natural para permitir a los desarrolladores crear chatbots y asistentes virtuales que pueden entender y responder a las preguntas y solicitudes de los usuarios de manera inteligente y contextualmente relevante. Rasa Open Source al ser su naturaleza de código abierto, permite a los desarrolladores acceder, modificar y contribuir al código fuente de la plataforma. Esto brinda flexibilidad y libertad para adaptar y personalizar las funcionalidades del chatbot según las necesidades específicas de cada proyecto (RasaOpenSource, 2023).

2.2.5. DIET Classifier

Rasa utiliza la red neuronal Dual Intent and Entity Transformer (DIET) classifier (RasaOpenSource, 2023). Una arquitectura multitarea para la clasificación de intenciones y el reconocimiento de entidades, que utiliza un transformador compartido para las dos tareas. La característica clave del clasificador DIET es la capacidad de integrarse con incrustaciones de palabras previamente entrenadas a partir de modelos de lenguaje y combinarlas con características de n rangos a nivel de caracteres y palabras dispersas de una manera plug-and-play (Mishra et al., 2022). DIET muestra alta precisión, supera las representaciones de codificador bidireccional plug and play tradicionales como BERT, y se ha demostrado que es 6 veces más rápido en tiempo de entrenamiento(Bunk et al., 2020). En la sección de diseño se explica el funcionamiento.

2.3. Geoinformación

2.3.1. Geoservicio

Un geoservicio es una entidad tecnológica que proporciona acceso a datos geoespaciales y funcionalidades relacionadas a geoinformación a través de Internet. Se trata de un servicio basado en la web que permite a los usuarios interactuar con información geográfica, como mapas, capas de datos y herramientas de análisis espacial. Un geoservicio puede incluir diferentes tipos de servicios, como servicios de mapas, servicios de geocodificación, servicios de enrutamiento, servicios de geoprocesamiento, entre otros (Cartagena et al., 2010).

2.3.2. Capas

Una capa de mapas, en el contexto de los Sistemas de Información Geográfica (GIS), se refiere a una representación visual de datos geoespaciales superpuestos en un mapa. Una capa de mapas está compuesta por elementos gráficos que representan entidades geográficas como puntos, líneas o polígonos, y se utiliza para visualizar y analizar información geográfica en un contexto espacial. Cada capa de mapas contiene atributos asociados a los elementos geográficos, lo que permite agregar contexto y detalles adicionales a la representación visual. Las capas pueden ser vectoriales o raster. (Santander and Morocho, 2018).

2.3.3. Servicio WMS

Un Servicio WMS (Web Map Service) es un estándar de interoperabilidad utilizado en sistemas de información geográfica (GIS) para compartir y visualizar mapas en entornos web. Proporciona un mecanismo para solicitar imágenes georreferenciadas a través de una interfaz web. El Servicio WMS permite a los usuarios acceder a capas de mapas predefinidas y obtener representaciones gráficas de dichas capas en formato de imágenes, que pueden ser visualizadas y superpuestas en aplicaciones web y software de GIS. Los Servicios WMS son ampliamente utilizados para la visualización y consulta de datos geoespaciales en línea, brindando una forma eficiente de compartir información geográfica en tiempo real.

2.3.4. Servicio WFS

Un Servicio WFS (Web Feature Service) es un estándar de interoperabilidad utilizado en sistemas de información geográfica (GIS) para acceder y recuperar datos geoespaciales vectoriales a través de una interfaz web. A diferencia de los servicios WMS que proporcionan imágenes estáticas de mapas, los servicios WFS permiten el acceso y la manipulación de los datos geográficos subyacentes, como puntos, líneas y polígonos. Los usuarios pueden realizar consultas espaciales y atributivas para recuperar entidades geográficas específicas o conjuntos de datos completos en formato vectorial. Los servicios WFS son ampliamente utilizados para compartir y editar datos geográficos en aplicaciones y plataformas web, lo que facilita la colaboración y el intercambio de información geoespacial de manera eficiente.

2.3.5. Servicios ArcGis

ArcGIS es una plataforma líder en el campo de los Sistemas de Información Geográfica (GIS) desarrollada por Esri. Proporciona herramientas y funcionalidades para la captura, análisis, almacenamiento, gestión y visualización de datos geoespaciales. ArcGIS se compone de software de escritorio, servidores, servicios en la nube y aplicaciones web, lo que permite a los usuarios crear, compartir y utilizar información geográfica en diversos sectores como la planificación urbana, gestión de recursos naturales, servicios públicos, salud y muchas otras áreas (ArcGIS, 2023).

2.4. Infraestructura de Datos Espaciales (IDE)

Una Infraestructura de Datos Espaciales (IDE) se definen como un “sistema de información compuesto por un conjunto de recursos que integra datos, metadatos, servicios web y visores de tipo geográfico”(Hendriks et al., 2012, p.1). Una IDE está compuesta por una combinación de políticas, estándares, tecnologías y recursos humanos que trabajan en conjunto para proporcionar un entorno eficiente y coordinado en el que los datos espaciales puedan ser recopilados, almacenados, compartidos y utilizados de manera efectiva (Hendriks et al., 2012). El objetivo principal de una IDE es garantizar la interoperabilidad, la calidad y la disponibilidad de los datos geoespaciales, permitiendo a los usuarios acceder a información geográfica precisa y actualizada, y utilizarla para la toma de decisiones en diversos ámbitos, como la planificación urbana, la gestión del territorio, la navegación, el análisis ambiental y muchos otros(Izdebski et al., 2021). Al facilitar la integración y la conexión entre diferentes fuentes de datos geoespa-

ciales, una IDE promueve la colaboración entre organizaciones y fomenta la utilización eficiente de los recursos espaciales.

2.5. Docker y Docker-compose

Docker es una plataforma de código abierto que permite la creación, implementación y ejecución de aplicaciones de manera eficiente y confiable utilizando contenedores. Un contenedor Docker es una unidad ligera y portátil que encapsula una aplicación junto con todas sus dependencias y configuraciones necesarias para ejecutarse de manera consistente en cualquier entorno. Esto se logra mediante el uso de la virtualización a nivel de sistema operativo, donde cada contenedor se ejecuta de forma aislada, compartiendo los recursos del sistema subyacente de manera controlada. Docker ofrece un entorno consistente y reproducible, lo que facilita la migración de aplicaciones entre diferentes entornos de desarrollo, pruebas y producción, eliminando las inconsistencias y los problemas de dependencias (DockerInc., 2023b).
Por otro lado, Docker-compose es una herramienta que permite definir y gestionar múltiples contenedores Docker como una aplicación única y coherente. Proporciona una forma sencilla de definir la configuración de múltiples servicios y su relación, permitiendo la especificación de redes, volúmenes y variables de entorno comunes para los contenedores relacionados. Con Docker-compose, es posible crear y orquestar entornos completos de desarrollo o producción con varios servicios interconectados, simplificando la configuración y el despliegue de aplicaciones complejas. Esta herramienta ofrece una manera eficiente de gestionar la infraestructura de aplicaciones basadas en contenedores Docker, mejorando la escalabilidad, la portabilidad y la flexibilidad en el desarrollo y despliegue de sistemas distribuidos(DockerInc, 2023a).

3. Metodología

La metodología de investigación y desarrollo para el Asistente Virtual se basó en un enfoque estructurado y sistemático que abarca desde la comprensión de los requisitos hasta la implementación y despliegue. Se han seleccionado dos metodologías: *Cross Industry Standard Process for Machine Learning* (CRISP-ML) y la metodología propuesta por los autores de la investigación "How to Make Chatbots Productive" de Janssen.

CRISP-ML es un estándar reconocido por la comunidad informática, cuyo propósito es satisfacer las expectativas establecidas de una organización, con el objetivo de mejorar la eficiencia y éxito de los proyectos de aprendizaje automático. Además, proporciona una estructura clara, sistemática e iterativa, lo que significa que se pueden realizar ajustes y mejoras en el modelo a medida que se avanza en el proceso (Studer et al., 2021). Esta metodología está comprendida de 6 etapas:

- Entendimiento de los Datos y el Negocio
- Preparación de los Datos
- Arquitectura y Diseño del Modelo
- Evaluación del Modelo
- Despliegue
- Monitoreo y Mantenimiento

Los autores de "How to Make chatbots productive – A user-oriented implementation framework" (Janssen et al., 2022), presentan una metodología de trabajo específicamente para asistentes virtuales, con un enfoque de adaptación continua a medida que se obtienen nuevos datos o se descubren nuevos desafíos. Esto ayuda a asegurar que se sigan buenas prácticas y se maximice la eficiencia en el desarrollo. Esta metodología describe un proceso de 8 pasos para la implementación de un asistente virtual:

- Consideraciones Preliminares
- Determinación de Uso

- Definición de las características del Asistente Virtual.
- Estructuración de Diálogo (contenido, desarrollo y entrenamiento)
- Desarrollo de prototipo
- Refinamiento
- Evaluación
- Implementación

En esta tesis se propone la combinación de estas dos metodologías, aprovechando las ventajas de cada una de ellas. De CRISP-ML se aprovecha que es un estándar en la comunidad y de Janssen et, se enfoca específicamente a asistentes virtuales. La combinación de estas dos metodologías da como resultado un total de 8 pasos, como se ve en la siguiente tabla 3.1.

Metodología Resultante	Janssen et al. 2022	CRISP_ML
Análisis y Comparación de tecnología de AV	- Consideraciones Preliminares - Determinación de Uso	- Entendimiento del Negocio
Análisis de depuración y preparación de geoinformación		- Entendimiento de los Datos - Preparación de Datos
Arquitectura de Software del AV	- Definición de características del AV	- Arquitectura y Diseño del modelo
Diseño y Modelado del AV	- Estructuración de Dialogo y Desarrollo	- Arquitectura y Diseño del modelo
Entrenamiento y programación de Actions	- Estructura de entrenamiento - Refinamiento	
Despliegue e Integración con la IDE UCuenca	- Implementación	- Despliegue - Monitoreo y mantenimiento
Pruebas de Usabilidad y Experiencia de Usuario	- Evaluación	- Evaluación

Table 3.1: Metodología Resultante para el desarrollo.

Metodología resultante

- Análisis y Comparación de tecnología disponible de asistentes virtuales
- Análisis de metadatos para la categorización, depuración y preparación de geoinformación para el entrenamiento
- Arquitectura de Software del Asistente Virtual

- Diseño y Modelado del Asistente Virtual
- Entrenamiento y Programación de funciones personalizadas
- Despliegue e Integración con la IDE UCuenca
- Pruebas de Usabilidad y Experiencia de Usuario

En primer lugar, se realiza un análisis y comparación de las tecnologías disponibles para los Asistentes Virtuales, lo que permite identificar las opciones más adecuadas para el proyecto en cuestión.

A continuación, se realiza un análisis de los metadatos para una categorización, depuración y preparación de la geoinformación necesaria para el entrenamiento del Asistente Virtual. Lo que implica organizar y limpiar los datos para garantizar su calidad y relevancia en el proceso de aprendizaje y la creación de base de datos espacial.

Luego, se aborda la arquitectura de software del Asistente Virtual, definiendo la infraestructura y los componentes necesarios para su funcionamiento. El diseño y modelado del Asistente Virtual se lleva a cabo considerando las características específicas y los objetivos del proyecto.

Posteriormente, se procede al entrenamiento del Asistente Virtual, implementado funciones personalizadas y optimizando su capacidad de respuesta y entendimiento. Una vez entrenado, se procede al despliegue e integración del Asistente Virtual con la IDE UCuenca, asegurando su correcta implementación en el servidor requerido.

Finalmente, se realizan pruebas de usabilidad y experiencia de usuario para evaluar la eficacia y la satisfacción de los usuarios al interactuar con el Asistente Virtual.

Gracias a esta combinación de metodologías, se logró un enfoque completo y estructurado que garantizó la calidad y el éxito en el desarrollo y despliegue del Asistente Virtual.

3.1. Análisis y Comparación de tecnología disponible de asistentes virtuales

Como punto de partida para el desarrollo de esta tesis se seleccionaron las mejores herramientas para el desarrollo de un asistente virtual y un framework de trabajo que permita cumplir con las consideraciones y requisitos que se exponen a continuación.

El framework de trabajo debe permitir un entrenamiento supervisado para el caso de estudio, ya que el procesamiento de lenguaje natural sigue varias fases como fonética, morfología, sintaxis y semántica. La máquina necesita dividir todo el texto que ingresa al AV en párrafos, oraciones y palabras. Además, debe aprender a reconocer las relaciones entre las diferentes

palabras, extraer información exacta del texto, comprender oraciones en diferentes situaciones y considerar el contexto de la conversación previa.

El AV desarrollado debe cumplir con los siguientes requisitos funcionales y no funcionales:

3.1.1. Requisitos Funcionales

- Integración con la IDE UCuenca: El AV será desplegado y debe llevar familiaridad con la interfaz gráfica de la IDE UCuenca.
- Procesamiento de Lenguaje natural: El AV debe comprender y analizar el lenguaje natural, coloquial y jerga local utilizada por los usuarios, identificando la intención del mensaje detrás de las consultas y generando respuestas coherentes.
- Interacción multiplataforma: El AV será capaz de interactuar con los usuarios a través de diversos equipos, como navegadores y sistemas operativos.
- Adaptación responsiva: El AV será capaz de ajustarse a distintas dimensiones de ventanas y tamaños de pantalla.
- Integración con sistemas externos: El AV será capaz de integrarse con otros sistemas y servicios, en este caso bases de datos espaciales y APIs de geoservicios.
- Personalización y aprendizaje: El AV será capaz de adaptarse a los usuarios individuales, considerando su lenguaje y comportamiento para mejorar su desempeño con el tiempo y realizar ajustes en su funcionamiento.
- El asistente virtual siempre dará una respuesta en relación con el contexto de Ordenamiento Territorial y se verá limitado su accionar a otro tipo de información abierta.

3.1.2. Requisitos No funcionales

- Escalabilidad: El asistente virtual será capaz de manejar un alto volumen de interacciones simultáneas sin degradar su rendimiento.
- Disponibilidad: El asistente virtual está disponible y accesible para los usuarios cuando el visor este activo.
- Rendimiento: El asistente virtual responderá de manera rápida y eficiente a las consultas y solicitudes de los usuarios. Los tiempos de respuesta debe ser máximo de 1 minuto para brindar una experiencia fluida y satisfactoria.

- Usabilidad: El asistente virtual será fácil de usar y comprensible para los usuarios. Contará con una interfaz intuitiva y amigable, que permita una interacción fluida y sin complicaciones.
- Integración: El asistente virtual será capaz de comunicarse con la IDE UCuenca sus geoservicios y gestión de contenido.

Para lograr cumplir con los requisitos funcionales y no funcionales, se realizó un exhaustivo análisis de las ofertas en el mercado más conocidas, principalmente que se encuentre en un formato que el usuario esté familiarizado, estas resultaron en un cuadro comparativo, ver tabla 3.2. Luego de una comparación asertiva y analizando el caso de uso para la situación del proyecto, se recomendó el uso del framework para RASA.

Asistente	Características	Requisitos del Sistema	Idiomas	Precio
Alexa	• Como servicio • Alexa Skills, Servicio programables para inte-gración con otras aplica-ciones o dispositivos. • Interfaz de voz interactiva. • Speech Recognition • Maching Learning • NLU • Text to Speech • Documentación TécnicaActualizada • Alexa for Bussines (Depago)	**Hardware:** • Como servicio **Software:** • Windows • MacOS • Linux • Android 6+ • iOS	• Inglés (Reino Unido) • Inglés (US) • Francés (FR) • Alemán (DE) • Italiano (IT) • Japonés (JP) • Español (ES) • Español (US)	750 horas Gratis 1 mes Gratis COSTO SEGÚN USO
Siri	• El uso de SIRI es exclu- sivo de productos de Ap-ple. • Como Servicio • Integración con el SistemaOperativo de Apple. • Herramientas de desar-rollo • Versiones Beta de los SO. • Permite instalar las apli-caciones en los dispos-itivos con registro en la App Store • Permite publicar y comer- cializar aplicaciones en la App Store con el benefi- cio de 70% en caso de ser venta.	**Hardware:** • Como servicio • MacOS **Software:** • MacOS • Linux • iPadOS • TvOS • iOS • WatchOS	• Alemán • Chino (simplifi-cado) • Chino (tradi-cional) • Coreano • Danés • Español • Francés • Hebreo	Licencia con costode $99 al año. Licencia con un costo de USD 299 al año

Siri	• Dado que el sistema es escalable dentro de la gama de productos de Apple, Ahora el asistente está disponible en más de 1000 millones de iPhonesalrededor del mundo, así como en los Apple Watch,los iPad, los Mac, los HomePod, los coches conCarPlay e iPod, es parte del sistema operativo. • Swift, lenguaje de progra-mación oficial de Apple • Documentación Actual-izada diariamente • Cursos de Desarrollo • Siri Kit: Servicio program-able para integración conotras aplicaciones o dis-positivos. (de pago)		• Inglés • Italiano • Japonés • Neerlandés • Noruego • Portugués • Ruso • Sueco • Tailandés • Turco	

Google Assistant	• Google Actions es una plataforma para desarrol-ladores que permite de- sarrollar software para ampliar la funcionalidad de Google Assistant. • Se usa junto con Di-alogflow. • Integración con aplica-ciones para dispositivos móviles, aplicación web, dispositivo, bot, sistema de respuesta de voz inter-activa. • Text to Speech. • Documentación Ordenada. • Alta Reutilización decódigo.	**Hardware:** • Como servicio • MacOS **Software:** • MacOS • Linux • iPadOS • TvOS • iOS • WatchOS	• Alemán • Chino (simplifi-cado) • Coreano • Danés • Español • Francés • Hebreo • Inglés • Italiano • Japonés • Neerlandés • Portugués • Ruso • Sueco • Turco	El uso de Dialogflow Standard Edition es gratu-ito, pero la cantidad de solici-tudes quese pueden realizar es limi-tada. Para obtener más infor-mación, consulta el artículo Cuotas y límites.

Rasa Open Source	• Open Source • Implementación en nube privada • Construcción a escala • Infraestructura versátil y reutilizable • Aprendizaje Interactivo • Compatible con canales de mensaje instantánea • Skills y APIs Personal-izadas • No hay documentación actualizada • Rasa X está diseñado para implementarse en un servidor y no en una máquina personal o local. Se recomienda la imple-mentación en un servidor porque Rasa X está dis-eñado para permanecer activo continuamente y no para ser detenido o reinici-ado con frecuencia • Curva de aprendizaje muy amplia • HCI Personalizable • No cuenta con Text to Speech • Añadir conocimiento SPACY 3.0 Crear modelo de mata datos con JSON+ Python: https://blog.rasa.com/custom spacy-3-0-models-in-rasa/	**Hardware:** • Mínimo : 2 vCPU • Recomen-dado: 2-6 CPU virtuales • Mínimo: 4 GBde RAM • Recome n- dado: 8 GBde RAM • Recomen-dado: 50 GBde espacio disponible endisco **Software:** • Windows • MacOS • Linux • NodeJS • Python	• Inglés • Italiano • Japonés • Neerlandés • Noruego • Portugués • Ruso • Sueco • Tailandés • Turco	Rasa Open Source: Gratis Rasa X Enterprise : Costos Adi-cionales.

Wit. Ai (Faceook)	• Reconoce palabras y dis-cursos y los pasa al for- mato JSON (JavaScript Object Notation) para que los desarrolladores puedan disponer de ellosen sus aplicaciones. • Utiliza la inteligencia artifi-cial de Facebook Meta. • Fácil de implementar. • Interfaz amigable. • Se necesita una cuenta de Facebook para interacción con el chatbot. • Curva de aprendizajecorta. • Se usa como servicio, noconsume recursos. • Facebook controla todo. • La aplicación obligada-mente debe ser registradaen Facebook.	No consume re-cursos	• Alemán • Chino (simplifi-cado) • Coreano • Danés • Español • Francés • Hebreo • Inglés • Italiano • Japonés • Neerlandés • Portugués • Ruso • Sueco • Turco	Gratis

Table 3.2: Tabla Comparativa de Asistentes Virtuales

Existen varias opciones para el desarrollo de sistemas AV en el mercado, como Alexa de Amazon, Siri de Apple, Asistente de Google, Rasa Open Source y Wit.ai. Cada uno de estos frameworks tiene sus propias características y ventajas.

Alexa de Amazon, Siri de Apple y el Asistente de Google disponen de librerías populares que ofrecen una integración directa con los dispositivos y servicios de sus respectivas plataformas. Estos frameworks se destacan por su capacidad de reconocimiento de lenguaje y su amplia gama de funcionalidades, como: realizar tareas cotidianas, brindar información y controlar dispositivos domésticos inteligentes. A pesar de su gran ventaja al ser de *dominio abierto* su ecosistema es cerrado, lo que no permite es utilizar un conocimiento personalizado y se restringe su uso en algunos dispositivos. También se requiere de una suscripción o pago por consumo de servicio, lo que incumple con el segundo objetivo específico de esta tesis.

Por otro lado, Rasa Open Source y Wit.ai son frameworks de código abierto que ofrecen mayor flexibilidad y personalización en el desarrollo de asistentes virtuales. A pesar de que Wit.ai es bastante utilizado al ser muy potente para procesos de NLP y cuenta con un soporte 24/7, el desarrollo debe seguir subprocesos de programación y validación por Facebook Meta, razón por la que fue descartado.

Rasa Open Source fue elegido como la plataforma de desarrollo para el asistente virtual por ser de código abierto, ofrece mayor transparencia y control sobre los algoritmos y modelos utilizados, lo cual es importante para proyectos que requieren un mayor nivel de personalización y adaptación a necesidades específicas.

Rasa ofrece la capacidad de implementar conocimiento tanto de *dominio abierto* como de *dominio específico*, es decir, permite aprovechar fuentes de información variadas y adaptarse a necesidades específicas. Además, Rasa facilita la integración de paquetes o bases de conocimiento abiertas disponibles en internet, enriqueciendo la funcionalidad y el contenido del asistente virtual.

Otra ventaja significativa de Rasa es su capacidad para crear una base de conocimiento desde cero, permitiendo restringir y definir el área de conocimiento del asistente virtual. En este caso, mediante el entrenamiento y enseñanza del contexto específico de ordenamiento territorial, obtenemos un alto grado de personalización y adaptación a las necesidades del proyecto.

Rasa es altamente versátil en términos de despliegue, ya que puede funcionar en una amplia gama de procesadores, desde PCs personales hasta servidores, siempre y cuando cumplan con los requisitos técnicos necesarios, ver anexo A. Esta versatilidad brinda flexibilidad en cuanto a la infraestructura en la que fue implementado el asistente virtual.

Una ventaja clave de Rasa Open Source es su capacidad para funcionar sin conexión a inter-

net. Esto es especialmente beneficioso en escenarios donde se requiere privacidad o acceso limitado a internet, lo que ha sido aplicado en esta tesis, ya que el asistente virtual puede operar de manera independiente sin depender de servicios externos. En contraste a otras opciones en el mercado que suelen requerir una conexión constante con los servicios de sus proveedores para su funcionamiento.

Los paquetes de Python que se utilizaron en el desarrollo de esta tesis son:

Paquetes de Python para Rasa

- Python: Rasa requiere tener instalado Python 3.6 o superior en el sistema.
- Pip: requiere pip3 y pip -U para gestionar las dependencias de los paquetes de Python.
- Spacy: Rasa utiliza Spacy para procesar y comprender el lenguaje natural. En este caso, se utilizó Spacy-ES para crear el modelo en español.
- scikit-learn: Es una biblioteca para aprendizaje automático de software libre específicamente para Python.
- Otros paquetes importantes: TensorFlow, pandas, NumPy, open web, requests y JSON.

3.2. Análisis de metadatos para la categorización, depuración y preparación de geoinformación para el entrenamiento

Para la preparación de los datos y un mayor entendimiento de la información que se debe procesar, como parte del proyecto principal AVPPGIS, se identificaron las instituciones generadoras de información geográfica relacionada con el tema de ordenamiento territorial para la ciudad de Cuenca. Algunas de las instituciones son: GAD Municipal Cuenca, IGM, Universidad del Azuay, Etapa EP, Instituto Nacional de Estadística y Censos, Ministerio del Ambiente del Ecuador, Ministerio de Agricultura del Ecuador, Sistema Nacional de Información de Tierras, Código Postal en Ecuador, Instituto de Investigación Geológico y Energético (IIGE), Instituto Espacial Ecuatoriano (IEE), Ministerio de Salud Pública (MSP), Empresa Eléctrica Regional Centro Sur, en total 14 organismos.

Se realizó un barrido manual de los geoportales de estas instituciones en busca de geoservicios publicados de acceso abierto. Luego, se compiló la información relacionada con las *capas* disponibles en cada uno de estos geoservicios, proceso que se realizó de forma automatizada, a través del script *recovery layer info*, programado en Python utilizando la librería OWSLib y obteniendo como resultado una matriz con información de más de 2700 capas.

Un extracto del script se puede ver en la figura 3.1 en el que a partir de un enlace WMS, tomando sus metadatos en formato XML se obtiene sus capas y para cada una se obtiene el nombre, título y resumen.

```
from owslib.wms import WebMapService

url = 'http://geoportal.sigtierras.gob.ec:8080/geoserver/sigtierras/wms?'
wms = WebMapService(url)

layers=list(wms.contents)
print("Layers:\n",list(layers))
title=wms['division_politica'].title
name=wms['division_politica'].name
abstract=wms['division_politica'].abstract

print("\n")
print("Title:",title)
print("Name:",name)
print("Abstract:",abstract)
```

```
Layers:
 ['relieve', 'ortofoto_proyecto', 'vias_acc', 'acc_educacion_mas_destino', 'acc

Title: Límites provinciales y cantonales
Name: division_politica
Abstract: CELIR 2013. Límites políticos referenciales del Ecuador Continental
```

Fig 3.1: Script para extraer capas

Definiendo a los datos recuperados como *matriz de Data Recuperada*. Esta matriz es la lista

de las capas disponibles de todos los geoservicios de las 14 instituciones, ver anexo B.1. Estas capas fueron clasificadas en el “Eje de Planificación”, según la guía del SENPLADES, que en el caso del cantón Cuenca son 6 ejes, ver Figura 3.2 (SENPLADES, 2021).

Fig 3.2: Seis Ejes de Planificación

La matriz de información de las capas Data Recuperada fue conformada por las siguientes columnas: Nombre de institución, Título del servicio, enlace de geoservicio, nombre de la capa, título de la capa y resumen. Esta información ayudó a catalogar la capa en su eje correspondiente. Sin cabe recalcar que antes de llevar a cabo el proceso de clasificación, se evidenció que la información de las capas estaba desorganizada y existía una falta de congruencia y relación entre los títulos, el contenido y el resumen de las capas. Además, se encontró que muchos de los títulos, nombres de capas y resúmenes, estaban redactados en lenguaje técnico utilizado en áreas como arquitectura e ingeniería civil. También, dicha información al originarse en diferentes instituciones, se observó una falta de estandarización en el manejo de la geoinformación, se identificaron capas con contenido similar, pero con etiquetas y nombres diferentes, algunas columnas contenían información o códigos que resultaban útiles para las instituciones, pero incomprensibles para usuarios externos.

Como parte del proceso de la preparación de los datos para la creación de la base de conocimiento, se llevó a cabo un minucioso proceso de Minería de Datos con el fin de lograr una depuración, normalización, clasificación de la geoinformación de la matriz Datos Recuperados y crear una sólida base de conocimiento.

Primero se realizó una intervención manual fila por fila, durante este proceso se agregaron dos columnas adicionales: “**Identificador**” y “**Palabras Clave**”.

La columna “Identificador” se utilizó para asignar un nombre más claro a cada capa, basado en su contenido y relevancia para el proyecto. Se consideraron el nombre de la capa, el título y el resumen para crear la etiqueta identificador. Así mismo, aquellas capas que no pertenecían al cantón Cuenca, que no aportaban valor o que presentaban inconvenientes técnicos, fueron etiquetadas como *“no relevante”* y *“no disponible”*. Ver tabla 3.3

La columna “Palabras Clave” se utilizó como una descripción informal del contenido de cada capa, con el objetivo de facilitar una mejor comprensión de su contenido. Esta información fue utilizada posteriormente en el proceso de entrenamiento del Asistente Virtual. Ver tabla 3.3

Nombre Capa	*Título Capa*	**Identificador**	**Palabras Clave**
az_wc_prec_avg_200	az_wc_prec	cuenca_precipitaciones	Azuay precipitaciones y lluvia
m_cobertura_homologado_a	m_cobertura_homologado_a	cuenca_vegetacion_cutivo	cultivos de vegetales y suelos agrícolas
iss_banras	_iss_bananeras	no relevante	ubicación de banaeras en la costa del Ecuador
iee	Aster_2017	no disponible	sin información

Table 3.3: Extracto de columnas en matriz Data Recuperada

Mediante el uso de la herramienta RapidMiner se aplicaron diversas técnicas como: la depuración de datos, filtrado, agrupamiento, seguimiento de patrones, asociación, clasificación, regresión y otras técnicas de minería de datos, como se puede ver en el esquema de la Figura 3.3.

Para la clasificación por los seis Ejes de Planificación se importó la matriz Data Recuperada al software RapidMiner y mediante el componente de filtrado se identificó los registros con las etiquetas **no relevantes** o **no disponibles** para descartarlos. De igual forma se filtró cuando una fila presentaba valores vacíos en cualquiera de sus columnas, revisándolas antes de eliminarlas, con el fin de enfocarse en la información válida y relevante para el caso de estudio. Luego, para el tratamiento de estos datos, se trabajó con la geoinformación que podría llegar a los a usuarios como *Nombres de las Instituciones, Nombres de las capas, leyendas(información de capas cuando están sobre el visor), Identificador y palabras clave*, sin alterar la información necesaria para los procesos técnicos de conexión e interoperabilidad como *Enlaces a instituciones, enlaces de geoservicios, título de servicio* y en el caso de que la institución use tecnología ArcGIS, además del *enlace*, la *identificación de la capa*.

Principalmente, se utilizó el *Identificador*, *palabras clave* y en menor grado de importancia el *nombre de la capa*, usando las herramientas que brinda RapidMiner como seguimiento de patrones, agrupación y regresión, se agregó la columna Eje de Planificación, con las siguientes etiquetas según:

- Asentamientos Humanos y Canales de Relación
- Medio Físico

- Uso y Ocupación del Suelo
- Culturales y Patrimoniales
- Económico
- Sociocultural

Esta clasificación se basó en la guía del SENPLADES y la experiencia profesional de la Arq. Natalia Pacurúcu, quien pertenece al grupo de investigadores del proyecto principal AVPPGIS, ver anexo D.2

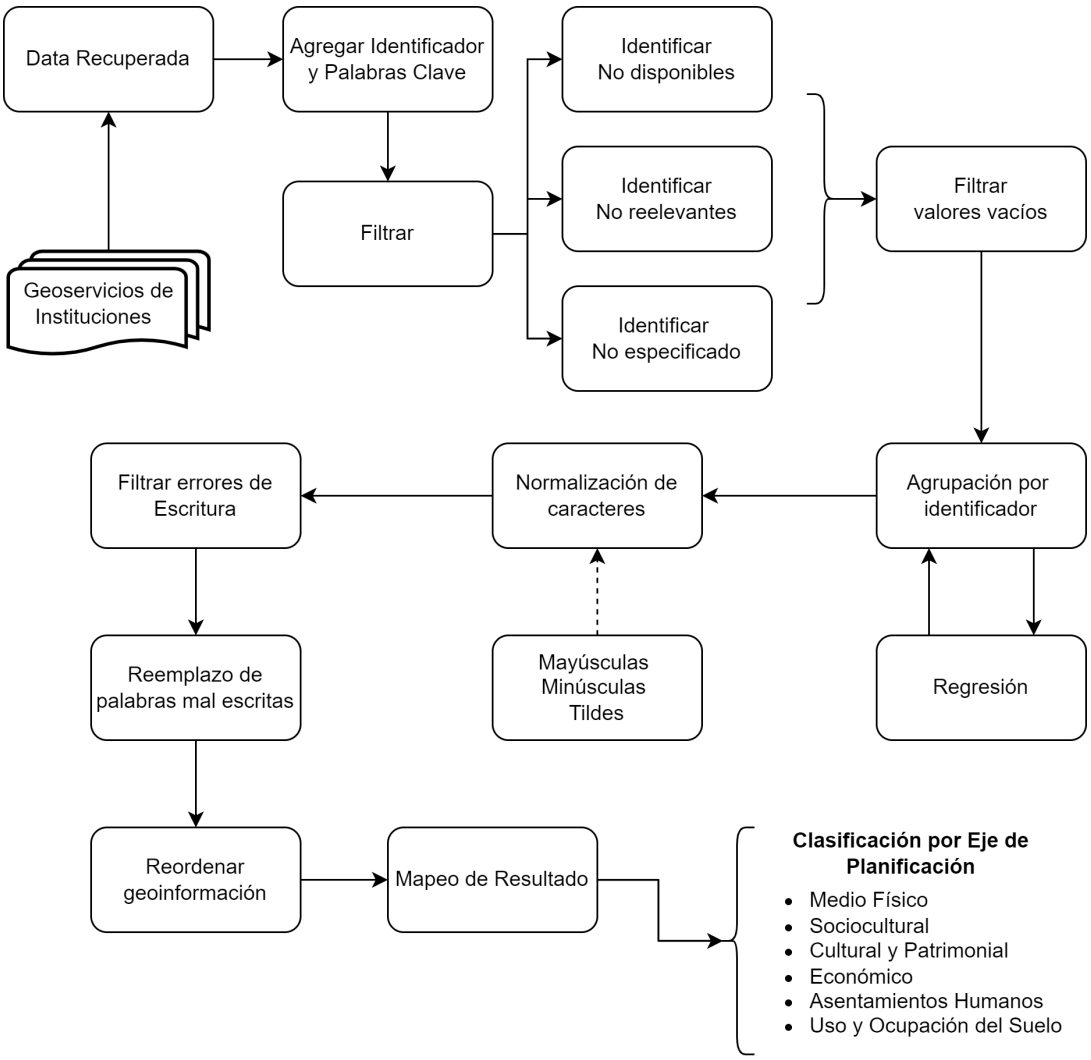


Fig 3.3: Esquema de Minería de Datos realizado

Así mismo, se ejecutó un proceso de *limpieza y normalización de los datos* de la matriz de Data Recuperada, donde se hizo una revisión para corregir espacios en blanco, errores ortográficos y reemplazar palabras mal escritas. Además, se realizó una estandarización de ciertos términos con el objetivo de eliminar redundancias y asegurar que la información que se

muestra en el visor llegue al usuario final de forma precisa y coherente. Durante este proceso se enfatizó en la corrección de errores comunes como tipográficos y gramaticales, así como en la uniformidad en la escritura de términos técnicos y geográficos, como es el caso de los nombres de las capas, ver figura 3.4

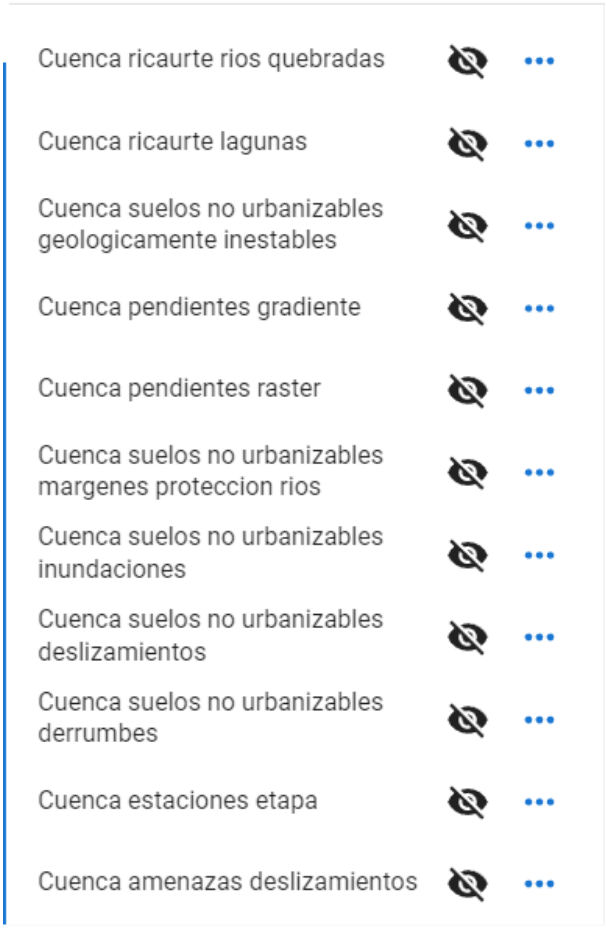


Fig 3.4: Extracto de listado de capas disponibles para el usuario en el visor de la IDE UCuenca.

Además, la limpieza de datos realizada con RapidMiner que se puede ver en el anexo B.3, garantizó la coherencia y consistencia de los datos, mejorando la precisión y la confiabilidad del posterior entrenamiento del Asistente Virtual. Una vez realizado este proceso, se obtuvieron un total de 1075 capas válidas de más de 2700 capas obtenidas al inicio del análisis. En la sección de resultados se puede visualizar en la tabla 4.1 el número de capas por eje.

Este proceso de Minería de Datos dio como resultado una **matriz de geoinformación** con varias columnas, su esquema se puede ver en la tabla 3.4. En esta matriz se basó la construcción de la base de conocimiento del Asistente Virtual que se define en el punto *3.5 Entrenamiento y Programación de funciones personalizadas*, así como la creación de la base de datos espacial de la IDE UCuenca, cuyo esquema se explica en la sección *3.6 Despliegue e*

id	identificador numérico de ítem
Institución	Nombre de la instutución a la que preténsese este ítem
Enlace de Institución	En enlace al portal web de la entidad al que pertenece el ítem.
Título Servicio	El título del geoservicio que contiene este ítem.
Enlace Servicio	El enlace al geoservicio de la institución para consumo de la capa.
Título Capa	El nombre de la capa.
Eje Pertenece	El Eje de planificación al que pertenece la información de la capa.
Identificador	Identificador propio para estandarizar la geoinformación
Palabras clave	Palabras clave para identificar problemas de la capa, explicación del contenido.
Tipo	Tipo de servicio, WMS o ArcGIS.
Estado	Estado de la capa, activo o inactivo
Error	Registra el código de error en el consumo del geoservicio.

Table 3.4: Matriz resultante de geoinformación recuperada.

3.3. Arquitectura de Software del Asistente Virtual

La arquitectura propuesta evolucionó según el avance del proyecto, alcanzando los objetivos luego de dos iteraciones, llegando a obtener una arquitectura óptima y funcional.

Un primer esquema de arquitectura planteado fue crear tres conversaciones guiadas en las que el usuario tenía que seleccionar cuál quería seguir. La primera conversación estaba enfocada directamente a temas de Ordenamiento Territorial, conectada a una API del GAD Municipal de Cuenca. Una segunda conversación en la que se solicita al usuario un mensaje de entrada para realizar una búsqueda entre los metadatos de los geoservicios de las instituciones para ser cargados al visor de mapas y una tercera conversación con el mismo fin, empleando una entrada de texto de preguntas o consultas en un lenguaje técnico. Este primer planteamiento con 3 conversaciones guiadas fue descartado, dado que el GAD de Cuenca tiene en constante cambio su metodología de desarrollo de geoinformación y la solicitud para el acceso a esta información es considerada de tipo sensible, por lo tanto, se negó el acceso a esta API. Así mismo, con esta arquitectura las consultas ingresadas al AV debían ser textuales, es decir, lo que se quería visualizar en el mapa debía estar escrito tal cual está la información en los servidores de las instituciones, sin permitir el uso de sinónimos o variantes del lenguaje empleados en el cantón Cuenca ver figura 3.5.

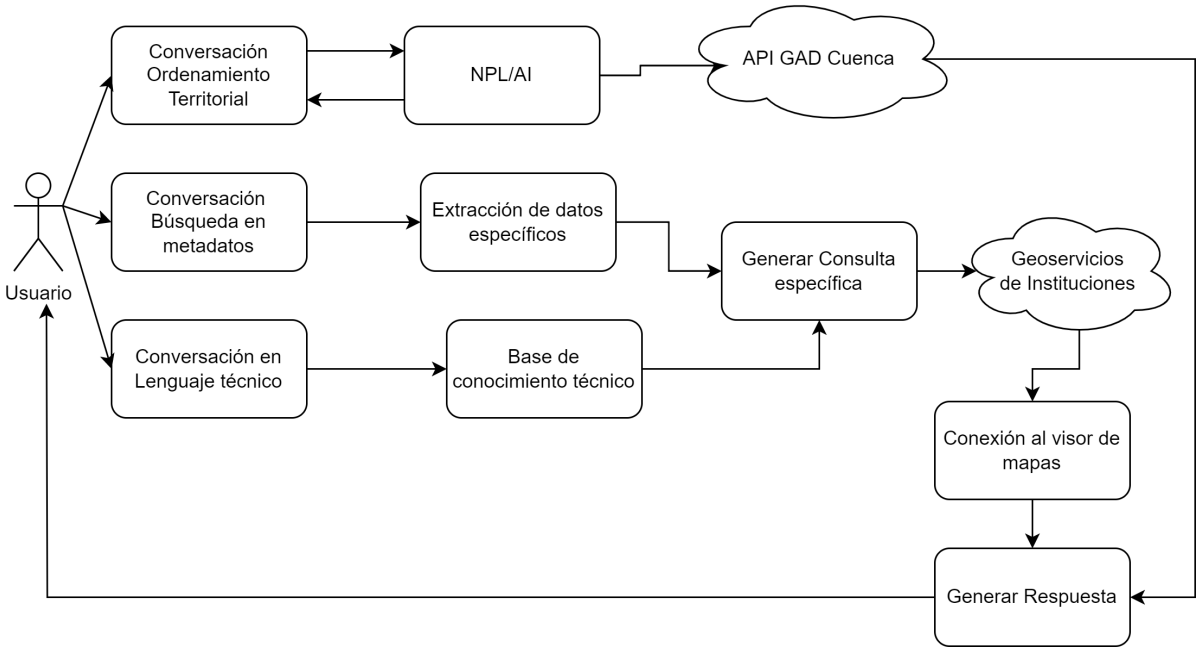


Fig 3.5: Esquema representativo de planteamiento de Arquitectura 1

Por lo tanto, se planteó un segundo esquema de arquitectura que tenía dos conversaciones considerando dos perfiles de usuario: Ciudadano común y técnico. La primera consistía en

una búsqueda de geoinformación solicitada por el usuario a través el Asistente, en el cual según el texto en el mensaje de entrada se procesaba para identificar uno de los seis Eje de Planificación. Sin embargo, la información recuperada incluía todas las capas del eje identificado y se cargaban al visor de mapas. La segunda conversación mantenía el uso del lenguaje técnico en el mensaje de entrada. Ver figura 3.6.

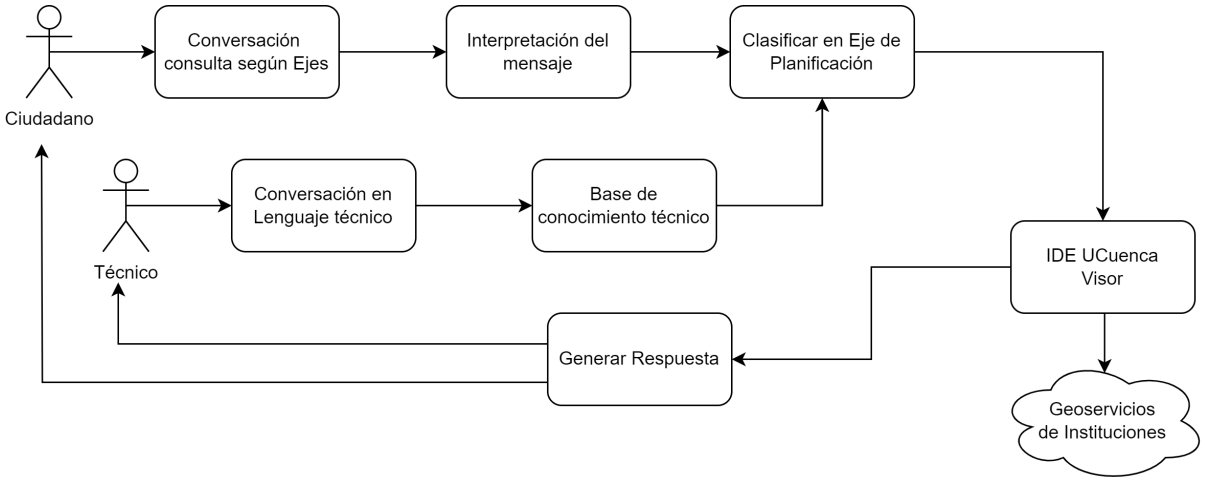


Fig 3.6: Esquema representativo de planteamiento de Arquitectura 2

Al analizar este planteamiento se evidencia que se tiene una gran cantidad de geoinformación dentro de un eje de planificación, por lo que se necesita una mayor granularidad en la clasificación dentro de un mismo eje. Por ejemplo, dentro del eje “Uso y Ocupación del Suelo” están Equipamientos de Salud y Zonas Ganaderas, por lo que debían ser diferenciadas de alguna manera.

Para solucionar este problema se buscó un subnivel que clasificaría la geoinformación dentro de los ejes, llegando a la creación de la etiqueta *Subejes*, que será única, lo que permitió una clasificación más precisa. Para la creación del *Subeje* se tomó la columna “Palabras Clave” de la matriz de geoinformación y se basó en la “Diversificación de Territorios del Consejo Provincial del Azuay” (ConsejoProvinciadelAzuay, 2018), dando como resultando la creación de 132 subejes que se pueden observar en la sección de resultados en la tabla 4.1

Cabe recalcar que para este punto del desarrollo de la arquitectura la matriz de geoinformación (tabla 3.4), estaba plasmada en lenguaje técnico de áreas profesionales como Arquitectura e Ingeniería. Como solución a esto se llevó a cabo una exploración de campo con el objetivo de comprender a profundidad la dinámica de comunicación y el uso del lenguaje por parte de las comunidades involucradas en el proyecto AVPP-GIS. Esta exploración de campo consistió en la interacción directa con los miembros de las comunidades, tomando como referencia el

artículo de Cuesta Meléndez “Comunicación Urbana”(Cuesta and Mélendez, 2017).

Con el propósito de verificar la funcionalidad y aplicabilidad del asistente virtual se mantuvieron reuniones directas con los miembros de las comunidades de los GADs parroquiales de Baños, San Joaquín y Sayausí, donde se aplicaron entrevistas y observaciones. Así se llegó a recolectar datos empíricos sobre las prácticas comunicativas y el tipo de lenguaje utilizado en diferentes contextos de ordenamiento territorial de estas comunidades. La evidencia de estas entrevistas se puede ver en el anexo C.

Los resultados de estas interacciones con las comunidades fueron dos: el primero es que el ciudadano no maneja los conceptos de lo que es un Eje de Planificación y el segundo es el uso de jergas locales en su habla. Por ejemplo, dentro del medio cuencano existen vías que coloquialmente son llamadas “Autopista” o “Circunvalación”, mientras que en los datos recopilados figuran como “vía estatal E35” o “Panamericana N o S”, los cuales deberían tener un procesamiento y tratamiento diferente. Lo mismo ocurre con el Parque Nacional Cajas, que necesitan un enfoque más granular que facilite la identificación y búsqueda de información específica de ese territorio, brindando a los usuarios una experiencia de consulta más eficiente y personalizada.

Así mismo, se comprendió que la consulta ingresada al AV en lenguaje técnico debía llegar a la misma información con una consulta formada en lenguaje natural o coloquial. Por ejemplo, si un arquitecto busca “red de Transdistrubución”, es lo mismo lo que un ciudadano común entiende por “cables de red eléctrica” o coloquialmente llamados “postes de luz”.

Para solucionar estos problemas, se rediseñó toda la arquitectura alrededor del mensaje de ingreso por parte del usuario al asistente virtual, es decir, cuál es el mensaje y la intención de consulta o búsqueda de geoinformación que este contiene, procesándolo sin importar el uso de lenguaje técnico, lenguaje natural y jerga local en el mensaje de entrada.

De este mensaje de entrada se identifica y se extrae la geoinformación y el lugar que el usuario quiere ver en el mapa, con estos datos el asistente virtual recupera la geoinformación de la base de datos espacial, verifica la disponibilidad del geoservicio de la institución y carga los datos geoespaciales al visor de mapas de la IDE UCuenca.

Finalmente, genera una respuesta en la ventana del asistente con un enlace al visor de los datos geoespaciales con la respuesta y dejando siempre abierta la posibilidad de ingresar otro mensaje, logrando así una conversación fluida entre el asistente y el ciudadano.

Esta arquitectura se puede ver en la figura 3.7:

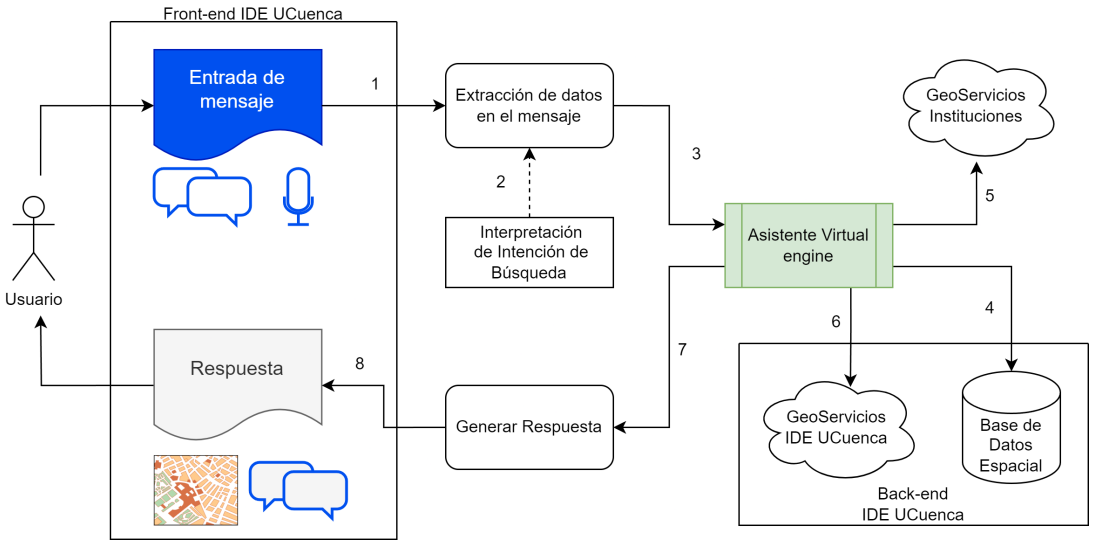


Fig 3.7: Arquitectura del Sistema Asistente Virtual

Con esta arquitectura se logró la integración del Asistente Virtual con los geoservicios de la IDE UCuenca y de las instituciones generadoras de geoinformación y sus tecnologías de implementación.

Flujo de la arquitectura del Asistente Virtual de la figura 3.7:

- Paso 1: Desde el front-end de la IDE UCuenca se envía el mensaje de entrada al asistente virtual mediante un socket de comunicación.
- Paso 2: El AV Engine interpreta y analiza el mensaje, dentro de este paso se extrae la intención de búsqueda que contiene mensaje.
- Paso 3: El AV Engine elabora la consulta y ejecuta los pasos necesarios para formar la respuesta.
- Paso 4: Se recupera la geoinformación según el crimeo de búsqueda, desde la base de datos espacial de la IDE UCuenca, como id, título, tipo, y la URL del geoservicio.
- Paso 5: Verificar la disponibilidad del geoservicio de la institución.
- Paso 6: Seleccionar el visor de mapas y comunicarse con la IDE UCuenca para cargar las capas con la geoinformación recuperada en el paso 5 y 6.
- Paso 7: Crear el enlace y generar respuesta para el usuario.
- Paso 8: Se envía la respuesta al usuario en el front-end de la IDE UCuenca mediante el socket creado en el paso 1.

Para el despliegue del AV en el servidor físico, hay que considerar que los demás servicios de la IDE como el visor de mapas, la base de datos espaciales, entre otros, también están alojados en el servidor de la IDE UCuenca. Por lo que, se ha propuesto una arquitectura basada en contenedores *docker-compose* para lograr un entorno independiente y resiliente. Esta arquitectura se explica y se aprecia en la sección *3.6 Despliegue e Integración con el portal de Infraestructura de Datos Espaciales*.

3.4. Diseño y Modelado del Asistente Virtual

Una vez que los datos han sido depurados y preparados y la arquitectura ha sido planteada, se procede al diseño del Asistente Virtual que consta de 3 fases:

- Diseñar las posibles interacciones y conversaciones que un usuario puede tener con el Asistente Virtual.
- Modular los componentes del Asistente Virtual para crear la base de conocimiento y la red neuronal para la toma de decisiones y crear la conexión de AV con la IDE UCuenca y los geoservicios de las instituciones.
- Diseñar los métodos de comunicación con el AV, tanto de entrada de mensajes como salida de respuestas.

3.4.1. Diseño de conversaciones

Uno de los puntos claves para un funcionamiento fluido y coherente del AV es la correcta compresión y seguimiento de la conversación, para ello se tomaron en cuenta las siguientes consideraciones:

- Un orden lógico de las interacciones y las posibles ramificaciones.
- Definir los mensajes de entrada del usuario y las respuestas del AV en cada paso.
- Determinar los objetivos específicos de la conversación, es decir, qué se espera lograr con cada interacción. Esto puede incluir proporcionar información, realizar acciones o resolver consultas.
- Diseñar respuestas claras, concisas y relevantes para cada situación. Estas respuestas deben ser adaptadas a las intenciones y datos geoespaciales identificados, brindando la información requerida o realizando las acciones solicitadas por el usuario.
- Tener en cuenta el contexto de la conversación, es decir, la información previa o los eventos anteriores que puedan influir en la interacción actual.

Se consideraron 4 interacciones principales: Saludo inicial, realizar una consulta, la sección de ayuda y cuando el usuario solicita información no relacionada. Adicionalmente, se crearon respuestas para preguntas informativas como “¿Quién eres?”, o “¿Cuál es tu propósito?”.

1. **Saludo Inicial, bienvenida y presentación:**

   Al abrir el geoportal de la IDE UCuenca, el Asistente Virtual (AV) dará inicio a la interacción con un mensaje de bienvenida al usuario, proporcionando un contexto sobre su propósito y destacando las funcionalidades disponibles. A través de un mensaje inicial, el AV guiará al usuario brindando instrucciones sobre cómo realizar consultas para obtener la información deseada. Este mensaje introductorio tiene como objetivo familiarizar al usuario con el AV y orientarlo en el uso adecuado de sus capacidades dentro del geoportal. Un segundo mensaje muestra ejemplos de como realizar una consulta, para finalizar la interacción el AV muestra un mensaje preguntando al usuario si desea más instrucciones de uso, con 2 botones para que el usuario pueda dar una respuesta, esta conversación se puede ver en la figura 3.8

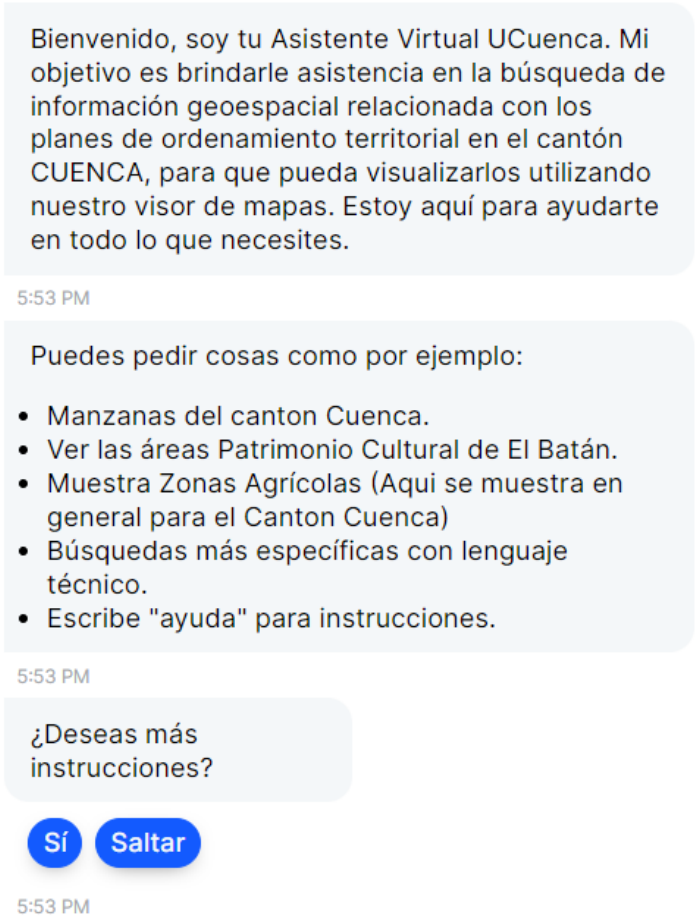


Fig 3.8: Conversación 1: Saludo inicial y presentación

2. **Consulta de geoinformación al Asistente Virtual:**

   Esta es la conversación principal del Asistente Virtual y en la que se basa la arquitectura del sistema, ver figura 3.7.

Al ser la principal función del Asistente Virtual, se tomaron en cuenta las siguientes consideraciones:

- El Asistente Virtual siempre debe estar a la espera de una consulta.
- Diseñar una respuesta clara y concisa para mostrar el resultado en el visor según la búsqueda solicitada por el usuario.
- La respuesta del AV debe contener el Eje de Planificación que se está consultando.
- La respuesta del asistente debe presentar las instituciones de las cuales se está recuperando la geoinformación que se carga al visor.
- El asistente siempre deja la posibilidad abierta de realizar otra consulta.

Dado que esta interacción es iniciada por el usuario, se diseñó un tipo de mensaje de entrada al AV que activa esta conversación. El mensaje que proporciona el usuario debe contener "la consulta de geoinformación" acompañado de un "lugar" de manera opcional, a este mensaje se lo ha denominado **"geo-consulta"**. Un ejemplo de la formación de una geo-consulta se puede ver en la figura 3.9

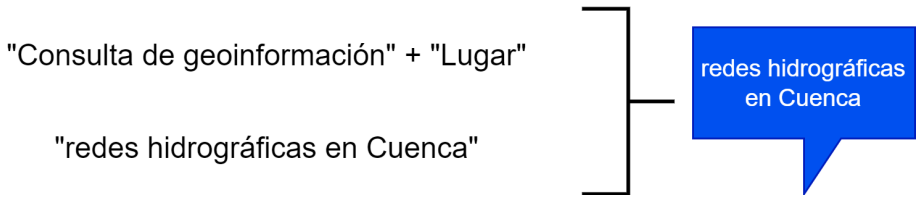


Fig 3.9: Formación de geo-consulta

La **"consulta de geoinformación"** es la intención de búsqueda del usuario, es decir, lo que se desea ver en el mapa relacionado con ordenamiento territorial, por ejemplo, parques, redes de energía eléctrica, población de adultos mayores, etc. El **"Lugar"** son las 37 parroquias del cantón Cuenca. El sector Urbano está formado por las 15 parroquias de: Bellavista, Cañaribamba, El Batán, El Sagrario, El Vecino, Gil Ramírez Dávalos, Hermano Miguel, Huayna Cápac, Machángara, Monay, San Blas, San Sebastián, Sucre, Totoracocha y Yanuncay. El territorio rural se encuentra dividido en 21 Parroquias: Baños, Chaucha, Checa, Chiquintad, Cumbe, El Valle, Llacao, Molleturo, Nulti, Octavio Cordero, Palacios, Paccha, Quingeo, Ricaurte, San Joaquín, Santa Ana, Sayausí, Sidcay, Sinincay, Tarqui, Turi y Victoria del Portete. Al ser opcional el ingreso del lugar, se mostrará el cantón Cuenca en el visor de mapas.

Tomando en cuenta las consideraciones, se puede ver un ejemplo de esta conversación en la figura 3.10

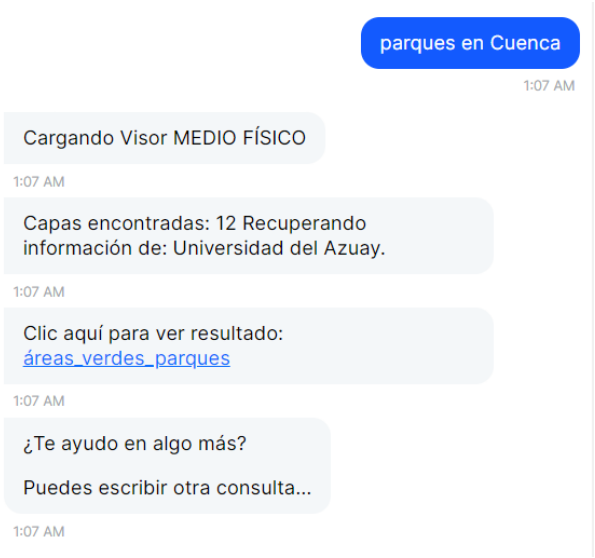


Fig 3.10: Conversación 2: Geo-consulta

3. **Solicitar ayuda o instrucciones de uso:**

   Esta conversación tiene la intención de brindar instrucciones más detalladas del funcionamiento del AV y es iniciada por usuario mediante el ingreso de las palabras clave "ayuda" o "instrucciones". No espera ninguna interacción adicional y vuelve a la espera de que el usuario ingrese una geo-consulta. Esta conversación se puede ver en la figura 3.11

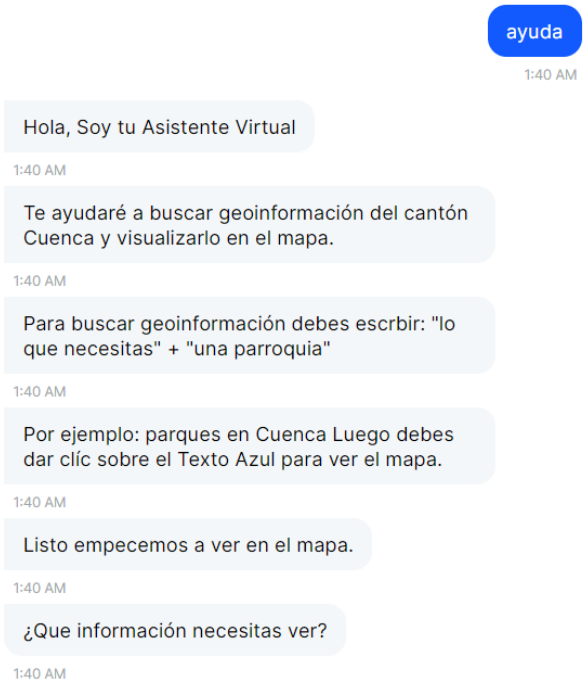


Fig 3.11: Conversación 3: ayuda, instrucciones de uso.

4. **Ingreso de mensaje no relacionado a Ordenamiento Territorial:**

Esta conversación tiene como objetivo responder al usuario cuando ha ingresado un mensaje no relacionado a temas de ordenamiento territorial, se puede ver la conversación en la figura 3.12

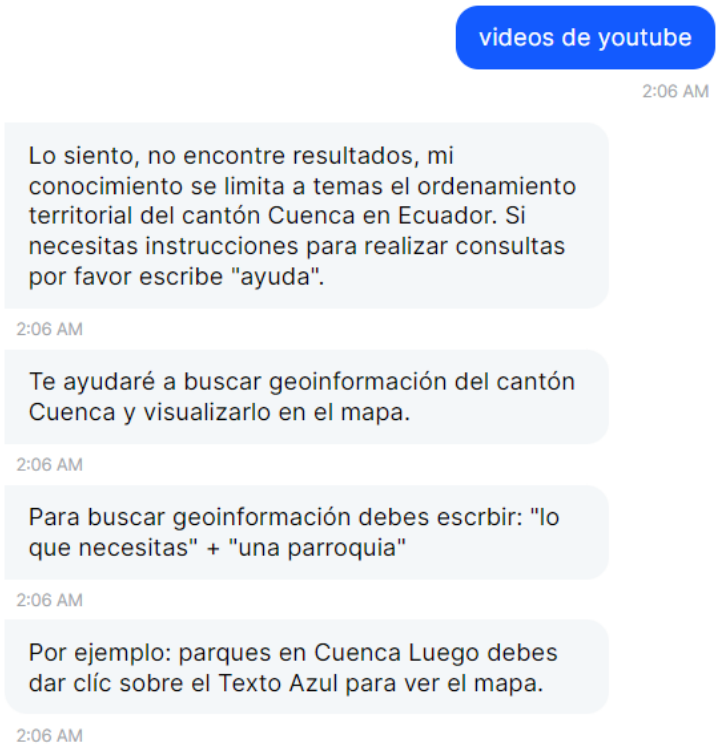


Fig 3.12: Conversación 4: Mensajes no relacionados.

3.4.2. Componentes y Modulación del AV

Para cumplir con el objetivo de las conversaciones, el Asistente Virtual se construyó utilizando varios componentes y 3 módulos principales. Estos componentes están distribuidos entre los 3 módulos llamados Rasa NLU, Rasa CORE y Action Server que conforman el AV Engine de la arquitectura.

Rasa NLU y Rasa CORE

Cuando se diseña un AV basado en el framework Rasa, se tiene Rasa NLU y Rasa Core ambos permiten el Procesamiento de Lenguaje Natural y la toma de decisiones mediante un modelo de red neuronal de capas ocultas. Cabe recalcar que este proceso está realizado para el idioma Español, utilizando características de *Spacy* y *Sklearn* de Python, que fueron mencionados como requisitos en la sección 3.1 Análisis de herramientas.

El módulo Rasa NLU se encarga de procesar, interpretar y dar seguimiento a los mensajes del usuario para obtener una *intención* y extraer *entidades* según el argumento de NLP con el

que ha sido programado. Estos argumentos es el conocimiento agregado al AV mediante el archivo *"nlu.yml"* que se describe en la subsección 3.5.2 del Entrenamiento. Dentro del módulo Rasa NLU se configuran el análisis *Pipelines* y las *Políticas de ejecución* que forman parte de la construcción de la red neuronal y base de conocimiento.

El framework Rasa nos permite crear bases de conocimiento desde cero, esto fue especialmente útil, ya que actualmente no se cuenta con esquema RDF, ontologías, tripletas o paquetes de conocimiento con temas de ordenamiento territorial del cantón Cuenca que se le puedan implementar al asistente virtual. Es la razón de haber elegido los componentes de Pipelines y las Políticas de ejecución.

Para realizar el análisis Pipelines se configuraron ocho componentes:

- **WhitespaceTokenizer:** Que es un separador de palabras, cuando encuentra un espacio, retorna una lista de "palabras" o "tokens".
- **RegexFeaturizer:** Crea una expresión con las palabras ingresadas por el usuario para la extracción de entidades y la clasificación de intenciones.
- **LexicalSyntacticFeaturizer:** Crea características léxicas y sintácticas para un mensaje de usuario para admitir el procesamiento del mensaje y por extraer información textual ingresada en el mensaje.
- **CountVectorsFeaturizer**: Crea una representación de "bolsa de palabras", usando el analizador "chart_wb" de Spacy, para extraer su significado, y crear un vector de números que represente esa palabra, para no tener que procesarla de nuevo. Es decir, cuando se procesa una palabra, crea un vector de números para mayor eficiencia y guarda la palabra esta "bolsa de palabras" y cuando vuelve a ser ingresada ya no realiza el proceso de clasificación, esta es una característica de sklearn, este caso el vector admite un mínimo de 1 palabra y máximo 4.
- **DIETClassifier:** Dual Intent Entity Transformer (DIET), Una vez que se han generado las características para todos las "palabras" y para toda la oración en vectores, pasa al modelo de clasificación de intenciones. Los clasificadores de intenciones asignan una de las intenciones definidas en el archivo de dominio a los mensajes de usuario entrantes, el archivo dominio se describe en la subsección 3.5.5 de Entrenamiento y programación. Este envía una JSON a Rasa CORE ver figura 3.13, que incluye una lista de las posibles intenciones y su nivel de confianza, y una lista de las entidades recolectadas en el mensaje de entrada, y así realizar la siguiente acción.

```
{
    "intent": {"name": "int_ayuda", "confidence": 0.7800},
    "intent_ranking": [
        {
            "confidence": 0.7800,
            "name": "int_ayuda"
        },
        {
            "confidence": 0.1400,
            "name": "goodbye"
        },
        {
            "confidence": 0.0800,
            "name": "construcciones_search"
        }
    ],
    "entities": [{
        "end": 53,
        "entity": "place",
        "start": 48,
        "value": "Cuenca",
        "confidence": 1.0,
        "extractor": "DIETClassifier"
    }]
}
```


Fig 3.13: Contenido JSON que NLU enviar a CORE.

Esta comunicación es automática y solo es visible en consola mediante logs, es decir, no necesita crear algún tipo de servicio de comunicación o creación de archivo.

- **EntitySynonymMapper:** Es el extractor de *entidades* como nombres de personas o ubicaciones de los datos de texto, en este caso la entidad a recuperar es *lugar* de la geo-consulta.
- **ResponseSelector:** Permite seleccionar una de las respuestas predefinidas en el archivo domain, mediante el llamado del CORE.
- **FallbackClassifier:** Esta característica entra en acción cuando se ha detectado un nivel de confianza menor 0.3 en la respuesta de la clasificación DIET y enviar al CORE la orden de ejecución de respuesta predefinida para el caso.

En la figura 3.14 se muestra los 8 componentes implementados en Rasa NLU.

```yaml
language: es

pipeline:
  - name: WhitespaceTokenizer
  - name: RegexFeaturizer
  - name: LexicalSyntacticFeaturizer
  - name: CountVectorsFeaturizer
  - name: CountVectorsFeaturizer
    analyzer: char_wb
    min_ngram: 1
    max_ngram: 4
  - name: DIETClassifier
    epochs: 100
    constrain_similarities: true
  - name: EntitySynonymMapper
  - name: ResponseSelector
    epochs: 100
    constrain_similarities: true
  - name: FallbackClassifier
    threshold: 0.3
    ambiguity_threshold: 0.1
```

Fig 3.14: Componentes para análisis Pipelines

En cuanto a las *Políticas,* se usaron 2 componentes:

- **MemoizationPolicy:** Permite la memorización del flujo de las conversaciones, para seguir aprendiendo de la interacción del usuario. Utiliza el algoritmo de aprendizaje por refuerzo conocido como Aprendizaje por Diferencia Temporal (Temporal Difference Learning) para actualizar y mejorar continuamente el modelo.
- **RulePolicy:** Seguimiento de reglas específicas de la conversación, que se definen en el archivo *rules.yml* que define en el siguiente capítulo.

En la figura 3.15 se muestra las dos políticas implementadas para Rasa CORE.

```yaml
policies:
  - name: MemoizationPolicy
  - name: RulePolicy
```

Fig 3.15: Políticas para Rasa NLU y Rasa CORE

Por defecto, Rasa CORE ejecuta una política de Diálogo Embebido de Transformadores, en inglés *Transformer Embedding Dialogue (TED)*, llamada “TED Policy” es una arquitectura de

multitarea para la predicción de la siguiente acción y el reconocimiento de entidades.

Al mismo tiempo, Rasa CORE es un administrador basado en diálogos, como un modelo de probabilidad de capas ocultas, para ensamblar la respuesta más precisa en función de los mensajes de entrada proporcionados por el usuario, es decir, tomar la decisión de cuál es la respuesta más adecuada según el mensaje de entrada procesado por Rasa NLU. La creación de la red neuronal de capas ocultas y funcionamiento de Rasa CORE se basa en el procesamiento de Rasa NLU y en los archivos story.yml y rules.yml. Estos archivos son descritos en la subsección 3.5.3 Story y 3.5.4 Rules respectivamente.

Rasa CORE llama a la acción adecuada, alojada en el módulo Action Server que se explica a continuación. Una vez ejecutada la acción, llama al componente ResponseSelector para enviar al usuario las respuestas construidas con el Action Server o respuestas predefinidas en el archivo domain.

Action Server: Servidor de funciones personalizas

Dentro de este proyecto se definió como *Action Server* al módulo de un servidor independiente que aloja *"funciones personalizadas"*, las cuales se refieren a funciones, tareas y acciones específicas que los módulos NLU y CORE no puede realizarlas solos, como por ejemplo: enviar correos electrónicos, realizar consultas a bases de datos, llamar a APIs externas o ejecutar cualquier otra tarea personalizada requerida por el asistente dentro del flujo de la conversación. Cuando el CORE identifica que se debe ejecutar una función específica durante la interacción del usuario, envía una solicitud al Action Server, este se encarga de procesar esta solicitud y ejecutar la acción necesaria. Una vez completada la acción, el Action Server envía la respuesta resultante al CORE que la devuelve al usuario final.

Las acciones personalizadas que se definieron para el Asistente son:

- **Conectarse a la Base de datos Geoespacial:** Servicio creado para extraer las capas de geoinformación de la base de datos espacial.
- **Verificar disponibilidad de geoservicio de instituciones:** Realiza una solicitud para verificar la disponibilidad del servicio.
- **Cargar capas al visor de mapas:** Conexión a la IDE UCuenca para cargar las capas recuperadas al visor de mapas.
- **Verificar coordenadas de lugar:** Consumo de servicio de la IDE UCuenca creado para consultar si la información que provee la capa está dentro las coordenadas de la parroquia solicitada.

- **Generar respuesta para CORE:** Generar respuesta con la información del resultado de la consulta: el número de capas cargadas al visor, la institución de donde se recupera la información y el enlace al visor, para la gestión del diálogo en el módulo CORE.

Las acciones descritas anteriormente trabajan en conjunto, el flujo de ejecución se puede ver en la figura 3.16:

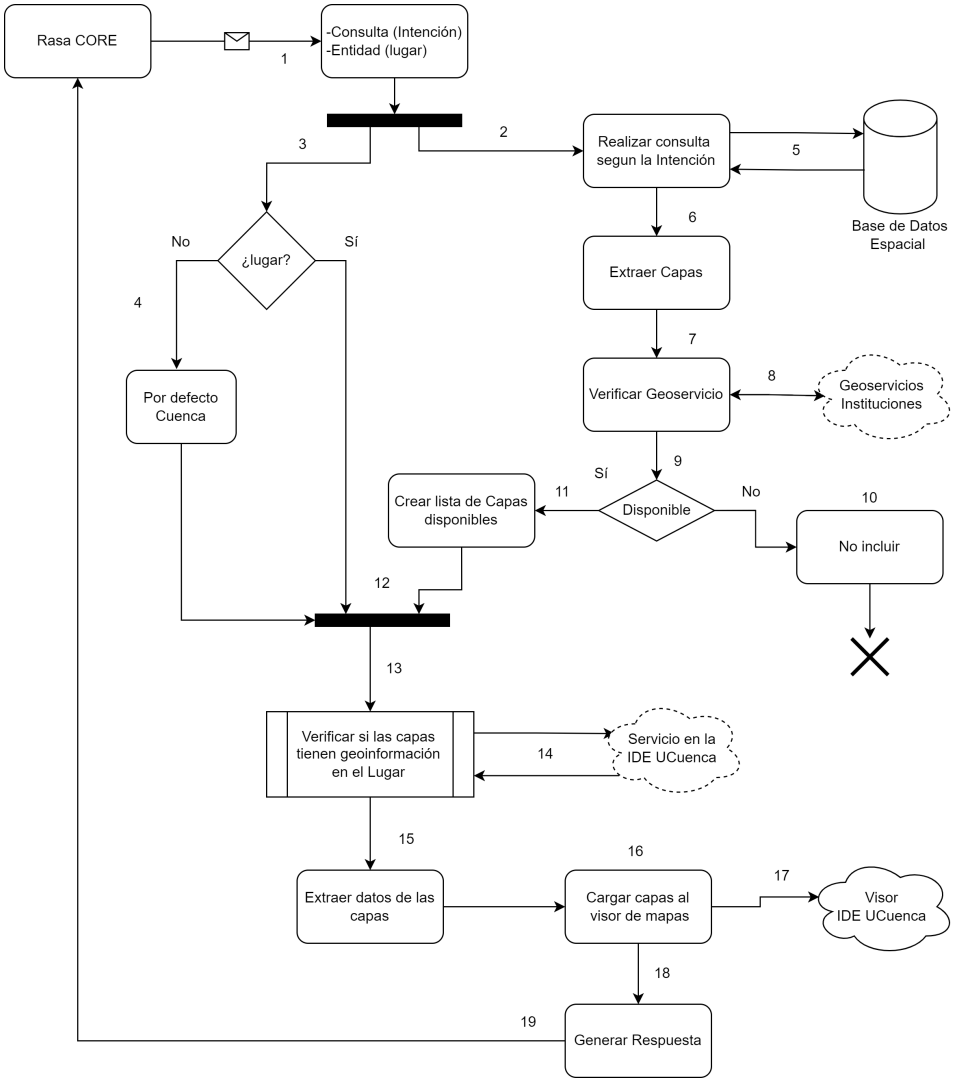


Fig 3.16: Diagrama de Actividades dentro de Action Server

Una vez que Rasa CORE decida llamar al Action Server enviándole la intención de búsqueda y el lugar en forma de entidad, en caso de no contener entidad se cargará el mapa con las coordenadas del cantón Cuenca.

Desde el Action Server, se conecta la base de datos espacial, donde recupera un listado de capas según la intención de búsqueda identificada por NLU y CORE. Luego verifica que el geoservicio de la institución esté disponible para generar un listado de capas disponibles, en caso de que algún servicio de la institución no esté disponible lo descarta.

Dentro de la IDE UCuenca existe un servicio para detectar si existe geoinformación dentro de una parroquia del cantón Cuenca, mediante cruce de coordenadas y polígonos de las capas, el Action Server se conecta a este servicio, enviando la parroquia que el CORE detectó y el listado de las capas disponibles seleccionadas. Este servicio de la IDE UCuenca devolverá un listado de las capas que tienen geoinformación dentro de la parroquia solicitada.

Una vez obtenido el listado final se conecta al geoservicio de la IDE UCuenca para cargar las capas al visor, así mismo se genera una respuesta con el número de capas y las instituciones de donde se las está recuperando, para devolverlas al CORE.

3.4.3. Procesamiento de mensaje mediante los módulos y componentes

Para analizar la **geo-consulta** hacemos uso del tokenizador para separar el mensaje ingresado palabra por palabra, ver figura 3.17.

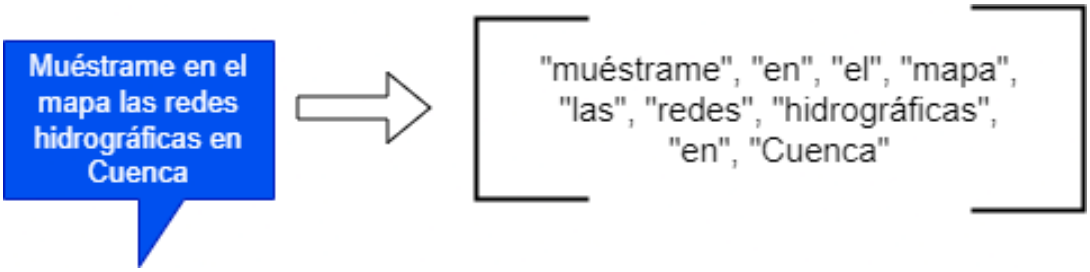


Fig 3.17: Tokenización de un mensaje de entrada

Luego, mediante los diferentes componentes de Pipelines, Rasa NLU convierte los mensajes no estructurados o mal escritos provenientes del usuario para que puedan ser clasificados según el formato de la base de conocimiento, mediante el componente DIET, como se puede ver en la figura 3.18

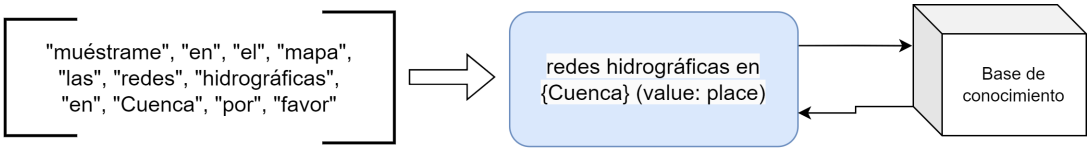


Fig 3.18: Extraer frase importante para clasificación

A lo largo de este proceso, Rasa NLU y Rasa CORE trabajan en conjunto como un componente integral del sistema, para obtener la intención de búsqueda y extraer información clave del mensaje, como se ve en la figura 3.19. En este punto entran en funcionamiento las políticas definidas anteriormente que llevan a cabo un subproceso de memorización donde la red neuronal continúa aprendiendo a medida que se reciben nuevas entradas y se generan salidas.

Este proceso de aprendizaje se basa en la retroalimentación proporcionada por los usuarios y la experiencia acumulada a lo largo del tiempo.

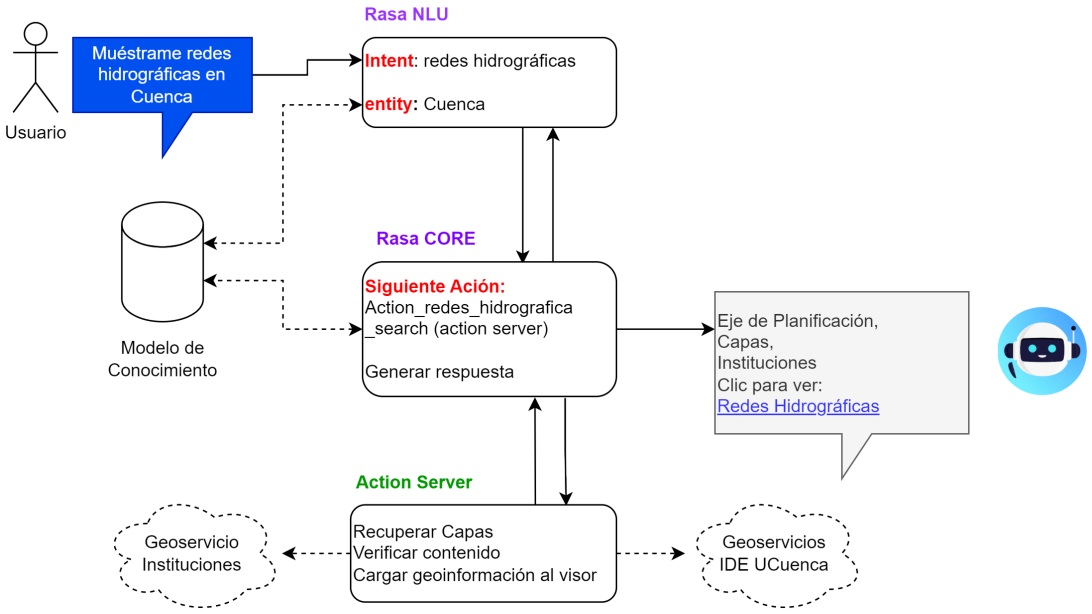


Fig 3.19: Diagrama de procesamiento de Mensaje con componentes de Rasa

3.4.4. Métodos de interacción con el Asistente Virtual

El diseño de los métodos de interacción y comunicación del Asistente Virtual (AV) se lleva a cabo teniendo en cuenta varias consideraciones importantes:

- Las formas en las que el usuario puede interactuar con el Asistente Virtual, se consideró texto y voz.
- Mediante qué servicio de la IDE UCuenca será ofertado el AV.
- El lenguaje de programación a usar para front-end y back-end.
- Protocolos de comunicación entre el front-end y el back-end.
- Manejo de múltiples conversaciones.
- Diseño de la interfaz gráfica con familiaridad con el Proyecto AV-PPGIS

Comunicación entre front-end y back-end

Para asegurar la comunicación fluida entre el front-end y el back-end, se ha seleccionado Socket.IO como protocolo de comunicación. Esta elección permite una comunicación en tiempo real y bidireccional.

Para abordar el manejo de múltiples conversaciones, se ha implementado un sistema de sesiones en el que cada dispositivo que se conecte al AV a través de Socket.IO obtendrá un ID único para garantizar que cada usuario pueda tener conversaciones independientes y no se mezclen los mensajes entre distintos usuarios. Ver figura 3.20

```
socketio:
  user_message_evt: user_uttered
  bot_message_evt: bot_uttered
  session_persistence: true
```

Fig 3.20: Configuración de Socket.IO en Rasa

Control de múltiples conversaciones

Adicionalmente, se ha diseñado una funcionalidad para reiniciar la conversación si el usuario permanece más de 1 minuto sin interactuar con el AV. De esta manera se aprovecha la eficiencia del AV tomando la siguiente interacción como una nueva conversación. Sin embargo, este no afecta al seguimiento de las conversaciones debido a la *Política de memorización*. Esta configuración se puede observar en la figura 3.21

```
session_config:
  session_expiration_time: 60
  carry_over_slots_to_new_session: true
```

Fig 3.21: Configuración de tiempo de Interacción con el AV.

Diseño de Interfaz Gráfica y Usabilidad HCI del Asistente Virtual

Dentro del proyecto AVPPGIS, se definió que el AV se desplegará en el geoportal web de la IDE UCuenca, el cual también se encontraba en una etapa de rediseño, por lo que se trabajó conjuntamente con el equipo encargado de esa área en los posibles diseños de front-end, quien resolvió utilizar el lenguaje de programación Angular. El diseño de la interfaz gráfica del Asistente Virtual se centró en la facilidad de uso y la capacidad de respuesta a las consultas de los usuarios dentro del geoportal. Se considerarán texto y voz como métodos de comunicación para facilitar a los usuarios interactuar con el AV de acuerdo a sus preferencias.

Según Houston (2017) las plataformas de mensajería para consumidores como Facebook Messenger, Snapchat, Kik; las empresas tecnológicas como Slack, Microsoft Teams, y Cisco Sparks, además de la experiencia móvil, son las responsables de haber creado un estándar

HCI para el funcionamiento y diseño de los chatbots, asistentes y agentes conversacionales (Houston, 2017). Otros autores basan el diseño de la interfaz gráfica en el contexto donde trabaja el asistente, por ejemplo en (University of Goettingen et al., 2022), se propone el desarrollo de la interfaz gráfica aplicando *"Principios de diseño para asistentes basados en procesos en entornos de trabajo digitales"*, dentro de estos principios se menciona que los asistentes deben brindar interfaces de usuario humanizadas y fáciles de manejar, que se puedan usar de manera receptiva e independiente del dispositivo, con la sensación de un contacto personal para captar la atención del usuario con lenguaje o señales. Además, se explica que el asistente debe ofrecer flexibilidad para no interferir en otros procesos que no permitan cumplir los objetivos de la plataforma, en este caso permitir una correcta visualización del visor.

Por lo tanto, se consideraron los siguientes aspectos para el diseño de la interfaz gráfica del Asistente Virtual:

- La interfaz gráfica del asistente debe llamar la atención del usuario, mediante señales y mensajes.
- La interfaz gráfica del asistente no debe impedir el funcionamiento de la IDE UCuenca y no debe interferir con sus tareas.
- La interfaz gráfica del asistente no debe impedir la visualización del visor de mapas y sus capas.
- La interfaz gráfica del asistente debe llevar coordinación con la interfaz gráfica de la IDE, como paleta de colores y tipografía.
- La interfaz gráfica del Asistente Virtual debe ser responsiva para adaptarse a cualquier tamaño de pantalla.
- Mientras el asistente está ejecutando un proceso, debe comunicarlo al usuario.

Para plasmar las consideraciones descritas, el asistente se desarrolló como una sección independiente dentro del front-end de la IDE, utilizando Angular y los estilos del framework Bootstrap5 para garantizar que tuviera un diseño receptivo y conseguir familiaridad con la paleta de colores de la IDE UCuenca.

El diseño de la interfaz gráfica del Asistente Virtual se basó en los estándares HCI. En primer lugar, definió la ubicación en la pantalla. Se creó una ventana flotante que se despliega, donde se muestra la conversación en forma de mensajes de texto, diferenciando los mensajes de interacción con colores diferentes, azul para el usuario y gris para los del asistente. Para no

interferir y otorgar una mejor visualización del mapa, esta ventana cuenta con la opción de ser minimizada en un círculo en la misma posición, con un icono que identifica la ventana del Asistente Virtual. Con esto se logra un diseño estándar en Chatbots y Asistentes Virtuales, conocidos y manejados por los usuarios, como se observa en la figura 3.22:

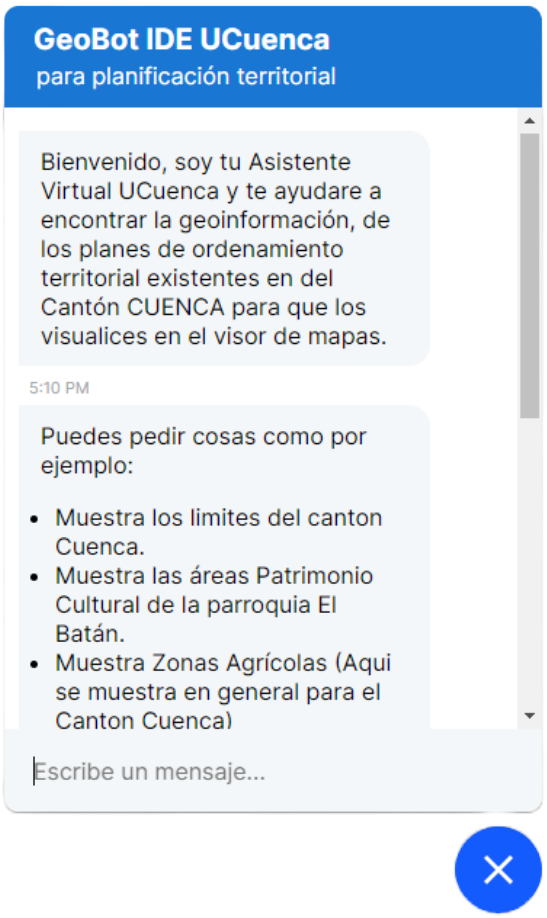


Fig 3.22: Diseño front-end del Asistente Virtual

Para llamar la atención del usuario y destacar la función del Asistente Virtual, se creó un pequeño recuadro flotante para presentarlo, como se puede ver en la figura 3.23:

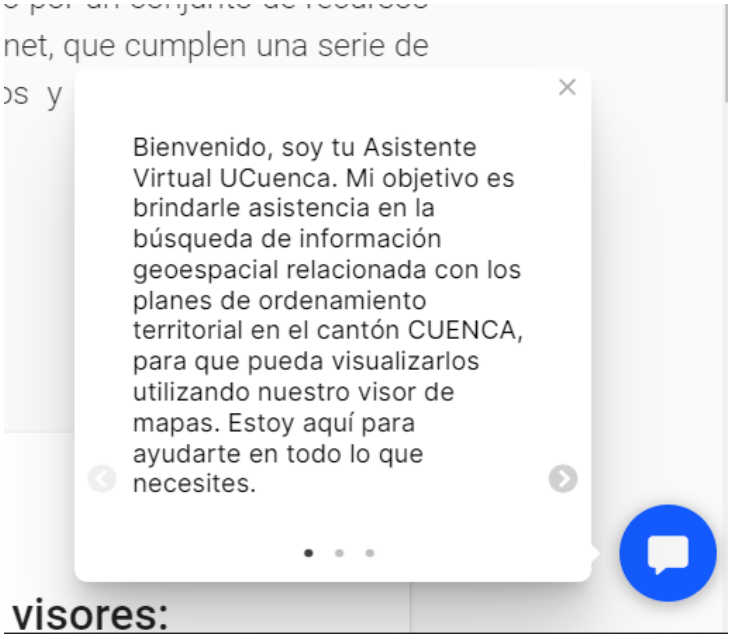


Fig 3.23: Recuadro flotante del AV para llamar la atención del usuario.

Para mejorar la usabilidad y resaltar la integración del AV, este recuadro se activa al momento de acceder al geoportal de la IDE UCuenca, también muestra los mensajes que el AV responde en caso de que este minimizado, como se puede en la figura 3.24

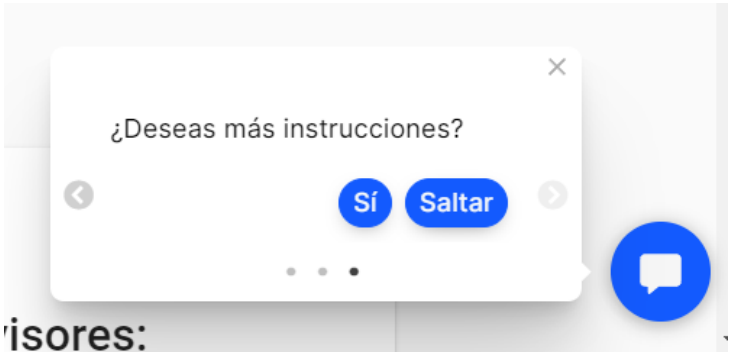


Fig 3.24: Recuadro flotante del AV para mostrar mensajes.

3.5. Entrenamiento del Asistente y Programación de Funciones Personalizadas

Para una mejor comprensión se recomienda una lectura del Glosario, es necesario tener en cuenta estas definiciones:

- **Intent:** Intención de búsqueda del usuario, como parques, vías, redes, etc.
- **Entity:** Variable que se puede entrada de forma textual del mensaje de entrada
- **Evento:** Interacciones entre el Asistente Virtual y el Usuario.
- **Action:** Evento que contiene la orden de ejecutar una función personalizada en el Action Server.

3.5.1. Creación de la base de conocimiento y Modelo IA

Como se mencionó en la sección 3.4.2 Componentes y Modulación del AV, no se cuenta con una base de conocimiento, esquema RDF u ontologías con el tema de ordenamiento territorial para el cantón Cuenca, por lo tanto, se creó una base de conocimiento desde cero.
Para la generación de esta base y de la red neuronal para la toma de decisiones se siguen varios pasos. En primer lugar, se crearon ejemplos de diálogos que representen las diferentes *intenciones de búsqueda* y *entidades* que el asistente virtual debe reconocer e interactuar. La formación de este conjunto de ejemplos contiene frases representativas y diversas que abarca las diferentes formas en las que los usuarios, tanto técnicos como ciudadanos comunes, pueden expresar sus intenciones de búsqueda, por ejemplo:

- Quiero ver parques en Monay.
- Muéstrame redes de transdistribución.
- densidad de adultos mayores en Cuenca.
- ¿Cómo funciona la IDE?
- relieve de El Cajas.
- Autopistas en Cuenca.

Estos ejemplos se crearon manualmente, obtenido un banco de frases con base en los subejes de planificación, sinónimos, preguntas y diferentes maneras en las que un usuario puede

referirse a la geoinformación que desean ver en el mapa. Dentro de estos ejemplos se consideró a los lugares especiales y jerga local, y se implementaría mediante los componentes del módulo NLU.

Para implementar la base de conocimiento al AV se siguió un proceso de *entrenamiento supervisado*, en este caso, se cuenta con 1056 ejemplos de frases de entrada y se utilizaron los 132 Subejes de planificación como salida. El objetivo es que el modelo adquiera la capacidad de clasificar una entrada en una **Intent**. Esta Intent se define como la intención de búsqueda de geoinformación en función de los Subejes de planificación. Cada Intent contiene entre 5 y 12 frases de ejemplo.

A continuación, se crean los archivos de entrenamiento correspondientes> Para definir las entradas y salidas se crea *nlu.yml*. Para el modelo de IA que toma las decisiones en el módulo Rasa CORE se crean los archivos *stories.yml y rules.yml*, finalmente para definir el área de conocimiento se construye el archivo *domain.yml* donde se definen las intenciones de búsqueda, las entidades, actiones personalizadas, respuestas predefinidas y generación de respuestas.

3.5.2. nlu.yml

Este archivo es el responsable de la creación de la base de conocimiento que se procesa en Rasa NLU, definiendo las frases y ejemplos de cada uno de los subejes etiquetados por una *Intent*, dentro de los ejemplos describimos la *entidad* siguiendo la estructura, documentación y versión de Rasa.

Cuando se crea el archivo nlu.yml, el framework RASA lo detecta y utiliza los componentes descritos en la sección 3.4.2 para recoger todos los ejemplos, Intenciones y Entidades definidas en el archivo. Un fragmento de programación de este archivo, se puede ver en la figura 3.25.

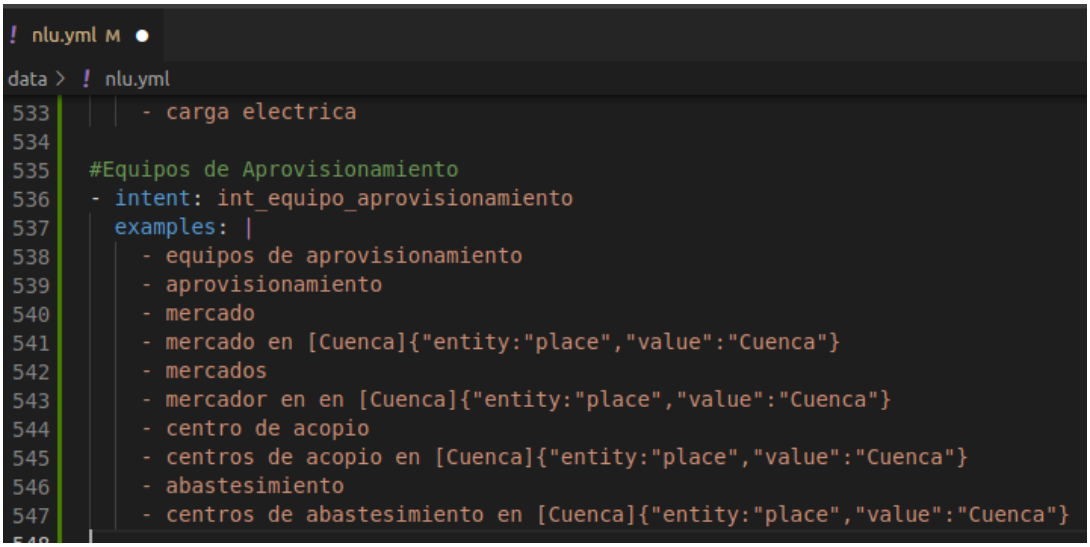


Fig 3.25: Fragmento de código de entrenamiento de entradas y salidas para base de conocimiento.

Como se puede observar, la *Intent* es “int_equipo_aprovisionamiento”, que en términos generales, son mercados de productos de consumo humano, donde se puede ver los ejemplos tanto en lenguaje coloquial como en el lenguaje técnico.

3.5.3. Stories.yml

El siguiente paso es definir un flujo de conversación para el entrenamiento y construcción del modelo de IA, el cual se basa en las interacciones de diálogo entre el usuario y el asistente virtual.
Como por ejemplo:

- AV: Bienvenido, quieres ver las instrucciones de uso de la IDE?
- Usuario: Si
- AV: Aquí tiene el curso de introducción al uso del AV, http:cursoide.com
- Usuario: Gracias.
- AV: ¿Te puedo ayudar en algo más? Escribe una consulta.

Cada una de estas interacciones representan una serie de *eventos* que describen el flujo de una conversación y las acciones que el asistente debe tomar en respuesta a las entradas del usuario.
Cada evento puede ser una acción del usuario, una acción del asistente o una acción de eventos de sistema. Se organizaron los eventos en orden cronológico para capturar una dinámica

de conversación, como se puede ver en el fragmento de programación de una Story en la figura 3.26

```
- story: saludo_inicial
  steps:
  - intent: greet
  - action: utter_msj_bienvenida
  - action: utter_msj_ejemplos
  - action: action_buttons_sino

- story: story_resp_buttons_affirmacion
  steps:
  - intent: affirmacion
  - action: action_open_index
  - action: utter_te_ayudo_algo_mas

- story: story_resp_buttons_int_saltar
  steps:
  - intent: int_saltar
  - action: utter_listo_veamos_el_mapa
```

Fig 3.26: Fragmento de código de programación de una Story

Cuando el Modelo de IA recibe un mensaje de entrada, utiliza las Stories para predecir la próxima acción que el asistente debe realizar. El proceso de entrenamiento con Stories implica iterar varias veces sobre los datos de entrenamiento y ajustar los parámetros del modelo, obtener la respuesta esperada. Durante el entrenamiento, el modelo aprende a mapear las intenciones y las acciones del usuario a las respuestas y acciones apropiadas del asistente.
Se definieron 6 Stories para determinar el flujo de conversación:

- **Saludo inicial:** saludo inicial del asistente al usuario.
- **Instrucciones de Uso:** El usuario pregunta como funciona el AV.
- **Búsquedas de geoinformación:** Para la ejecución de las geoconsultas.
- **Agradecimiento:** Cuando el usuario responde de manera agradecida.
- **Preguntas relacionadas:** cuando el usuario ingresa un mensaje de preguntas informativas, como: ¿Quién eres?, o ¿Cuál es tu propósito?.

3.5.4. rules.yml

El archivo “rules.yml” es utilizado para definir reglas de conversación que permiten guiar y controlar la interacción del asistente virtual con los usuarios. Las reglas se definen en forma de patrones y acciones, especificando condiciones y respuestas predefinidas. Estas reglas se utilizan para establecer directivas específicas sobre cómo debe responder el asistente virtual en determinadas situaciones.

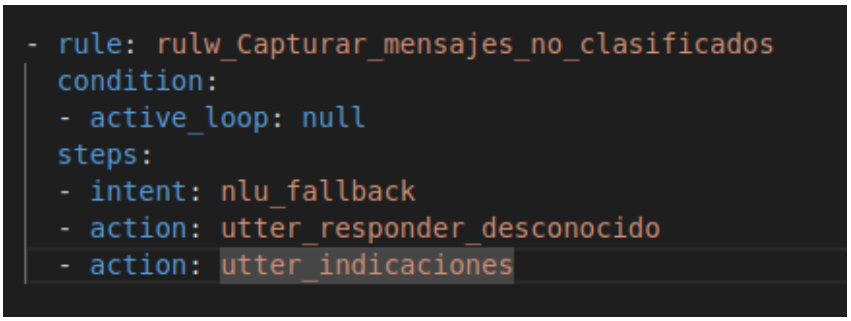


Fig 3.27: Fragmento de programación de reglas.

En esta figura se puede apreciar la creación de la regla que debe seguir el AV cuando detecta una entrada que no está en la base de conocimiento, también dentro de esta regla se aprecia un control de bucles para anularlos cuando son infinitos. La siguiente acción es generar respuesta a una consulta sobre información que no está relacionada a ordenamiento territorial.

3.5.5. Domain.yml

El siguiente paso es la creación del archivo domain.yml. Este archivo define el *dominio* en el que el asistente trabaja, es decir, el área de conocimiento. Aquí se definen todos los *Intent, Entities y Actions* con los que el AV puede trabajar. También, se incluyeron plantillas para las respuestas predefinidas que el asistente puede enviar al usuario, como por ejemplo, saludos iniciales, preguntas e instrucciones de lo que el Asistente Virtual puede hacer en la IDE.
Lo primero que encontramos en el domain.yml es el listado de las *Intent* que se programaron en el fichero nlu, como se ve en la figura 3.28

domain.yml

```yaml
version: "3.4"

intents:
  - greet
  - goodbye
  - affirm
  - deny
  - mood_great
  - bot_challenge
  - affirmacion
  - int_saltar
  - int_zona_agricola
  - int_zona_insdustrial_grande
  - int_tipos_de_suelo
  - int_relieves
  - int_riesgo_caidas
  - int_areas_protegidas
  - int_cajas
  - int_infraestructura_vial
  - int_alcantarillado
  - int_redes_hidrografias
```

Fig 3.28: Extracto de listado de Intents en domain.yml

A continuación definimos las *entities* que ocuparemos para recuperar el lugar, ver figura 3.29:

```yaml
entities:
  - place
  - button_value
  - note
```

Fig 3.29: Entity en doamin.yml

Por último, dentro de domain.yml incluimos las acciones con las que el CORE tendrá interacción, así como las respuestas predefinidas que se enviarán como respuesta al usuario, como se puede ver en la figura 3.30, cabe recalcar que las respuestas predefinidas también se consideran acciones personalizadas.

```
utter_te_ayudo_algo_mas:
- text: "¿Te ayudo en algo más?
        \n
        \nPuedes escribir otra consulta..."
- text: "Si necesitas instrucciones escribe: ayuda
        \n
        \nEscribe otra consulta..."

utter_cargando_visor:
- text: "Cargando Visor {eje}."

utter_responder_desconocido:
- text: "Lo siento, no encontre resultados, mi función
- text: "Lo siento, no tengo conocimiento sobre ese te
- text: "Lo siento, no encontre resultados, mi conocim
- text: "Como modelo de IA, solo tengo acceso a geoinf
- text: "Lo siento, no puedo recuperar información que
```

Fig 3.30: Listado de respuestas y en domain.yml

Las respuestas que se entregan al Usuario son producto del trabajo en conjunto del los módulos Action Server y Rasa CORE, basándose en el diseño de la conversación 2 que se encuentra en la sección 3.10 Diseño y Modulación. Una vez ejecutada la acción el Action Server devuelve al CORE el número de capas, listado de las instituciones y el enlace al visor con las capas cargadas, donde el CORE ensambla las respuestas y las envía al usuario.

3.5.6. Programación de funciones personalizadas

Como se explicó en la sección 3.4.2 Diseño y Modulación, las funciones personalizadas se ejecutan dentro del módulo Action Server y este módulo funciona como servidor independiente, dentro de este se ejecuta *RASA_SDK* que es el kit de desarrollo de software de Rasa, que permite al framework RASA reconocerlo como parte de su arquitectura. En este servidor se crearon dos archivos principales llamados *actions.py* donde se declaran las Actions y el archivo *geoconsultas.py* en el cual están programadas las funciones personalizadas.

El archivo actions.py del Action Server de Rasa juega un papel fundamental en la implementación de las acciones del Asistente Virtual. En este archivo se declaran las acciones, las cuales tienen asignado un nombre específico y único para su identificación dentro del sistema. Estos nombres forman parte del entrenamiento del Asistente Virtual, lo que permite al CORE reconocer y seleccionar la acción correspondiente en función de la interacción con el usuario.

Gracias al entrenamiento previo y a los nombres asignados a las acciones, el CORE puede

tomar decisiones. Un fragmento de la definición de una Action se ve en la figura 3.31:

```python
class ActionGeoConsulta(Action):

    def name(self) -> Text:
        return "action_geoconsulta"

    def run(self, dispatcher: CollectingDispatcher,
            tracker: Tracker,
            domain: Dict[Text, Any]) -> List[Dict[Text, Any]]:

        eje_pertenece = geo.funDinamicQuerry(tracker.latest_message['intent'], tracker.latest_message.get('place'))
        dispatcher.utter_message(response="utter_cargando_visor", eje= eje_pertenece)

        return []
```

Fig 3.31: Ejemplo de definición de una Action

El formato de cada Action se define como una clase con dos funciones, además, una de las características de RASA es el componente *Tracker*, el cual es el encargado de recibir la solicitud que realiza el CORE. Esta solicitud está en formato JSON que incluye el nombre de la acción prevista, el contenido del dominio, la Intent y la Entity. Al igual que la comunicación entre NLU y CORE, no es necesario crear un servicio para este proceso.

Al realizar el entrenamiento se creó una Action por cada Intent, como se puede ver en el ejemplo para redes hidrográficas de la figura 3.32:

```python
class ActionRedesHidrograficas(Action):

    def name(self) -> Text:
        return "action_redes_hidrograficas"

    def run(self, dispatcher: CollectingDispatcher,
            tracker: Tracker,
            domain: Dict[Text, Any]) -> List[Dict[Text, Any]]:

        eje_pertenece = geo.funDinamicQuerry("redes_hidrograficas")
        dispatcher.utter_message(response="utter_cargando_visor", eje= eje_pertenece)
        dispatcher.utter_message(text="Click aqui para ver resultado: [redes_hidrográficas](https://ide.ucuenca.edu.ec/geoportal/viwer/100)")
        return []
```

Fig 3.32: Action de entrenamiento para una Intent

El archivo actions.py es esencial para establecer la lógica y el comportamiento del Asistente Virtual, ya que define las acciones que se llevarán a cabo en respuesta a las solicitudes y necesidades del usuario, sin embargo, dentro de este archivo no se pueden programar funciones personalizadas, por lo tanto, se definió el archivo geoconsultas.py.

El archivo geoconsultas.py contiene las funciones necesarias para realizar la comunicación con los geoservicios de la IDE y la base de datos espacial, usando Python como lenguaje de programación. Las funciones que se definieron en el archivo geoconsultas.py son:

- **Conectarse a la Base de datos Geoespacial:** Servicio creado para la búsqueda de capas de geoinformación.

- **Cargar capas al visor de mapas:** API creada para el visor de mapas a la que se envía la orden de crear el visor con las capas recuperadas visibles.
- **Verificar coordenadas de lugar:** Servicio creado para consultar que la capa de información este dentro las coordenadas de las parroquias del cantón Cuenca.
- **Verificar servicio:** Función para realizar una solitud *Request* al geoservicio de la institución para verificar que en línea.

Estas funciones personalizadas son llamadas desde las Actions, por ejemplo para realizar la búsqueda en la base de datos espacial, se invoca a la función para realizar la conexión de la base de datos espacial, esta recibe como parámetro la Intent para realizar la búsqueda como se puede ver en la figura 3.33:

```python
def funDinamicQuerry(intencion):
    intencion= intencion.replace("_", " ")
    url = "https://ide.ucuenca.edu.ec/api/basededatosespacial/capas"
    response = requests.get(url)

# Verificar si la solicitud fue exitosa
    if response.status_code == 200:
    # Obtener los registros de la respuesta JSON
        df = response.json()
    df.dropna()
    #print(df.head())
    df2=df[df.subeje.str.contains(pat=intencion,na=False, case=False)]
    df3=df2[['Organizacion','Titulo Capa', 'Enlace servicio', 'Nombre Capa', 'Identificador', 'Tipo', "Eje Pertenece"]]
    print("Coincidencias con subeje: "+str(len(df3)))
    #print(df3.head())

    #Para recrrer el data frame resultante
    for indice, fila in df3.iterrows():
        addinfDatos(fila['Identificador'], fila['Enlace servicio'], fila['Tipo'], fila['Nombre Capa'])

#Para comprobar que se crearon correctaente los json
    print("Lista a enviar subeje + 2="+str(len(datos)))
    with open(r'export_data.json', 'w') as fp:
        json.dump(datos, fp, indent=4)
```

Fig 3.33: Fragmento de código de la función para realizar la conexión y búsqueda en la base de datos espacial

De igual manera, tenemos la función de comunicación con la IDE para agregar las capas, resultantes y disponibles para ser cargadas en un formato JSON, como se puede ver en el extracto de código de la figura 3.34

```python
def createVisor(datos):
    endpoint= "http://ide.ucuenca.edu.ec/api/igo2/createTemporalViwer"
    # Convertir los datos en formato JSON
    json_datos= json.dumps(datos)
    # # Encabezados de la solicitud
    headers = {"Content-Type": "application/json"}
    # # Hacer la solicitud POST y obtener la respuesta
    response = requests.post(endpoint, data=json_datos, headers=headers, verify= False)
    # # Imprimir la respuesta
    print(response.text)


def addinfDatos(tittle, url, tipo, layer):
    a={
        "title": tittle,
        "baseLayer": False,
        "visible": False,
        "type": tipo,
        "url": url,
        "layers": layer,
        "version": "1.3.2",
        "queryable":True,
      }
    datos.append(a)
```

Fig 3.34: Fragmento de código de función para crear visor

El Asistente Virtual enviar la orden para crear el visor temporal, la que contiene un JSON con la información para crear el visor y la lista de capas a ser cargadas, como se ve en la figura 3.35:

```
[
    {
        "title": "Open Street Map - Humanitaire",
        "baseLayer": true,
        "visible": false,
        "type": "osm",
        "url": "https://a.tile.openstreetmap.fr/hot/{z}/{x}/{y}.png",
        "maxZoom": 19,
        "optionsFromCapabilities": true,
        "params": {}
    },
    {
        "title": "Open Street Map",
        "baseLayer": true,
        "visible": true,
        "type": "osm",
        "url": "https://tile.openstreetmap.org/{z}/{x}/{y}.png",
        "maxZoom": 19,
        "optionsFromCapabilities": true,
        "params": {}
    },
    {
        "title": "cuenca_fotografia_satelital_1",
        "baseLayer": false,
        "visible": true,
        "type": "wms",
        "url": "http://ide.cuenca.gob.ec/geoserver/ide/wms?",
        "layers": "ide:igm_cuenca_nv_f4b_d_25cm_rgb_20181023",
        "version": "1.3.2",
        "queryable": true
    },
```


Fig 3.35: Envío de capas en formato JSON para ser cargadas al visor de mapas

3.6. Despliegue del Asistente Virtual e Integración con el portal de Infraestructura de Datos Espaciales

Para el despliegue del Asistente Virtual en el servidor físico, se ha tenido en cuenta la coexistencia de otros servicios del proyecto AVPPGIS, como el Geoserver, Geonetwork, PostGIS para la base de datos espacial y nginx. La arquitectura de despliegue se muestra en la figura 3.36, este diagrama representa la implementación del servidor físico y la interconexión de los distintos servicios con el AV.

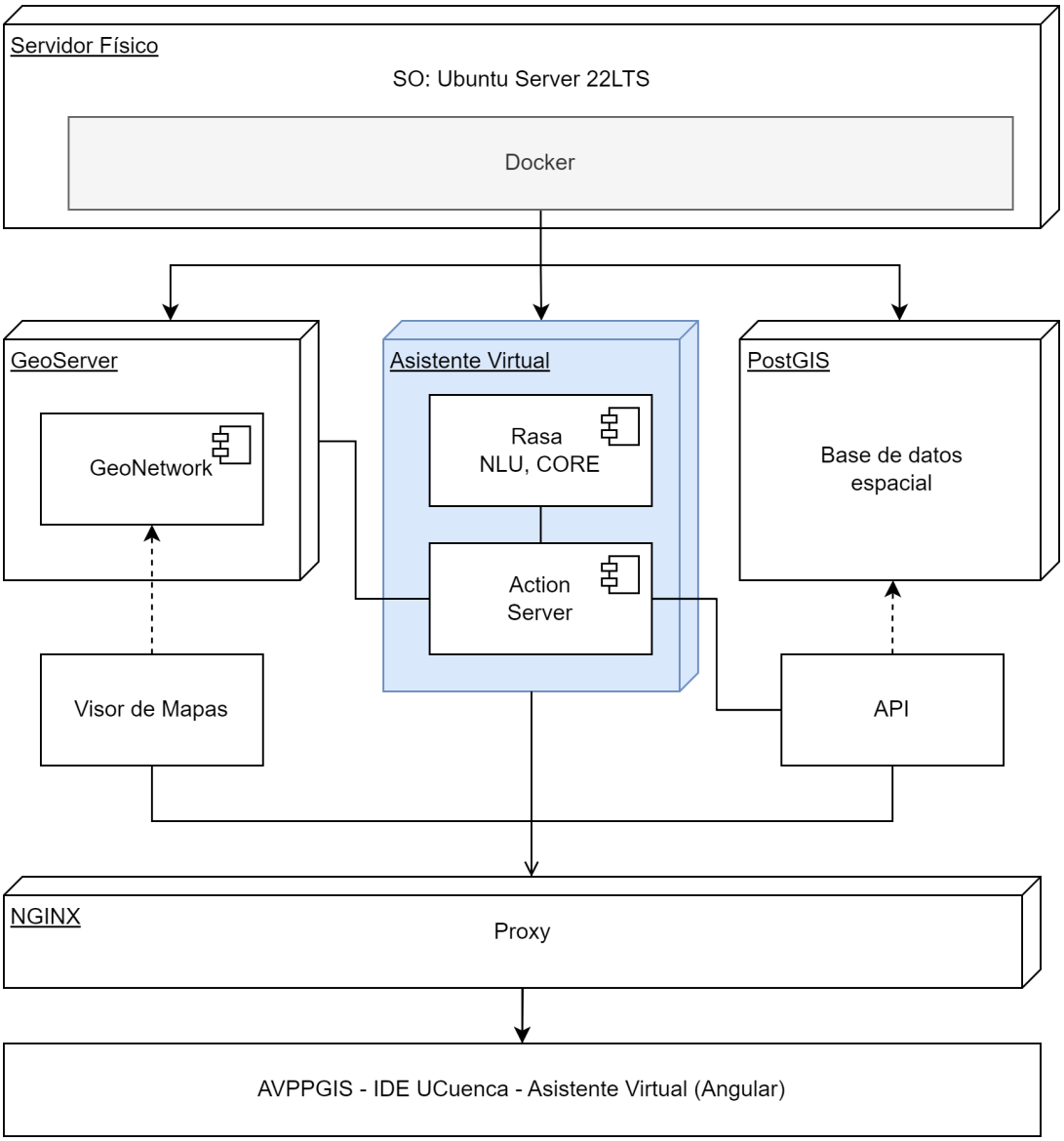


Fig 3.36: Diagrama de implementación en el Servidor físico

Para lograr un entorno independiente y resiliente para cada sistema, se propuso una arquitectura basada en contenedores Docker-compose. En esta arquitectura cada servicio se despl-

iega de manera individual en su propio contenedor Docker, lo que brinda un nivel adicional de aislamiento y flexibilidad.

En conjunto con el equipo de trabajo del proyecto principal, se resolvió desplegar la arquitectura sobre un sistema operativo Linux Ubuntu Server 22 LTS debido a su estabilidad y soporte a largo plazo, además, la propuesta de usar contenedores docker fue acepta por los demás miembros del proyecto.

Dentro del diagrama 3.36, también se puede ver el GeoServer que aloja el front-end y back-end del Visor de mapas. Por otro, se tiene la base de datos espacial que fue implementada con los datos de la matriz de geoinformación obtenida en la sección 3.1*Análisis, depuración y normalización de los datos*. El equipo encargado de la base de datos espacial, la diseño sobre tecnología PostGIS, las tablas a las cuales el AV tiene acceso siguen el siguiente esquema ver figura 3.37, el resto del esquema es transparente para el sistema del AV.

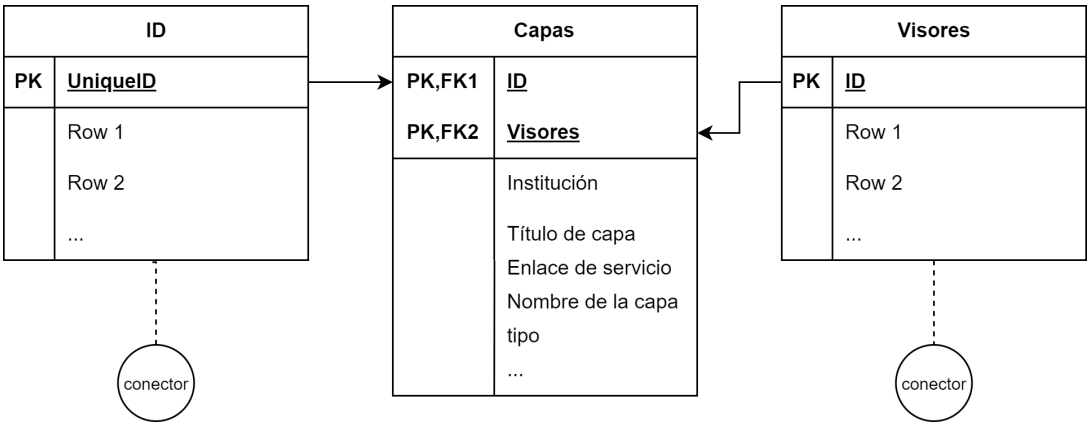


Fig 3.37: Fragmento de esquema de la base de datos espacial

3.6.1. Diseño del contenedor del Asistente Virtual

En esta sección se diseña el contenedor para alojar el Asistente Virtual, el cual se configuró con los recursos necesarios para asegurar su óptimo rendimiento:

- CPU: 4 núcleos
- RAM: 4 GB
- Disco: 50 GB
- Puertos: 5005, 5055

Como se explicó en la sección 3.4.2 los módulos Rasa NLU y Rasa CORE trabajan como uno solo, así mismo, el módulo Action Server es independiente. Por lo tanto, para lograr

un despliegue eficiente, se crearon dos contenedores docker-compose dentro del contenedor principal del Asistente Virtual. Esto permite crear la configuración necesaria para cada módulo y asegurar su correcto funcionamiento, es decir, cada uno puede instalar sus propias dependencias y paquetes de manera independiente sin afectarse entre ellos.

Este diseño mediante docker-compose facilita la gestión y control sobre cada componente del Asistente Virtual durante el despliegue. Además, se estableció una red privada entre los dos contenedores para facilitar la comunicación y el intercambio de datos entre los volúmenes e imágenes. Este diseño se puede ver en la figura 3.38:

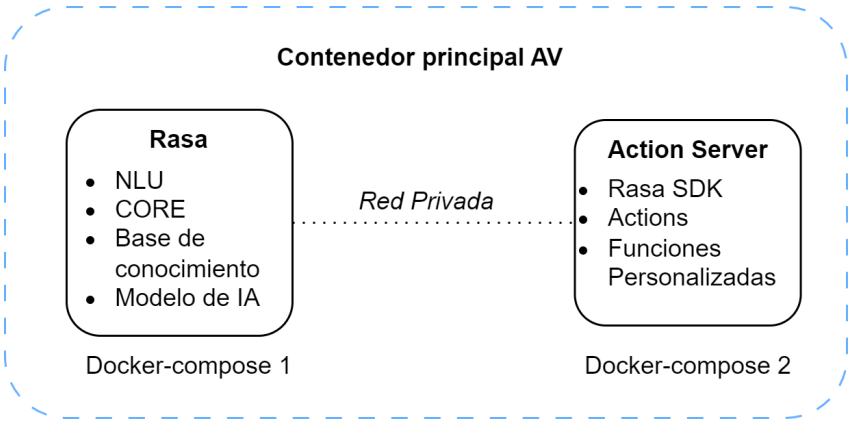


Fig 3.38: Diagrama de módulos docker-compose del AV

Para crear estos contenedores es necesario un Dockerfile para cada uno y un archivo docker-compose.yml para integrarlos, los cuales se detallan a continuación.

Contenedor para Módulos Rasa

Para la creación de este contenedor se define su configuración en el archivo “Dockerfile”, este se encarga de establecer las instrucciones para construir una imagen del contenedor, en este caso es la versión del framework Rasa en la versión 3.3.4 y se especifican los pasos necesarios para crear un entorno de ejecución coherente, como se puede ver en la figura 3.39.

```
Dockerfile > FROM
1   FROM rasa/rasa:3.4.0
2   RUN mkdir -p /app
3   COPY . /app
4   WORKDIR  '/app'
5   USER root
6
7   RUN  rasa train
8
9   VOLUME /app/models
10
11
12  CMD [ "run","-m","/app/models","--enable-api","--cors","
13
14  EXPOSE 5005
```

Fig 3.39: Construcción de Dockerfile para RASA

Dentro de este archivo se indica la versión del framework Rasa con que de se desarrolló el Asistente Virtual, se define un directorio de trabajo llamado “app”, se agrega el usuario responsable para la ejecución de comandos en la terminal, se define la ruta del volumen necesario para Docker y el comando a ejecutar para levantar el contenedor así como el puerto por el que este contenedor puede comunicarse con los demás contenedores.

Contenedor para Action Server

De la misma manera se crea el archivo Dockerfile para el contenedor del Action Server. En el caso de este Dockerfile, utilizamos la imagen de RASA-SDK. De la misma forma, en el archivo Dockerfile se incluye el listado de las dependencias y paquetes requeridos, especificados en la sección 3.1, en el archivo denominado *requisitos.txt*. Además, se incluye la configuración de los directorios de trabajo y la instalación de los requisitos para el despliegue en el ambiente de producción.

```
actions > Dockerfile > ...
1   FROM rasa/rasa-sdk:3.4.0
2   WORKDIR /app
3   USER root
4   COPY requirements.txt requirements.txt
5
6   RUN python -m pip install -U pip
7   #RUN pip3 install -r requirements.txt
8   RUN pip install -r requirements.txt
9   EXPOSE 5055
10  USER 1001
```

Fig 3.40: Dockerfile para Action Server

3.6.2. Configuración de archivo Docker-compose

El archivo docker-compose.yml se utiliza para configurar e integrar múltiples contenedores de un sistema. En este archivo se reúnen los contadores a usar, en este caso Rasa y Action server. De los cuales se debe especificar sus imágenes, un contexto o descripción, los directorios de trabajo, los volúmenes y los puertos de comunicación definidos en el Dockerfile de cada contenedor. Además, se define la configuración de la red privada entre contenedores como se ve en la figura 3.38, así como la red de comunicación en la que trabajan todos los contendores del sistema alojados en el servidor. Todas estas configuraciones se pueden ver en la figura 3.41

docker-compose.yml

```yaml
version: '3.0'
services:
    rasa:
      container_name: "rasa_server"
      user: root
      build:
        context:  .
      volumes:
      - "./:/app"
      ports:
        - "5005:5005"
      networks:
        - ide-network
    action_server:
      container_name: "action_server"
      build:
        context: actions
      volumes:
        - ./actions:/app/actions
        - ./data:/app/data
      ports:
        - 5055:5055
      networks:
        - ide-network
networks:
  ide-network:
    driver: bridge
```

Fig 3.41: Construcción de archivo docker-compose.yml

Como se puede observar en la figura anterior, los servicios que se despliegan dentro del contenedor principal son Rasa y Action Server, cada uno con sus configuraciones que deben coincidir con las de su Dockerfile. Además, se puede ver la configuración de la red privada para su comunicación. Con este único archivo es posible crear, iniciar y detener los contenedores y la red privada del asistente virtual, mediante el comando *docker-compose up –build*.

Los contenedores desplegados dentro del servidor se pueden ver en la figura 3.42

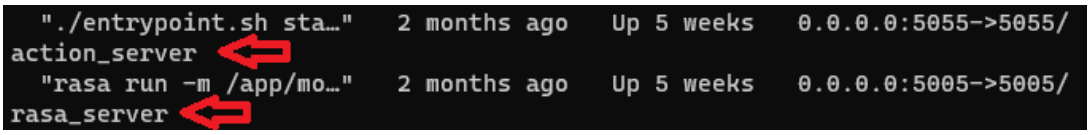


Fig 3.42: Captura de los contenedores desplegados en el servidor físico

Una vez desplegado estos contenedores, el servicio del Asistente virtual está disponible para ser consumido desde el front-end de la IDE UCuenca mediante la interfaz gráfica del Asistente Virtual, como se mencionó en la sección 3.4.4 Métodos de comunicación entre front-end y back-end, se definió el uso de Socket.IO, el cual maneja los eventos que ocurren en la interfaz gráfica del AV, mediante la URL del servicio y el puerto de comunicación que de específica en el archivo docker-compose.yml, como se puede ver en la figura 3.43 en el extracto de código de la interfaz gráfica del Asistente Virtual.

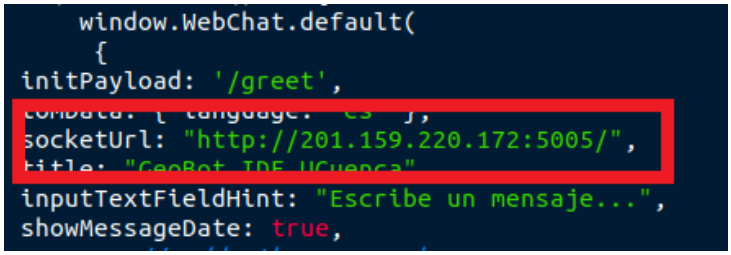


Fig 3.43: Extracto de código para comunicación desde el front-end del AV

3.7. Secuencia de Interacción y Casos de Uso

3.7.1. Diseño de Secuencia entre el AV y los componentes del Sistema

Es de vital importancia demostrar y visualizar la interacción entre el Asistente Virtual y la IDE UCuenca. La creación de un Diagrama de Secuencia permite representar de manera clara y precisa cómo se comunican y colaboran los componentes del sistema.

El Diagrama de Secuencia de la figura 3.44 muestra el flujo de intercambio de mensajes y acciones entre el usuario, el Asistente Virtual, la IDE UCuenca y los geoservicios, en diferentes escenarios y situaciones. Esto ayuda a comprender la dinámica de la interacción y a identificar los puntos de comunicación e integración.

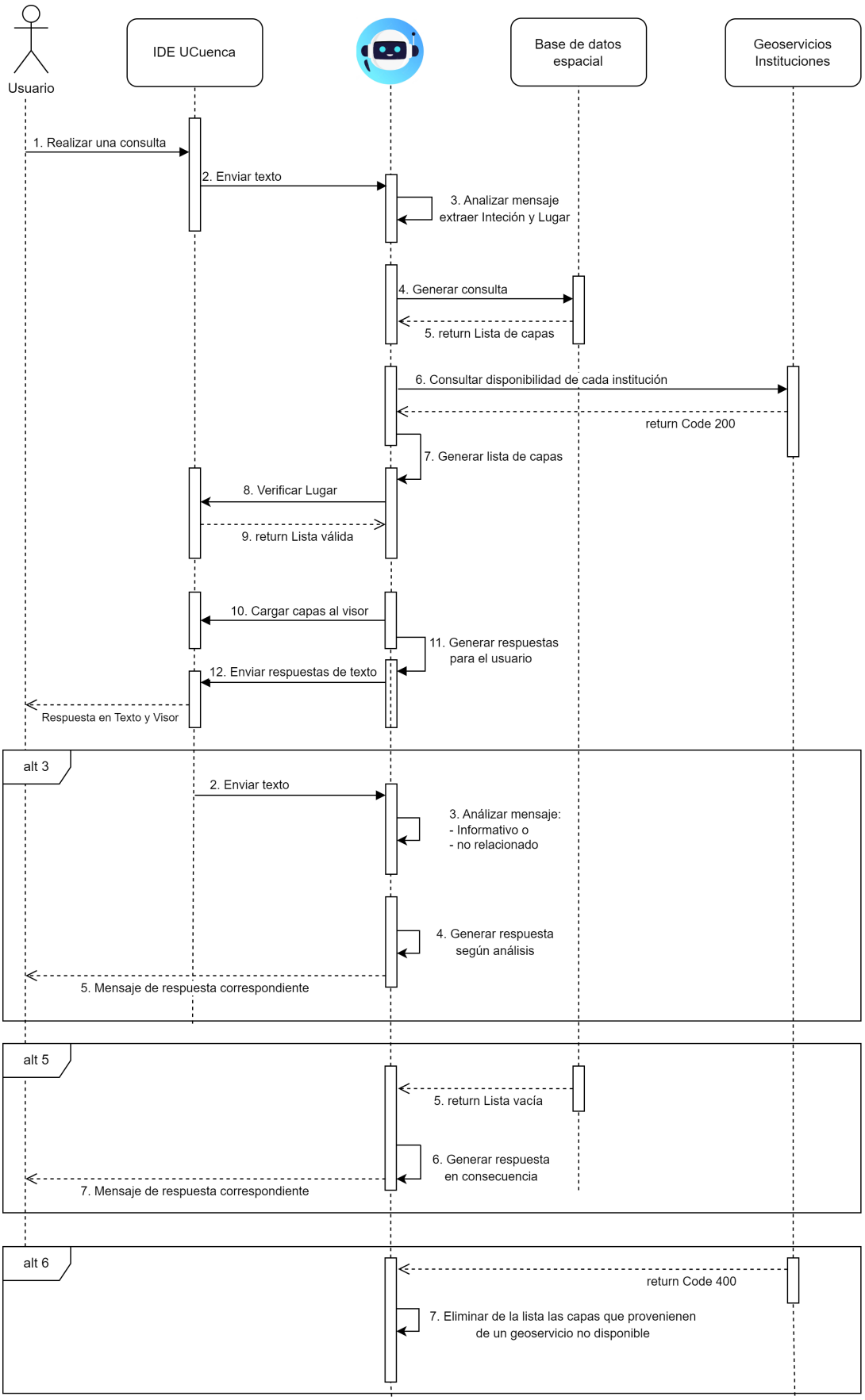


Fig 3.44: Diagrama de Secuencia

3.7.2. Diseño de escenarios de Casos de Uso

Se diseñaron tres Casos de Uso, según el diseño de secuencia presentado en la subsección anterior en la figura 3.44:

Caso de Uso 1: Realizar una geo-consulta

Este caso de uso es la función principal del Asistente Virtual y se da cuando el usuario realiza una geo-consulta al AV, ver figura 3.45:

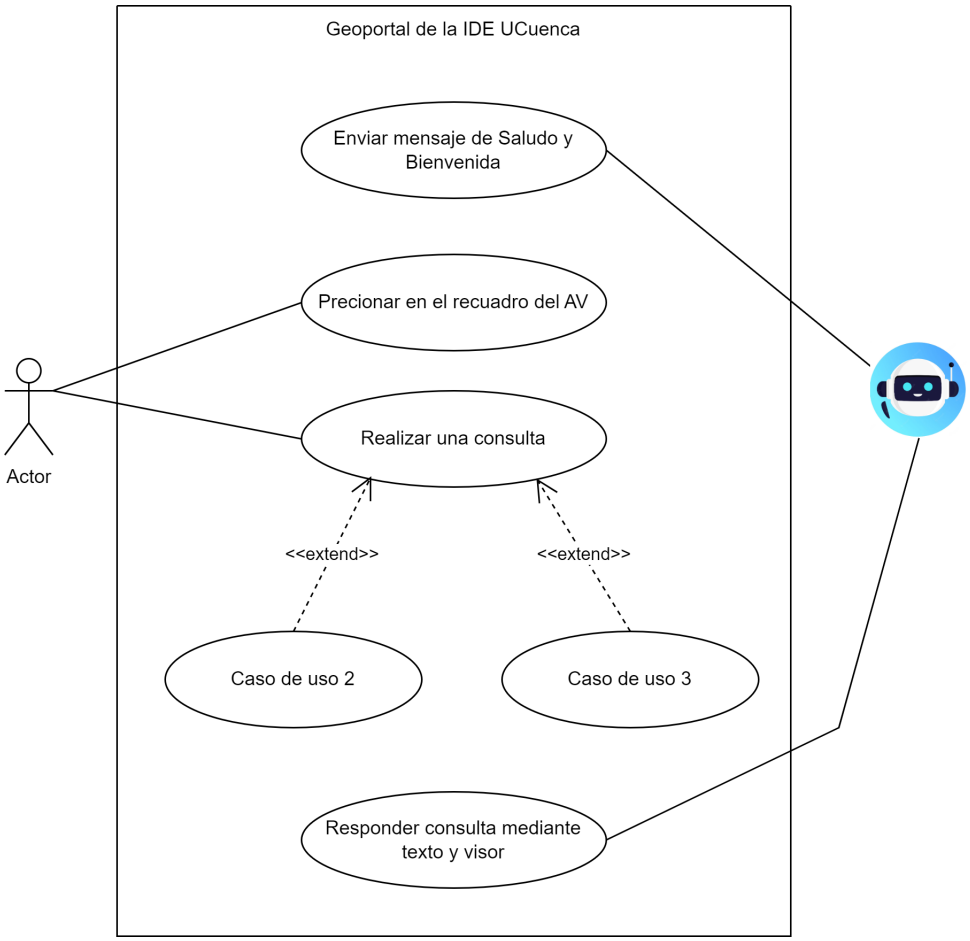


Fig 3.45: Caso de uno 1: Realizar una geo-consulta

Caso de Uso 2: Consulta no relacionada a ordenamiento territorial

Este caso se da cuando el usuario realiza una consulta no relacionada a ordenamiento territorial o solicita algo fuera de las capacidades para las que fue diseñado el Asistente Virtual, ver figura 3.46

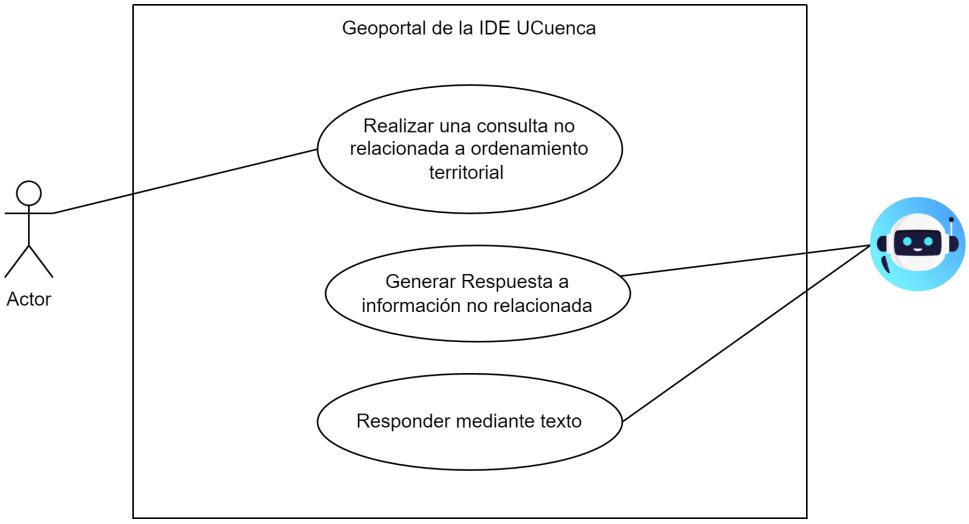


Fig 3.46: Caso de uno 2: Consulta no relacionada a ordenamiento territorial

Caso de Uso 3: Realizar consultas informativas

Este caso de uso se da cuando el usuario realiza preguntas informativas o envía mensajes que no necesitan de geoservicios y el Asistente Virtual puede solventarlas solo, respondiendo en consecuencia al mensaje del usuario.

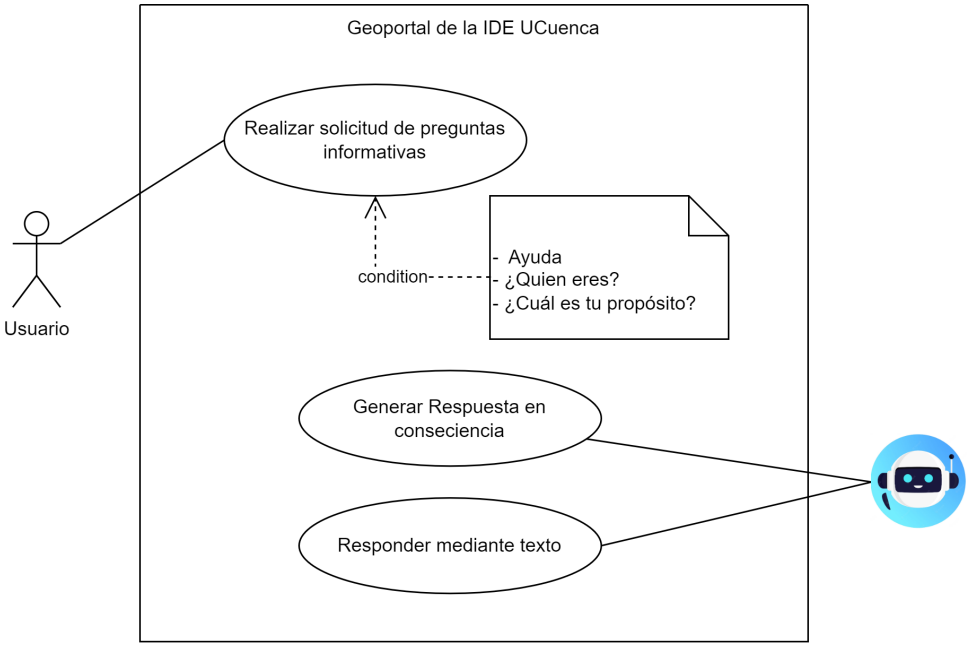


Fig 3.47: Caso de uso 3: Consulta que el AV puede resolver sin conectarse a los geoservicios.

3.8. Pruebas de usabilidad y Experiencia de usuario

Después de desarrollar y desplegar con éxito el Asistente Virtual basado en facilitar el acceso a geoinformación de ordenamiento territorial del cantón Cuenca, pasamos a evaluar los resultados, para determinar sus fortalezas y cómo lo percibirían los usuarios. Por lo tanto, evaluamos el sistema del AV desde dos perspectivas: la usabilidad técnica y la experiencia de usuario. Algunas de estas pruebas, tanto técnicas como de experiencia de usuario, se fundamentan en (Janssen et al., 2022), donde se sugiere un proceso para evaluar agentes conversacionales, también en (University of Goettingen et al., 2022) hace referencia a la evaluación de asistentes que se integran a un marco de trabajo nuevo.

3.8.1. Evaluación de Usabilidad técnica del Asistente Virtual

Las pruebas de usabilidad técnica para el AV en el ambiente de producción, desplegado en el geoportal de la IDE UCuenca se centran en evaluar la eficiencia y funcionalidad del asistente en términos de su rendimiento y capacidad de respuesta. Algunas de estas pruebas incluyeron varios aspectos clave: Prueba de rendimiento, prueba de tiempo de respuesta, prueba de compatibilidad y prueba de integración.
Estas pruebas fueron realizadas con 9 dispositivos: 5 computadoras(3 Windows, 2 Linux y 1 Mac) con 3 navegadores diferentes en cada una (Chrome, Firefox y Brave) y 4 celulares con sus navegadores por defecto, ver anexo E. A continuación, se detallan y se explican como fueron realizadas estas pruebas:

- **Prueba de rendimiento:** Se verifica cómo el AV responde cuando se le somete a una carga máxima de usuarios simultáneos. Se evalúa si el AV puede manejar múltiples solicitudes y conversaciones de manera efectiva sin ralentizarse o colapsar.

  Para realizar esta prueba, se prepararon los 9 dispositivos con los navegadores abiertos en el geoportal de la IDE UCuenca, desde diferentes dispositivos se realizaron varias interacciones con el asistente durante un tiempo aproximado de 7 minutos. Las interacciones fueron geoconsultas, consultas no relacionadas y preguntas informativas, poniendo a prueba los Casos de Uso de la sección anterior.

  Los resultados de estas interacciones simultáneas se pueden ver en la siguiente tabla 3.5

Navegador	Geoconsulta	Consulta no relacionada	Pregunta informativa
Chrome	5 interacciones	7 interacciones	7 interacciones
Firefox	7 interacciones	5 interacciones	7 interacciones
Brave	6 interacciones	5 interacciones	8 interacciones
iPhone (Safari)	3 interacciones	3 interacciones	3 interacciones
	18 interacciones Completas	20 interacciones Completas	25 interacciones Completas
		TOTAL:	63 interacciones

Table 3.5: Tabla de resultados de Prueba de Rendimiento

Se realizaron 63 interacciones con el Asistente virtual y no tuvo problemas en responder correctamente a cada una de las consultas que se le solicitaron.

- **Prueba de tiempo de respuesta:** Se mide el tiempo que tarda el AV en proporcionar respuestas a los usuarios. Se evalúa si el tiempo de respuesta es menor a 60 segundos o más rápido y cumple con las expectativas de los usuarios.

  Esta prueba puede depender de factores, como la velocidad de internet y el dispositivo donde se realiza la prueba. Esta realizó en una PC con Windows 10, con un procesador i7, 16gb de Ram y una velocidad de internet de 60 Mbps.

  Según (University of Goettingen et al., 2022), la mejor manera de capturar el tiempo de ejecución en un entorno de producción, es verificarlos con la herramienta de “Performance” de un navegador. Por lo tanto, se realizaron pruebas de tiempo para cada Caso de Uso, el inicio de captura de tiempo comienza cuando el usuario inicia la interacción con el AV, es decir, cuando despliega la ventana, escribe la geoconsulta y el AV responde con la información solicitada, como se ve en la figura 3.48 un ejemplo de captura de tiempo:

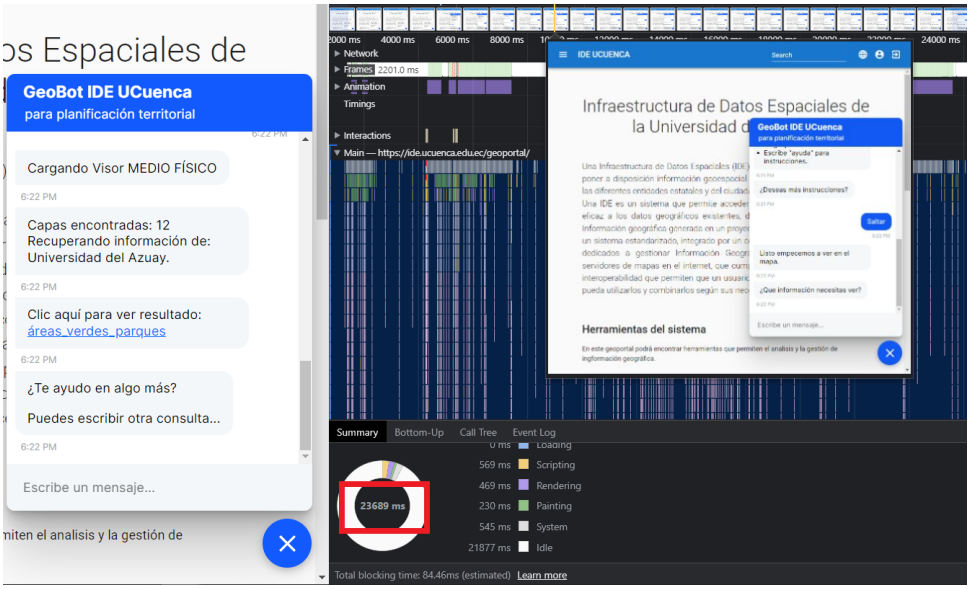


Fig 3.48: Captura de tiempo de una geoconsulta.

Como se puede ver en la figura anterior, el ejemplo realizado es de una geoconsulta, el tiempo recuperado es de 23,86 segundo, se realizaron más pruebas para lograr un tiempo promedio de interacción y respuesta con geoconsultas, consultas no relacionadas y preguntas informativas que son los Casos de Uso, como se puede ver en la tabla 3.6

Tipo	Consulta	Tiempo (s)
Geoconsulta		
	Parques	23.86
	Redes hidrográficas	22.12
	Imágenes satelitales	24.12
	Farmacias	19.99
	Adultos mayores	23.89
Consultas no relacionadas		
	Abre videos de Youtube	13.67
	Computadoras	11.63
	Locales de fiesta	12.33
	Perros y gatos	12.56
	Sillas y mesas	13.45
Pregunta informativa		
	Ayuda	9.24
	¿Quién eres?	10.03
	Instrucciones	9.45
	¿Con quién hablo?	10.94
	Gracias	7.33

Table 3.6: Tabla extracto de capturas de tiempos según casos de Uso.

De esta manera se capturó el tiempo de 75 interacciones con el Asistente Virtual para obtener un promedio de tiempo de respuesta según el caso de uso, como se ve en la tabla 3.7:

Consulta	Cantidad	Tiempo(s) promedio
Geoconsulta	25	21.79
Consulta no relacionada	25	12.31
Pregunta informativa	25	9.52

Table 3.7: Tabla de promedio de tiempo de ejecución según caso de uso.

Como se puede observar, se realizaron 25 consultas para cada caso de uso y el tiempo promedio para cada uno cumple con la condición de responder en menos de 60 segundos.

- **Prueba de compatibilidad:** Se verifica si el AV es compatible con diferentes navegadores web y dispositivos, como computadoras de escritorio, dispositivos móviles y tabletas. Se evalúa si el AV se visualiza y funciona correctamente en todos estos entornos.

  Las pruebas de compatibilidad se realizaron abriendo el geoportal e interactuando con el Asistente Virtual en varios navegadores instalados en diferentes sistemas operativos, los resultados se pueden ver en la tabla 3.8

Sistema Operativo	Navegador	Compatibilidad
Windows 10	Edge, Chrome, Firefox, Brave	Si
Linux: Ubuntu 22	Firefox, Chrome	Si
MacOS Monterrey	Safari, Chrome, Firefox, Brave	Si
iPhone (iOS 16)	Safari, Chrome	Si
Android 11	Chrome	Si

Table 3.8: Tabla de resultados pruebas de compatibilidad

  Como se puede ver en la tabla anterior, el Asistente Virtual trabaja en todos los dispositivos, también se puede ver en el anexo E, el AV desplegado en varios dispositivos y diferentes navegadores.

- **Prueba de integración:** Se prueba la capacidad del AV para integrarse con otros sistemas y servicios. En este caso la base de datos espacial y los geoservicios de la IDE UCuenca. Se evalúa si el AV puede acceder y recuperar información de manera adecuada y realizar las acciones esperadas como crear el visor y cargar las capas.

  Esta prueba se realizó aplicando pruebas unitarias a cada una de las 132 intenciones de búsqueda, es decir, se realizó una por una la geoconsulta correspondiente a cada intención de búsqueda, este proceso se realizó en la interfaz del Asistente Virtual en la IDE UCuenca. Una primera iteración arrojó que 21 de las intenciones no estaban trabajando correctamente, se detectaron errores de escritura, en los archivos de entrenamiento y en la base de datos espacial, esto se solucionó refinado los ejemplos y reentrenando el modelo. Esta prueba se iteró hasta verificar que para cada una de las intenciones el AV clasifique, recupere, cargue la información al visor de mapas y genere la respuesta para el usuario de manera correcta, ver anexo E.3.

3.8.2. Pruebas de Experiencia de Usuario y evaluación HCI

Como se detalla en la sección 3.4.4 *Diseño HCI del front-end del AV*, el diseño de la interfaz gráfica está centrado en el usuario. En el artículo (Janssen et al., 2022) hace referencia al marco PACT (personas, actividades, contexto y tecnología) para diseñar y evaluar interfaces gráficas de agentes conversacionales. Esta evaluación lo hace realizando encuestas a los expertos que usaran el asistente. Por lo tanto, se ha creado una encuesta para evaluar la satisfacción de la interfaz gráfica del Asistente virtual. La creación de esta encuesta está fundamentada en la metodología de evaluación presentada en (Kay, 2011), donde se recomienda el uso de no más de 10 interrogantes, las preguntas tienen que ser directas, evitando ambigüedades en las mismas. En cuanto a las opciones de respuesta, deben ser fáciles de entender para el usuario con valores de 1 a 5.

La encuesta creada consta de 8 preguntas referentes a la interfaz gráfica del AV, si le resultó útil o no y si recomendaría el uso del Asistente Virtual en un portal IDE. Esta encuesta fue aplicada a 15 personas, profesionales de ingeniería civil, arquitectura y ciudadanos comunes, ver anexo E.4. Los resultados se detallan en la sección 4.3 *Resultados de encuesta de Experiencia de Usuario*. Las preguntas se pueden ver a continuación:

Encuesta para evaluación de experiencia de usuario HCI del Asistente Virtual:

1. ¿El uso de GeoBot facilitó el uso de la IDE?
   (a) Sí
   (b) No
2. Las respuestas del GeoBot fueron:
   (a) Muy útil
   (b) Más bien útil
   (c) Normal
   (d) Más bien inútil
   (e) Totalmente inútil
3. El proceso de obtener lo que buscaba con el uso del GeoBot fue:
   (a) Rápido
   (b) Normal

(c) Lento

4. ¿Es la interfaz del GeoBot fácil de usar?

   (a) Muy sencilla

   (b) Más bien fácil

   (c) De dificultad media

   (d) Más bien difícil

   (e) Muy complicada

5. Las indicaciones de uso que proporciona el GeoBot son:

   (a) Muy útil

   (b) Más bien útil

   (c) Normal

   (d) Más bien inútil

   (e) Totalmente inútil

6. ¿Con qué frecuencia “se cuelga” o “no responde” el GeoBot?

   (a) Muy a menudo

   (b) Bastante frecuente

   (c) A veces

   (d) Casi nunca

   (e) Nunca

7. ¿Cómo está usted satisfecho con el rendimiento del GeoBot?

   (a) Muy satisfecho

   (b) Satisfecho

   (c) Normal

   (d) Insatisfecho

   (e) Terriblemente insatisfecho

8. ¿Recomendaría el uso del GeoBot y la IDE UCuenca en lugar de otros geoportales? (Imagen: IDE GAD municipal de Cuenca, ver anexo E.2)

(a) Definitivamente sí

(b) Probablemente sí

(c) No lo sé

(d) Probablemente no

(e) Seguramente no

4. Resultados y Discusión

Este proyecto de investigación sobre el Diseño e Implementación de un Asistente Virtual para facilitar el acceso a geoinformación a través de una Infraestructura de Datos Espaciales (IDE) con el caso de estudio de Ordenamiento Territorial del cantón Cuenca, ha alcanzado un 95% en el cumplimiento de los objetivos propuestos. Mientras que el 5% no cumplido se explica a detalle en la sección *4.4 Implementación de voz*, donde se indica que el método de entrada de voz quedó únicamente desarrollado, pero es incompatible para interactuar con el AV debido a la arquitectura actual de la IDE UCuenca.

A continuación se explican los resultados alcanzados:

4.1. Resultados del análisis, depuración, clasificación y normalización de los datos

La clasificación y depuración de los datos recopilados de las instituciones generadoras de geoinformación permitió entender el contexto en el que el Asistente Virtual debe trabajar. Los resultados de este análisis y clasificación de la geoinformación en los seis Ejes de Planificación del cantón Cuenca, partió de un aproximado de 2700 registros, los cuales se depuraron y categorizaron, obteniendo como resultado 1075 capas relevantes. En la tabla 4.1 se puede observar el número de capas según su eje de planificación:

Ejes de planificación territorial	No. Capas
Asentamientos Humanos y Canales de Relación	384
Medio Físico	296
Uso y Ocupación del Suelo	249
Culturales y Patrimoniales	75
Económico	38
Sociocultural	33
TOTAL	**1075**

Table 4.1: Resultado de la clasificación de las capas en Ejes de Planificación.

En el desarrollo de la arquitectura se identificó que además de los 6 ejes de planificación se necesitaba una sub clasificación para la geoinformación dentro de los ejes, por lo que se crearon los Subejes, logrando una clasificación y entrenamiento del Asistente Virtual con mayor precisión. En la tabla 4.1 se detallan los 132 Subejes clasificados dentro de los Ejes de Planificación:

Ejes de Planificación					
MEDIO FÍSICO	SOCIO CULTURAL	CULTURAL Y PATRIMONIAL	ECONÓMICO	ASENTAMIENTOS HUMANOS Y CANALES DE RELACIÓN	USO Y OCUPACIÓN DEL SUELO
• Geología • Suelos • Clima • Agua (Cuencas hidrográficas, red hidrográfica) • Áreas Protegidas • Calidad del Agua • Calidad de aire • Ruido • Riesgos Inundaciones • Riesgos Movimientos en masa • riesgos sísmicos, • riesgos volcánico • riesgos sequía • riesgos deslizamientos • Riesgos caída • Explotación minera • Cobertura vegetal • biosfera • hidrografía • incendios • Adjudicaciones de agua	• Demografía población • tasa de crecimiento, • natalidad, • mortalidad, • índice de envejecimiento, • esperanza de vida • población bonos de desarrollo • organizaciones sociales • categoria de ocupación • autoidentificación • poblacion por Sexo	• Áreas históricas • Patrimonio Cultural • Patrimonio Natural (INPC)	• Actividades Económicas • Turismo	• Equipamientos • E.Salud • E.Educación • E. de Bienestar Social • E.Cultura • E. Recreación y deporte • E. Seguridad • E. Aprovisionamiento • E. servicio • E. Recolección • Servicios básicos: agua, alcantarillado, Recolección de basura, energía electrica • Infraestructura vial/transporte público: estaciones, rutas, etc • Telecomunicaciones: telefonía, internet, energía electrica • Jerarquía de asentamientos • asentamientos humanos	• Permisos de construcción • Uso de suelo • Zona Industrial • Zona Agrícola • Zona Ganadera • Zona residencial • Zona de explotación de materiales áridos • Zona de explotación de petroleo • Zona de conservación • Margenes de protección de ríos • Ocupación de suelo (construcciones, lotización) • Farmacias • Conflictos de uso de suelo • industria pequeña • industria grande

Fig 4.1: Tabla de ejes y subejes

Además, en el proceso de Minería de Datos en la sección *3.1 Limpieza y Normalización*, se realiza la normalización del lenguaje en los datos recuperados de las 14 instituciones. Este proceso permitió añadir enriquecimiento semántico a los datos recuperados, llevando a la incorporación de expresiones entendibles y manejables para el usuario común. Estos términos geográficos normalizados se implementaron en la base de datos espacial, los cuales son recuperados desde la IDE UCuenca. También sirvieron como punto de inicio para la creación de la base de conocimiento del Asistente Virtual. En la figura 4.2 se puede observar estos datos desplegados en el portal de la IDE UCuenca.

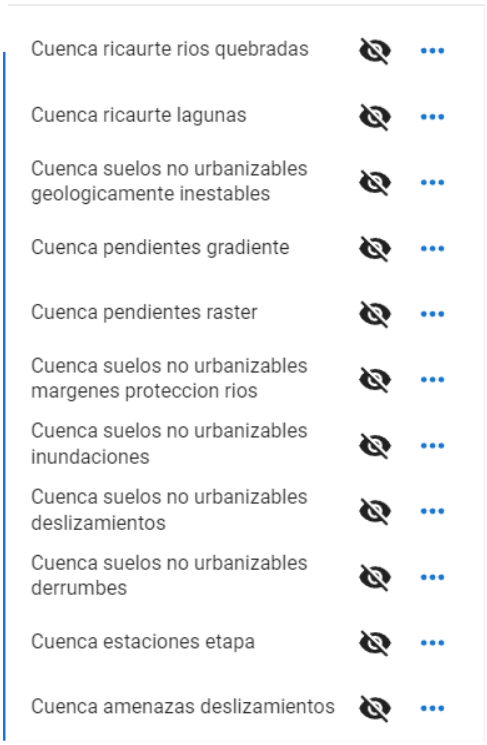


Fig 4.2: Extracto de listado de capas disponibles para el usuario en el visor Medio Físico de la IDE UCuenca.

Es importante recalcar que la metodología aplicada en la depuración y normalización de datos podrá ser utilizada de referencia para estudios posteriores en la estandarización de datos espaciales para IDEs.

4.2. Resultados del desarrollo del Asistente Virtual

En el desarrollo de Asistente Virtual se obtuvieron diferentes resultados: a) la formación de los datos para en entrenamiento para la creación de la base de conocimiento, b) el diseño del entrenamiento para la creación del Modelo de IA para la toma de decisiones, c) la creación del servidor de acciones Action Server, d) el diseño HCI e implementación con Interfaz gráfica del AV en el geoportal de la IDE UCuenca.

a) Base de Conocimiento

La creación de la base de conocimiento de geoinformación consta de 2 procesos:

- **Formación de los ejemplos de entrada y salida:** Se crearon manualmente 1056 ejemplos de entrada y se utilizaron los 132 Subejes de planificación como salida. Estos ejemplos incluyen lenguaje natural, lenguaje técnico geoespacial y jerga utilizada en el cantón Cuenca, definidos en la sección *3.5.2 Archivo nlu.yml*.

- **La elección de los ocho componentes para Pipelines:** Al no existir una base de conocimiento de ordenamiento territorial del cantón Cuenca, se experimentó con diferentes componentes de NLP, seleccionando ocho que se describen en la sección *3.4.2 Componentes y Modulación del AV*, que permiten realizar un entrenamiento supervisado, utilizando los ejemplos de entrada y salida a través de Rasa NLU.

La fusión de estos dos procesos dan como resultado la Base de Conocimiento, siendo la combinación de varios archivos de los componentes implementados, estos se pueden ver en la figura 4.3:

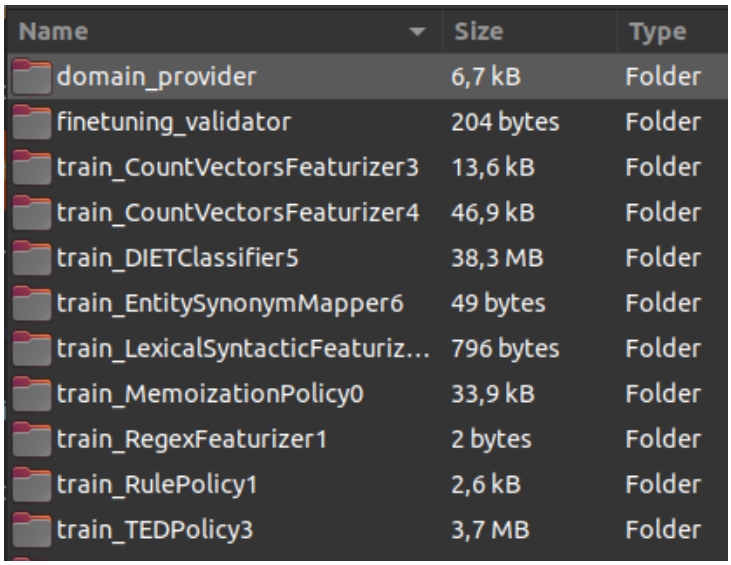


Fig 4.3: Archivos de la base de conocimiento

b) Modelo de IA para toma de decisiones

La creación del Modelo de IA se basa en el diseño de: los eventos de interacción descritos en el archivo stories.yml en la sección *3.5.3 Stories.yml*; las reglas construidas que el Asistente debe seguir, definidas en *3.5.4 Rules.yml*; y la implementación de las políticas de ejecución detalladas en la sección 3.4.2; y la Base de Conocimiento. Este Modelo de IA es una red neuronal de capas ocultas y se visualiza como un empaquetado. Gracias a la política de memorización y las interacciones de los usuarios el modelo realiza ajustes a lo largo del tiempo y genera nuevos empaquetados, como se puede ver en la figura 4.4 se tienen 11 archivos:

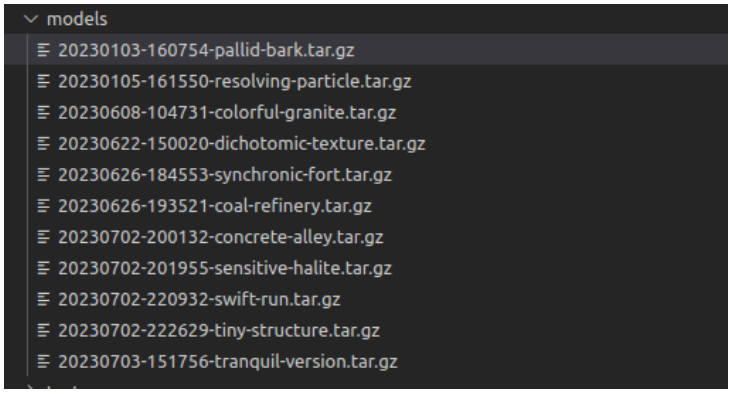


Fig 4.4: Modelo de IA para la toma de decisiones resultantes.

c) Creación de Action Server

Este es el servidor encargado de ejecutar las funciones personalidades del asistente como: recuperación de información de la base de datos espacial implementada en la IDE UCuenca, cargar las capas al visor de mapas, verificar la disponibilidad de los geoservicios de las instituciones y la formación de los datos de las respuestas que le llegarán al usuario cuando realice una consulta al Asistente Virtual. Este servidor independiente se puede ver desplegado en la figura 4.5:

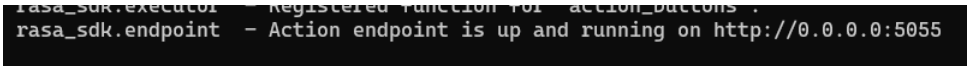


Fig 4.5: Despliegue de Action Server dentro su contenedor.

d) Despliegue e Implementación del Asistente Virtual en la IDE UCuenca

El despliegue e implementación exitosa del Asistente Virtual en el geoportal de la IDE UCuenca ha sido el resultado de un proceso integral que abarcó varios aspectos. En primer lugar, se ha desarrollado una Base de Conocimiento sólida y actualizada, respaldada por un Modelo de IA eficiente que permite al Asistente brindar respuestas rápidas y concisas. Además, se ha puesto un énfasis especial en el diseño de la interfaz gráfica, centrándose en la usabilidad y la experiencia del usuario, lo cual se detalla en la sección *3.4.4 Diseño de la interfaz gráfica y Usabilidad HCI*. La implementación del Asistente Virtual se llevó a cabo utilizando contenedores Docker, lo que garantiza una arquitectura flexible y escalable en el servidor físico, como se describe en la sección *3.6 Integración del AV a la IDE UCuenca*. En última instancia, el Asistente Virtual desplegado en el geoportal cumple con los requisitos funcionales y no funcionales establecidos en la sección *3.1*.

A continuación se presenta la implementación realizada:

- Cuando se abre el geoportal, el Asistente Virtual da la bienvenida, por lo que destaca su funcionamiento y presencia en el geoportal, como se ve en la figura: 4.6

Fig 4.6: Asistente Virtual destacando su funcionamiento en el geoportal.

- Congruencia en la interfaz gráfica de la IDE UCuenca, sin quitarle protagonismo al visor mapas, con la opción de minimizar el Asistente para una visualización completa del mapa, ver figura 4.7

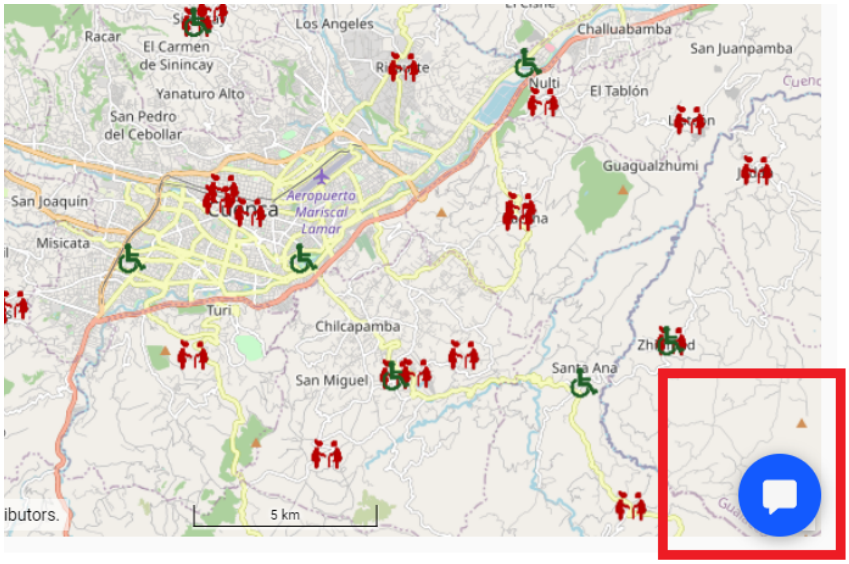


Fig 4.7: Asistente Virtual minimizado.

- El Asistente Virtual desplegado e implementado a la IDE UCuenca, ver figura 4.8

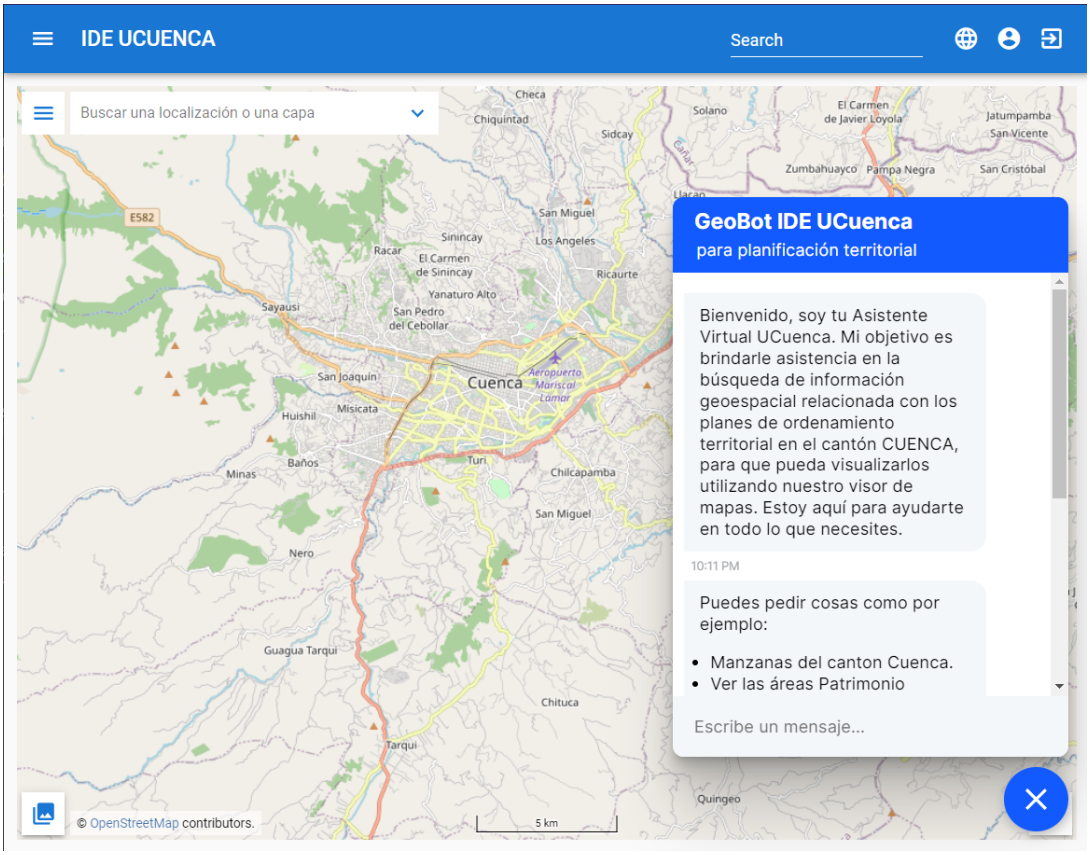


Fig 4.8: Asistente Virtual integrado a la IDE UCuenca.

El funcionamiento del Asistente Virtual se puede ver en la figura 4.9, realizando una interacción con el usuario, comunicándose con el back-end para realizar consultas de geoinformación, procesar mensaje, recuperar y validar servicio, cargar capas al visor y generar respuesta al usuario:

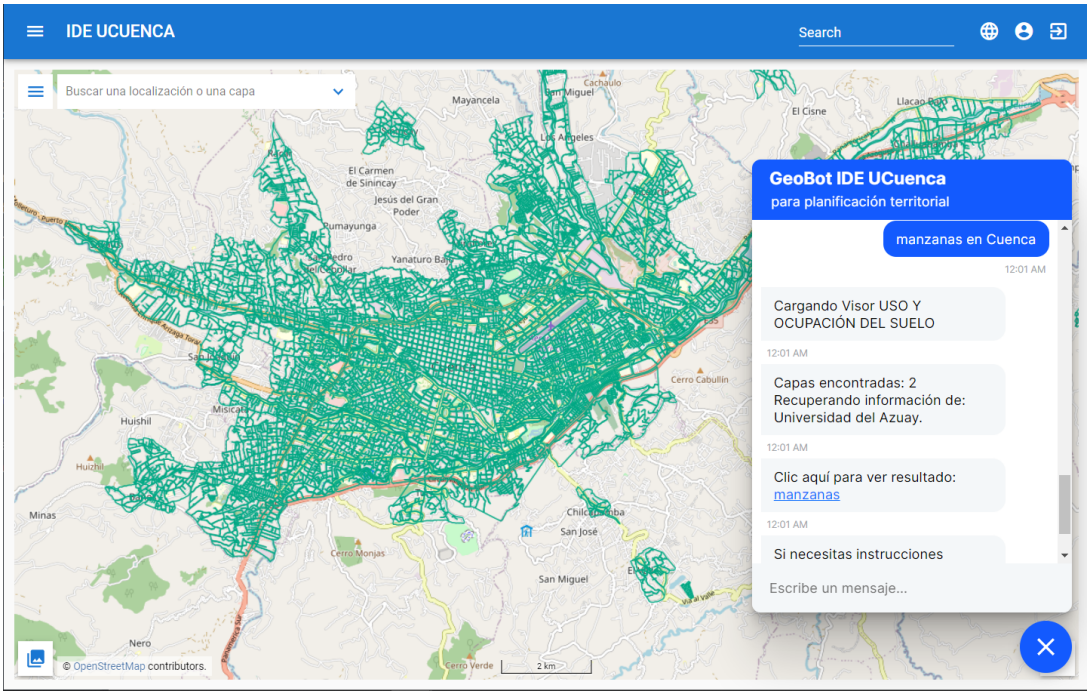


Fig 4.9: Asistente Virtual funcional en IDE UCuenca.

El Asistente Virtual cumple con las consideraciones de funcionalidad HCI:

- Disponibilidad en cualquier tipo de navegador conectado a internet.
- La ventana de Asistente virtual es responsiva, y se ejecuta correctamente en PC y en dispositivos móviles, como se ve en la figura 4.10.

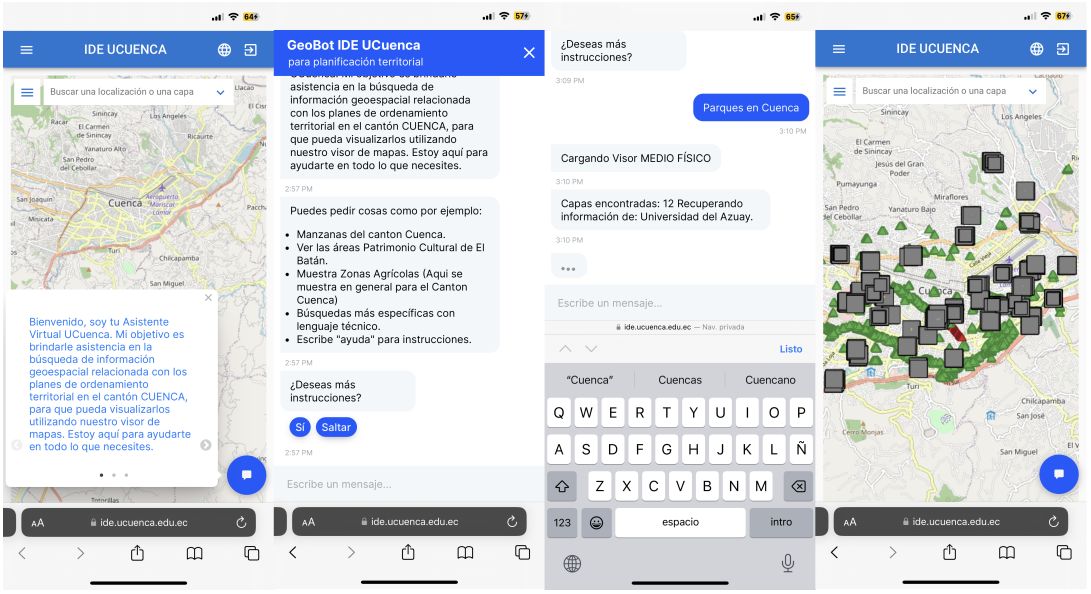


Fig 4.10: Asistente Virtual funcional en un dispositivo móvil.

Todo el código del desarrollo del Asistente Virtual, incluyendo la base de conocimiento, el modelo de IA, la interfaz gráfica y la implementación con contenedores Docker, se encuentra

alojado en el repositorio GitHub - Juan Bustamante. Este repositorio es de acceso público y contiene todo el código fuente del proyecto, lo que permite a otros investigadores y desarrolladores revisar, colaborar y utilizar el trabajo realizado para futuros proyectos o mejoras.

4.3. Resultados de la Encuesta de Experiencia de Usuario

Se aplicaron encuestas a 15 usuarios potenciales, entre ellos 8 ingenieros civiles, 5 arquitectos y 2 personas naturales. La encuesta consta de 8 preguntas, las cueles se detallaron en la sección *3.8 Pruebas de usabilidad y experiencia de usuario.* Las respuestas de cada pregunta se pueden ver a continuación:

Pregunta 1: ¿El uso del GeoBot facilitó el uso de la IDE?

Como se puede ver en la figura 4.11, a un 93.3% de las personas encuestadas les parece mucho más fácil el uso la IDE con ayuda del Asistente Virtual.

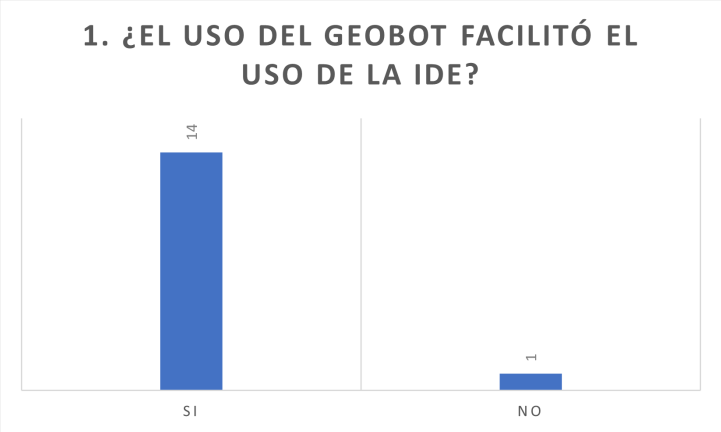


Fig 4.11: Resultado de pregunta 1

Pregunta 2: Las respuestas del GeoBot fueron:

Como se puede ver en la figura 4.12, el 53.33% de las personas encuestadas, piensan que las respuestas del AV, son útiles, y 40% piensa que es muy útil la información presentada por el AV.

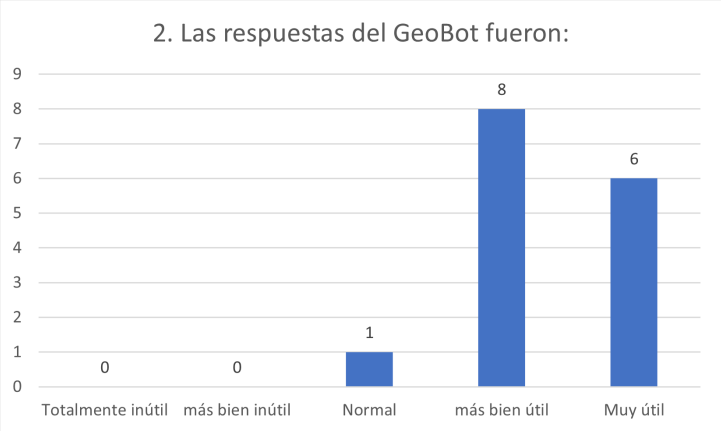


Fig 4.12: Resultado de pregunta 2

Pregunta 3: El proceso de obtener lo que buscaba con el uso del Geobot fue:

Como se puede ver en la figura 4.13, el 66% reportan *Normal* el tiempo de respuesta del AV, ya que se ve influenciado por el tiempo de carga de las capas, puesto que algunas son pesadas al momento de mostrarse en el visor.

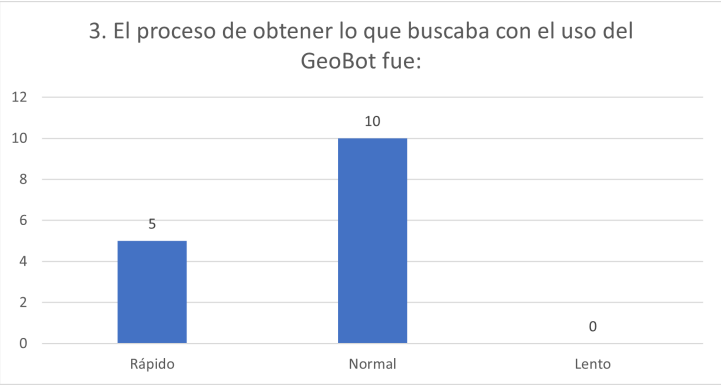


Fig 4.13: Resultado de pregunta 3

Pregunta 4: ¿La interfaz del GeoBot es fácil de usar?

Como se puede ver en la figura 4.14, el 93% de las personas encuestadas han respondido que es *Muy Sencilla* la interfaz gráfica del Asistente Virtual.

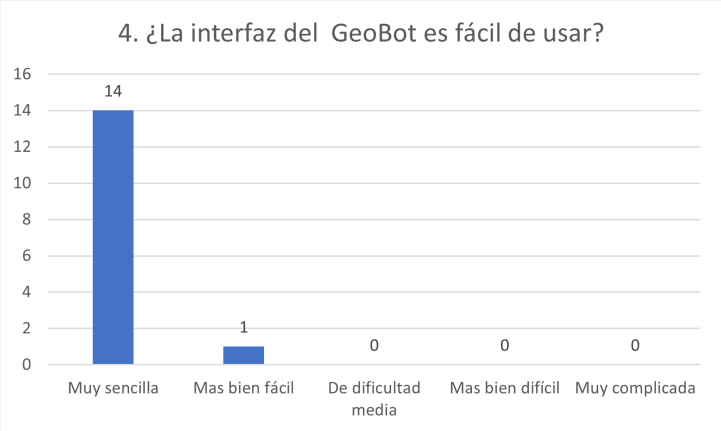


Fig 4.14: Resultado de pregunta 4

Pregunta 5: Las indicaciones de uso que proporciona el GeoBot son:

Como se puede ver en la figura 4.15, el 73% de las personas encuestadas han respondido que las indicaciones ofrecidas por el Asistente Virtual son útiles.

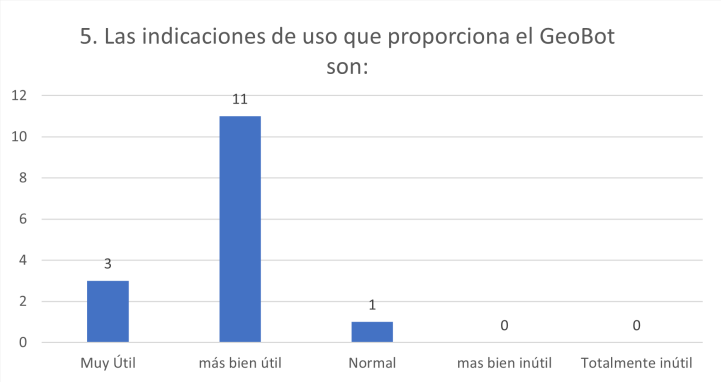


Fig 4.15: Resultado de pregunta 5

Pregunta 6: ¿Con qué frecuencia "se cuelga" o "no responde" el GeoBot:

Como se puede ver en la figura 4.16, el 100% de las personas encuestadas han respondido que el Asistente Virtual no ha presentado problemas técnicos.

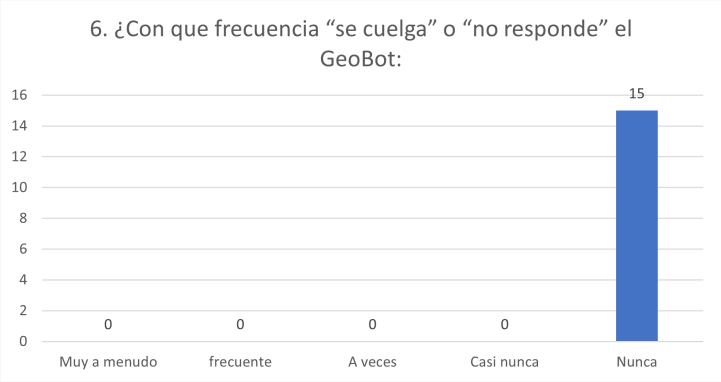


Fig 4.16: Resultado de pregunta 6

Pregunta 7: ¿Cómo está usted satisfecho con el rendimiento del chatbot?:

Como se puede ver en la figura 4.17, el 80% de las personas encuestadas se sientan satisfechos con el uso del Asistente Virtual, como retroalimentación solicitaron una respuesta un poco más personalizada.

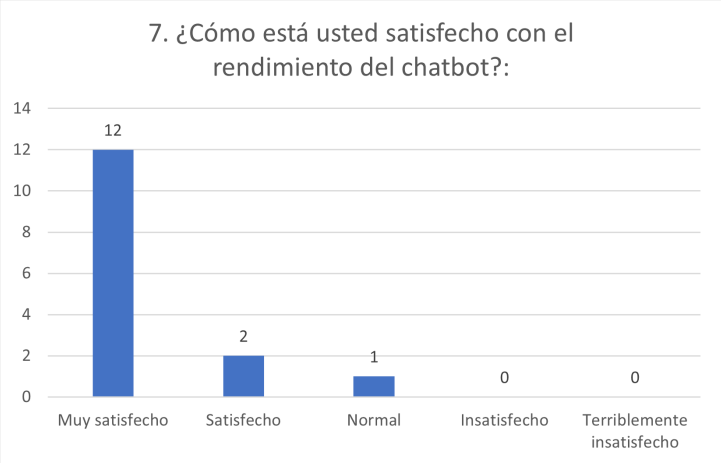


Fig 4.17: Resultado de pregunta 7

Pregunta 8: ¿Recomendaría el uso del GeoBot y la IDE UCuenca en lugar de otros geoportales?:

Como se puede ver en la figura 4.18, el 80% de las personas sí recomendarían el uso del GeoBot y la IDE UCuenca, pero se puede ver un considerable número que no. La retroalimentación otorgada es que ya utilizan un geoportal que les permite visualizar geoinformación de su profesión.

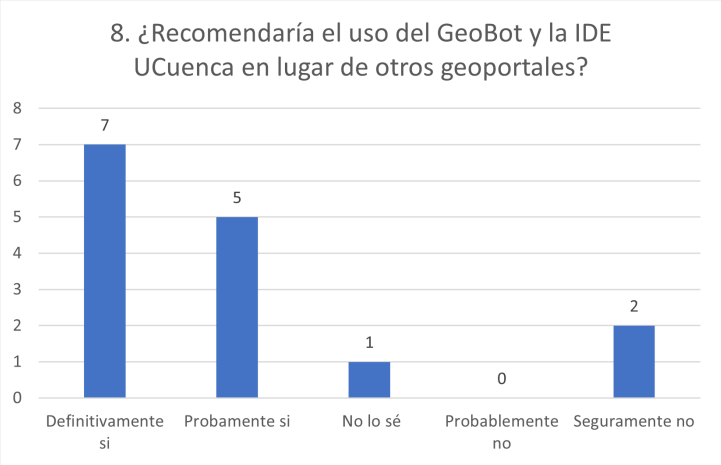


Fig 4.18: Resultado de pregunta 8

Tras realizar la encuesta a 15 usuarios potenciales, entre ingenieros civiles, arquitectos y personas naturales, se obtuvieron resultados alentadores. La encuesta, que constaba de 8 preguntas relacionadas con la interfaz gráfica del Asistente Virtual, su utilidad percibida y la recomendación del uso del AV en el portal IDE, arrojó una positiva impresión general. Los usuarios encontraron que el Asistente Virtual facilitó considerablemente el acceso a los datos geoespaciales a través de la IDE UCuenca. Además, destacaron la sencillez y utilidad de la interfaz gráfica, valoraron positivamente las respuestas proporcionadas por el AV y no reportaron problemas técnicos. Aunque algunos usuarios sugirieron respuestas más personalizadas, la retroalimentación en general fue favorable. Estos resultados confirman que el Asistente Virtual cumple su objetivo de facilitar la interacción y el acceso a la geoinformación en la IDE, mejorando la experiencia del usuario y brindando un valioso recurso para profesionales y público en general.

4.4. Implementación de Voz en el Asistente en la IDE UCuenca.

Durante la implementación del Asistente Virtual, se desarrolló la funcionalidad de la voz como método de entrada. Sin embargo, se encontraron desafíos significativos que limitaron su viabilidad dentro del proyecto. La forma de interactuar con voz al AV es mediante el envío de grabaciones de audio, es decir, mediante un front-end que permita capturar el audio y enviarlo al back-end para ser analizado por componentes que permitan el tratamiento de audio dentro del asistente. Estos componentes permiten detectar las palabras en el audio para transformarlas a texto y realizar el flujo ya descrito.

Durante el desarrollo de esta funcionalidad se evidenció que el tiempo de análisis de un solo audio demora 4 minutos aproximadamente, un tiempo considerablemente largo. Esto afectaba

negativamente la experiencia del usuario al hacer que las interacciones fueran tediosas y poco prácticas. Además, se descubrió que el análisis de comandos de voz dentro del framework Rasa, en el cual se basa el Asistente Virtual, solo está disponible en el idioma inglés. Esto imposibilitó la integración de una funcionalidad de voz efectiva y precisa en idioma español.

Otro inconveniente que se ha identificado es la limitación del lenguaje de programación del front-end que permite el manejo de grabaciones de audio con Rasa. Actualmente, esta funcionalidad se encuentra en versión beta dentro del framework Rasa y está en constante desarrollo, esta funcionalidad solo está disponible en el lenguaje de programación React, mientras que el desarrollo del front-end de la IDE UCuenca está construido en Angular.

En consecuencia, se tomó la decisión de no continuar con la implementación del método de entrada voz en el Asistente Virtual y centrándose solamente en texto como medio de interacción. A pesar de lo antes mencionado y para probar la posibilidad de uso del AV con interfaz de voz se desarrolló un front-end en React, este front utiliza componentes de reconocimiento de voz que permiten la comunicación con el framework Rasa, como se ve en la figura 4.19:

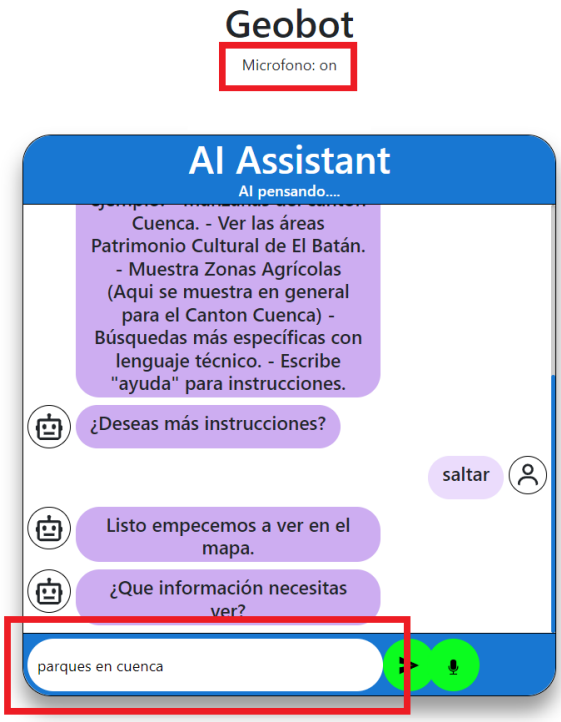


Fig 4.19: Interfaz gráfica en React que permite el método de entrada de voz.

4.5. Comparación con chat GPT

Una red o chat GPT es un sistema conversacional basado en el modelo de lenguaje GPT. Utiliza redes neuronales transformadoras para procesar y generar texto en respuesta a las

interacciones del usuario. El modelo se entrena previamente con grandes cantidades de texto sin supervisión y luego se utiliza para generar respuestas relevantes y coherentes en función del contexto y la información contextual proporcionada. A pesar de que esta red neuronal es muy potente para la comprensión y generación de texto, es importante tener en cuenta que las respuestas formadas pueden no ser siempre precisas o verídicas a causa del entrenamiento no supervisado. En el caso de esta tesis es necesario el entrenamiento supervisado, ya que dentro del contexto de geoinformación y del proyecto no se contaba con una gran cantidad de texto para realizar un entrenamiento no supervisado. Por lo que se resuelve implementar una red clasificatoria (DIET), más no una deductiva para comprender lo que usuario solicita. Además, se necesitan crear acciones personalizadas, a diferencia de la forma de trabajo de la red GPT que entrega una respuesta de dominio abierto, lo que puede ser contraproducente para el resultado que el usuario busca en este proyecto. Además, el fuerte de una red GPT es la generación de texto, al contrario de los objetivos perseguidos en esta tesis, que implica las acciones personalizadas que realiza el AV como la clasificación de geoinformación para la integración con la IDE.

5. Conclusiones

Gracias al aporte de esta tesis, el geoportal IDE UCuenca cuenta desde ahora con un Asistente Virtual que facilita el acceso a geoinformación de ordenamiento territorial del cantón Cuenca. Ofreciendo una herramienta en beneficio de los procesos de participación ciudadana como aporte a la elaboración de los planes de ordenamiento territorial.

El despliegue e implementación exitosa del Asistente Virtual en el geoportal de la IDE UCuenca ha sido el resultado de una cuidadosa planificación y ejecución en diferentes aspectos. En primer lugar, se ha llevado a cabo un arduo trabajo en la creación de la Base de Conocimiento del Asistente, que ha implicado recopilar y organizar una gran cantidad de información geoespacial desde las fuentes oficiales, y exclusivamente del cantón Cuenca. La información de al rededor de 1075 capas, procesada en la Base de Conocimiento ha conseguido garantizar la precisión y la calidad de las respuestas proporcionadas por el Asistente.

Además, el Modelo de IA utilizado en el Asistente ha sido entrenado con técnicas avanzadas de procesamiento de lenguaje natural y aprendizaje automático, lo que ha permitido al Asistente comprender y procesar de manera efectiva las consultas y solicitudes de los usuarios. El diseño de la interfaz gráfica del AV ha sido otro aspecto clave, con un enfoque en la usabilidad y la experiencia del usuario. Se ha buscado brindar respuestas rápidas, claras y concisas a través de una interfaz intuitiva y fácil de usar, lo que ha contribuido a una interacción fluida y satisfactoria con el Asistente.

La implementación del Asistente Virtual en el geoportal de la IDE UCuenca se ha realizado utilizando la tecnología de contenedores Docker, lo que ha brindado flexibilidad y escalabilidad al sistema. Los contenedores Docker han permitido gestionar de manera eficiente los diferentes componentes del Asistente, como el NLU, CORE y el Action Server, asegurando un despliegue y funcionamiento óptimos en el entorno del servidor físico. La descripción de toda esta instalación permitirá que otros investigadores puedan instalar plataforma similar para usos específicos, contribuyendo a la expansión de la IA en otros temas de investigación.

En cuanto a los requisitos funcionales y no funcionales, se ha realizado un exhaustivo análisis y seguimiento de los mismos, garantizando que el Asistente Virtual cumpla con las expectativas y necesidades establecidas en la sección correspondiente. Esto incluye aspectos como la ca-

pacidad de respuesta del Asistente, la compatibilidad con diferentes plataformas y dispositivos, entre otros.

En conclusión, el geoportal IDE UCuenca cuenta con un Asistente Virtual que facilita el acceso a geoinformación de ordenamiento territorial del cantón Cuenca, el cual está disponible para su uso en https://ide.ucuenca.edu.ec/geoportal/.

Adicionalmente, gracias a los procesos de investigación y desarrollo realizados en esta tesis, se han logrado publicar dos artículos científicos de relevancia. El primero, titulado "Virtual Assistants to bring geospatial information closer to a smart citizen" (Bustamante et al., 2022), aborda los beneficios y un posible prototipo de despliegue de los Asistentes Virtuales en la facilitación del acceso a la información geoespacial para los ciudadanos. El segundo artículo, denominado "Framework for Training a VA that Supports Territorial Planning" (Morocho et al., 2023), presenta una metodología detallada para el tratamiento de datos geoespaciales con el objetivo de crear una base de conocimiento y entrenar el modelo de inteligencia artificial del Asistente Virtual. Además, se está preparando un tercer artículo que abarcará más detalles específicos sobre el desarrollo del Asistente Virtual, el cual aún no ha sido publicado. Estos artículos contribuyen significativamente al campo de la inteligencia artificial aplicada a la geoinformación y promueven el uso de Asistentes Virtuales.

Conclusión del Objetivo especifico 1:

La información proveniente de las instituciones generadoras de geoinformación ha sido filtrada, organizada, clasificada y depurada, logrando un primer nivel de estandarización que garantizó la coherencia y consistencia de los datos, mejorando la precisión y la confiabilidad del entrenamiento del Asistente Virtual y la creación de la base de datos espacial.

Conclusión del Objetivo específico 2:

La elección de Rasa Open Source como framework de desarrollo para el asistente virtual demostró su capacidad para integrar conocimiento de dominio específico, lo que permitió un mayor control sobre el diseño y la lógica del asistente virtual. Resultando en una experiencia más personalizada y adaptada a las necesidades específicas del caso de estudio de ordenamiento territorial.

Se estandarizó la información de las 14 instituciones generadoras de geoinformación con la cual se creó una base de conocimiento específicamente para RASA, ya que no se contaba con una base de conocimiento de ordenamiento territorial del cantón Cuenca.

Se creó un Modelo de IA capaz de tomar decisiones coherentes de manera eficiente, brindando

respuestas rápidas y precisas al usuario.

Se logró una fácil gestión y escalabilidad de los servicios al utilizar contenedores Docker-compose en la implementación del Asistente Virtual, integrándose de forma adecuada a la IDE UCuenca y sus componentes, como el visor de mapas y la base de datos espacial. Esta arquitectura ha permitido que cada componente se despliegue de manera independiente. Esto se traduce en que si uno de los servicios experimenta alguna vulnerabilidad o caída, los demás servicios seguirán funcionando sin interrupciones, mejorando así la disponibilidad y la robustez del sistema en general.

Conclusión del Objetivo específico 3:

La integración del Asistente Virtual al visor de mapas de la IDE UCuenca permite a los usuarios comunes y a los profesionales técnicos en el campo del ordenamiento territorial, tener una experiencia de usuario más fluida y eficiente. Este asistente virtual es capaz de gestionar la información que recibe de diferentes entidades generadoras de geoinformación y realizar consultas para mostrar los resultados en el visor de mapas. Cabe recalcar que cada vez que un ciudadano utilice el Asistente Virtual, sigue añadiendo conocimiento al modelo de IA, este proceso se basa en la retroalimentación proporcionada por los usuarios y la experiencia acumulada a lo largo del tiempo. La red neuronal analiza y procesa esta información para adaptar y mejorar las respuestas ofrecidas, lo que contribuye a una interacción más fluida y precisa con los usuarios. La naturalidad con que se usa el asistente virtual es parecida a las existentes en los teléfonos inteligentes, por lo que los nuevos usuarios podrán acostumbrarse a dicho uso.

Conclusión del Objetivo específico 4:

La confiabilidad y eficiencia del Asistente Virtual es el resultado de la realización de diversas pruebas de usabilidad técnica. Estas pruebas, como la prueba de rendimiento, prueba de tiempo de respuesta, prueba de compatibilidad y prueba de integración, han permitido evaluar y validar el desempeño del Asistente Virtual en diferentes escenarios y condiciones. Además, se ha llevado a cabo la evaluación de la experiencia de usuario a través de la aplicación de encuestas a un público en general y a profesionales especializados en el campo de la geoinformación. Estas evaluaciones han brindado información valiosa sobre la usabilidad, funcionalidad y satisfacción del usuario con el Asistente Virtual, lo que garantiza una experiencia positiva para los usuarios finales.

6. Recomendaciones y Trabajo a futuro

- Organizaciones como el CONAGE podría proveer la estandarización de las etiquetas utilizadas en esta tesis para la homologación sintáctica de geoinformación al momento de incrementar la base de conocimiento que en este caso fue exclusiva para Ordenamiento Territorial.
- Utilizar la base de conocimiento creada en esta tesis para generar ontologías sobre ordenamiento territorial con fin de ampliar y generalizar el conocimiento para aplicaciones de inteligencia artificial.
- Combinar más componentes, como las habilidades de las redes neuronales GPT para ampliar la capacidad de respuesta con información de dominio abierto.
- Dar seguimiento a la retroalimentación que las comunidades y los usuarios profesionales otorguen a lo largo del tiempo para ofrecer la implementación de otros tipos de asistentes virtuales de acuerdo a sus necesidades y requerimientos.

Referencias

Adamopoulou, E., & Moussiades, L. (2020). An Overview of Chatbot Technology. In I. Maglogiannis, L. Iliadis, & E. Pimenidis (Eds.), *Artificial Intelligence Applications and Innovations* (pp. 373–383). Springer International Publishing. https://doi.org/10.1007/978-3-030-49186-4_31

ArcGIS, R. C. (2023). *¿qué es ArcGIS?* | *ArcGIS resource center*. Retrieved March 14, 2023, from https://resources.arcgis.com/es/help/getting-started/articles/026n00000014000000.htm

Arrieta, A. B., Rodríguez, N. D., Ser, J. D., Bennetot, A., Tabik, S., Barbado, A., García, S., Gil-Lopez, S., Molina, D., Benjamins, R., Chatila, R., & Herrera, F. (2019). Explainable artificial intelligence (xai): Concepts, taxonomies, opportunities and challenges toward responsible ai. *ArXiv*, *abs/1910.10045*.

Barricelli, B. R., & Fogli, D. (2021). Virtual assistants for personalizing IoT ecosystems: Challenges and opportunities, 1–5. https://doi.org/10.1145/3464385.3464699

Bunk, T., Varshneya, D., Vlasov, V., & Nichol, A. (2020, May 11). DIET: Lightweight language understanding for dialogue systems. https://doi.org/10.48550/arXiv.2004.09936

Bustamante, J., Morocho, V., Achig, R., & Mendieta, F. (2022). Virtual Assistants to bring geospatial information closer to a smart citizen. *2022 IEEE Sixth Ecuador Technical Chapters Meeting (ETCM)*, 01–06. https://doi.org/10.1109/ETCM56276.2022.9935761

Cartagena, M. i., Saur, J., Valenzuela, M., & Díaz-Moreno. (2010). La geoinformación: una necesidad creciente [Cartografía UNPSJB]. Retrieved November 9, 2022, from http://cartografiaunpsjb.jimdofree.com/art%C3%ADculos-para-compartir-y-reflexionar/la-geoinformaci%C3%B3n-una-necesidad-creciente/

ConsejoProvinciadelAzuay. (2018). PLAN DE DESARROLLO y ORDENAMIENTO TERRITORIAL DEL AZUAY ACTUALIZADO 2015 - 2030. *29 de mayor de 2018*.

Cruz, L. V., Mejía, R., & Morocho, V. (2018). POLÍTICAS PARA LA GESTIÒN DE LA INFORMACIÒN EN LA PLANIFICACIÒN TERRITORIAL [Number: 2]. *Revista Geoespacial*, *15*(2), 67–79. https://doi.org/10.24133/geoespacial.v15i2.1354

Cuesta, Ó., & Mélendez, S. (2017). Comunicación urbana: antecedentes y configuración de líneas de investigación en América Latina y España [Number: 37]. *Territorios*, (37), 205–228. https://doi.org/10.12804/revistas.urosario.edu.co/territorios/a.4889

Cutinha, D. D., Chiplunkar, N. N., Maved, S., & Bhat, A. (2021). Artificial intelligence-based chat-bot framework with authentication, authorization, and payment features (N. N. Chiplunkar & T. Fukao, Eds.), 179–187. https://doi.org/10.1007/978-981-15-3514-7_16

DockerInc, D. (2023a). Docker compose | docker documentation [(Accessed on 05/23/2023)].

DockerInc., D. (2023b). Docker documentation [(Accessed on 05/23/2023)].

Frei, F. A. J. (2018). Integrating a chatbot with a GIS-MCDM system, 96 pag. https://doi.org/http://unigis.sbg.ac.at/files_en/Mastertheses/Full/104383.pdf

García-Reina, L. F. (2018). Asistente virtual de tipo ChatBot [Accepted: 2018-09-18T20:50:20Z Publisher: Facultad de Ingeniería]. Retrieved August 11, 2022, from https://repository.ucatolica.edu.co/handle/10983/17726

Granell, C., Pesántez□Cabrera, P. G., Vilches□Blázquez, L. M., Achig, R., Luaces, M. R., Cortiñas□Álvarez, A., Chayle, C., & Morocho, V. (2021). A scoping review on the use, processing and fusion of geographic data in virtual assistants. *Transactions in GIS*, *25*(4), 1784–1808. https://doi.org/10.1111/tgis.12720

Hendriks, P. H., Dessers, E., & van Hootegem, G. (2012). Reconsidering the definition of a spatial data infrastructure [Publisher: Taylor & Francis _eprint: https://doi.org/10.1080/13658816.2011.6393 *International Journal of Geographical Information Science*, *26*(8), 1479–1494. https://doi.org/10.1080/13658816.2011.639301

Houston, J. (2017). The rise of chatbots! - identifying winners in the next wave of human-technology interaction. *Navidar*. Retrieved July 12, 2023, from https://www.navidar.com/insight/the-rise-of-chatbots-identifying-winners-in-the-next-wave-of-human-technology-interaction/

Izdebski, W., Zwirowicz-Rutkowska, A., & da Costa, J. N. (2021). Open data in spatial data infrastructure: The practices and experiences of poland. *International Journal of Digital Earth*, *14*, 1547–1560.

Janssen, A., Rodríguez Cardona, D., Passlick, J., & Breitner, M. H. (2022). How to make chatbots productive – a user-oriented implementation framework. *International Journal of Human-Computer Studies*, *168*, 102921. https://doi.org/10.1016/j.ijhcs.2022.102921

Kay, R. (2011). Evaluating learning, design, and engagement in web-based learning tools (WBLTs): The WBLT evaluation scale. *Computers in Human Behavior*, *27*(5), 1849–1856. https://doi.org/10.1016/j.chb.2011.04.007

Krishnan, J., Coronado, P., & Reed, T. (2019). SEVA: A systems engineer's virtual assistant.

Lalwani, T., Bhalotia, S., Pal, A., Bisen, S., & Rathod, V. (2018). Implementation of a chatbot system using AI and NLP. *6*(3), 5.

Lam, K. N., Le, N. N., & Kalita, J. (2020, December 18). *Building a chatbot on a closed domain using RASA*. https://doi.org/10.1145/3443279.3443308

Massai, L., Nesi, P., & Pantaleo, G. (2019). PAVAL: A location-aware virtual personal assistant for retrieving geolocated points of interest and location-based services. *Engineering Applications of Artificial Intelligence*, *77*, 70–85. https://doi.org/10.1016/j.engappai.2018.09.013

Mathew, A. (2023). Is artificial intelligence a world changer? a case study of OpenAI's chat GPT. *Recent Progress in Science and Technology Vol. 5*, 35–42. https://doi.org/10.9734/bpi/rpst/v5/18240D

Merrill, W. C., & Sabharwal, A. (2022). Transformers can be expressed in first-order logic with majority.

Mishra, D. S., Agarwal, A., Swathi, B. P., & Akshay, K. C. (2022). Natural language query formalization to SPARQL for querying knowledge bases using rasa. *Progress in Artificial Intelligence*, *11*(3), 193–206. https://doi.org/10.1007/s13748-021-00271-1

Morocho, V., Achig, R., Vivanco, L., Pacurucu, N., & Bustamante, J. (2023). Framework for training a VA that supports territorial planning [ISSN: 2573-1998]. *2023 Ninth International Conference on eDemocracy & eGovernment (ICEDEG)*, 1–5. https://doi.org/10.1109/ICEDEG58167.2023.10122094

Pavel, I. (2021). COMPARING CHATBOT FRAMEWORKS: A STUDY OF RASA AND BOTKIT.

RasaOpenSource. (2023). Introduction to rasa open source & rasa pro [(Accessed on 05/10/2023)].

Ridley, M. (2022). Explainable artificial intelligence (xai). *Information Technology and Libraries*.

Santander, F., & Morocho, V. (2018). LEVANTAMIENTO, ANÁLISIS Y PUBLICACIÓN DE GEOINFORMACIÓN EN UN VISUALIZADOR 3D. [Number: 2]. *Revista Geoespacial*, *15*(2), 24–37. https://doi.org/10.24133/geoespacial.v15i2.1350

SENPLADES. (2021). Plan de desarrollo y ordenamiento territorial del cantón cuenca [(Accessed on 05/19/2023)]. https://www.cuenca.gob.ec/sites/default/files/transparencia2017/PDOT

Studer, S., Bui, T. B., Drescher, C., Hanuschkin, A., Winkler, L., Peters, S., & Mueller, K.-R. (2021, February 24). Towards CRISP-ML(q): A machine learning process model with quality assurance methodology. https://doi.org/10.48550/arXiv.2003.05155

University of Goettingen, Von Wolff, R. M., Hobert, S., University of Goettingen, Schumann, M., & University of Goettingen. (2022). Designing process-based chatbots in enterprises: The case of business travel organization considering the users' perspective and business value. *AIS Transactions on Human-Computer Interaction*, *14*(4), 578–623. https://doi.org/10.17705/1thci.00180

Vivanco Cruz, L., Mejía, R., & Morocho, V. (2019). Políticas para la gestiòn de la informaciòn en la planificaciòn territorial. *Revista Geoespacial*, *15*(2), 67–79. https://doi.org/10.24133/geoespacial.v15i2.1354

Vivanco Cruz, L., Pacurucu Cáceres, N., Morocho, V., Lucero, A., Astudillo, J., & Calderòn, A. (2020). Democratizar el acceso y uso de la información como mecanismo de transparencia para los procesos deplanificación territorial [Section: Derechos a la Ciudad y Territorio: Memorias del XI Simposio Nacional de Desarrollo Urbano y Planificación Territorial], 102–111. Retrieved May 24, 2023, from https://dialnet.unirioja.es/servlet/articulo?codigo=8427106

Williams, J. D., & Zweig, G. (2016). End-to-end LSTM-based dialog control optimized with supervised and reinforcement learning. (arXiv:1606.01269). Retrieved May 22, 2023, from http://arxiv.org/abs/1606.01269

Anexos

A. Requisitos del sistema para instalación de Rasa Open Source:

- Sistema operativo: Rasa es compatible con Windows, macOS y Linux. En esta tesis se utilizó sobre Ubuntu Server 22 LTS.
- Memoria RAM: Se recomienda tener al menos 4 GB de RAM para un rendimiento óptimo.
- Espacio en disco: Se requiere un mínimo de 25 Gigabytes de espacio en disco.
- Procesador: Rasa puede funcionar en una amplia gama de procesadores, pero es necesario que sean arquitecturas del 2014 en adelante.

B. Matrices de Depuración y Normalización de geoinformación

B.1. matriz de data recuperada.xlsx

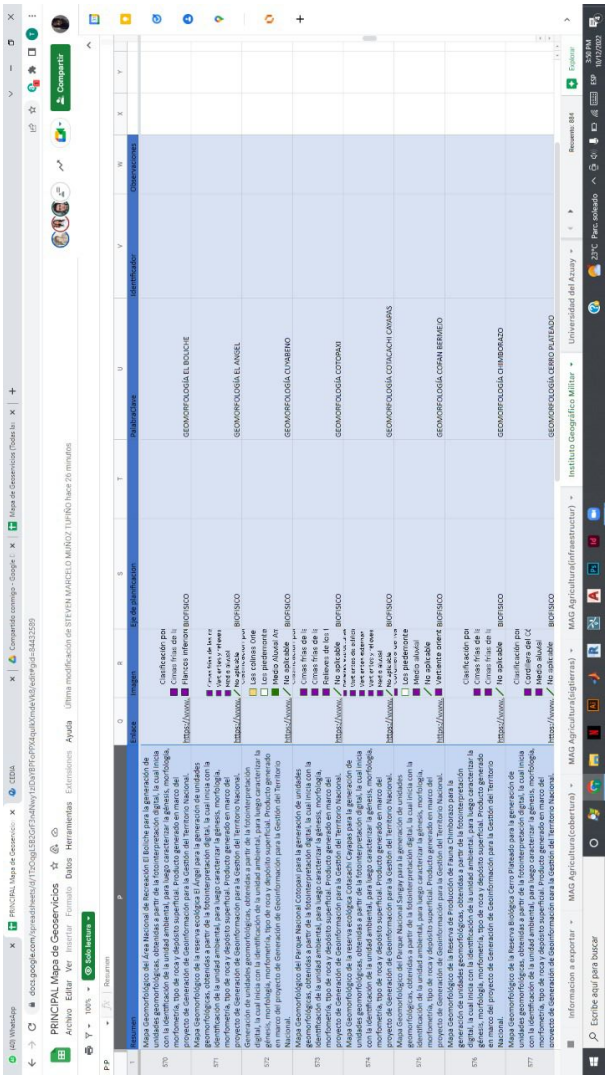


Fig B.1: Matriz de información resultante del script recovery layer info

Esta matriz es la información resultante del script recovery layer info. Este archivo está disponible en Repositorio de matrices Juan Bustamante, con el nombre *matriz de data recuperada.xlsx*

B.2. matriz de geoinformación.xlsx

Fig B.2: Trabajo en matriz data recuperada

En esta matriz se puede ver el trabajo en la matriz data recuperada durante el proceso de depuración. Este archivo está disponible en Repositorio de matrices Juan Bustamante, con el nombre *matriz de geoinformación.xlsx*

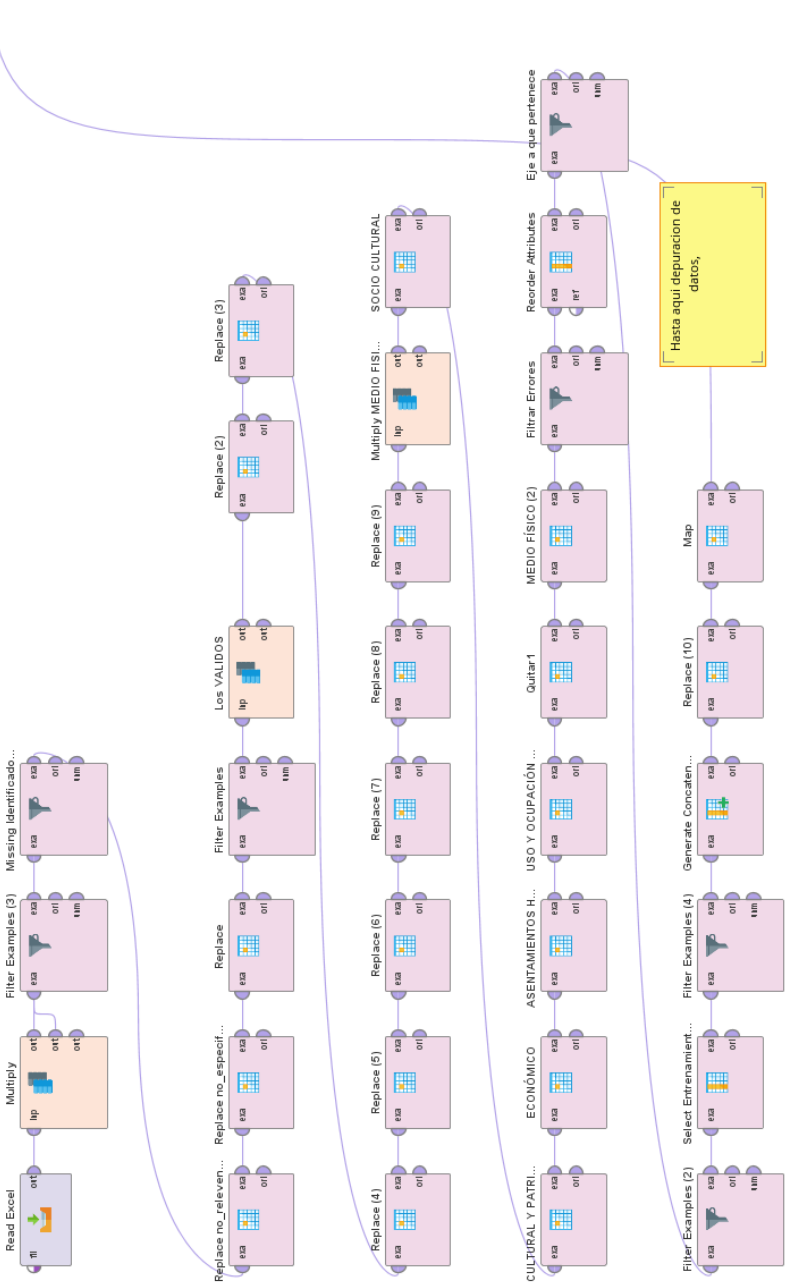


Fig B.3: Proceso desarrollado en RapidMiner

Este es el proceso de minería de datos en RapidMiner. Este archivo está disponible en Repositorio de matrices Juan Bustamante, con el nombre *depuración y nomalización.rmp*

C. Visitas a comunidades

C.1. Visita a San Joaquín:

Fig C.1: Fotografía de visita a la parroquia San Joaquín

C.2. Visita a Sayausí

Fig C.2: Fotografía de visita a la parroquia a Sayausí

D. Reuniones con expertos en Ordenamiento Territorial

D.1. Integrantes del Proyecto principal AVPPGIS

Fig D.1: Integrantes del Proyecto de investigación/vinculación AVPPGIS

D.2. Reuniones con expertos en Ordenamiento Territorial

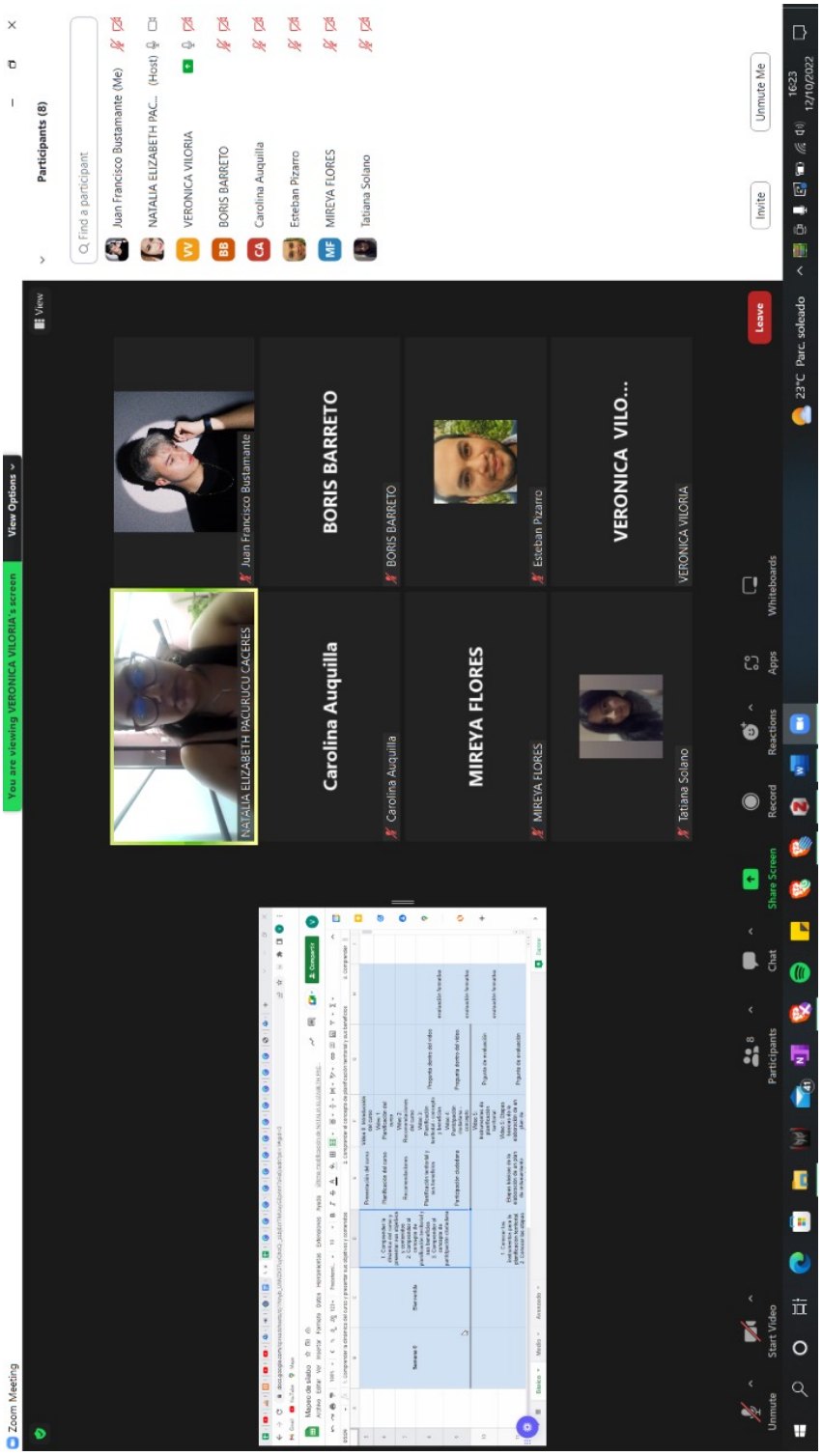


Fig D.2: Reuniones con Ing. Natalia Pacurúcu y demás expertos.

E. Pruebas de usabilidad y evaluación de HCI

E.1. Fotografías de usabilidad técnica

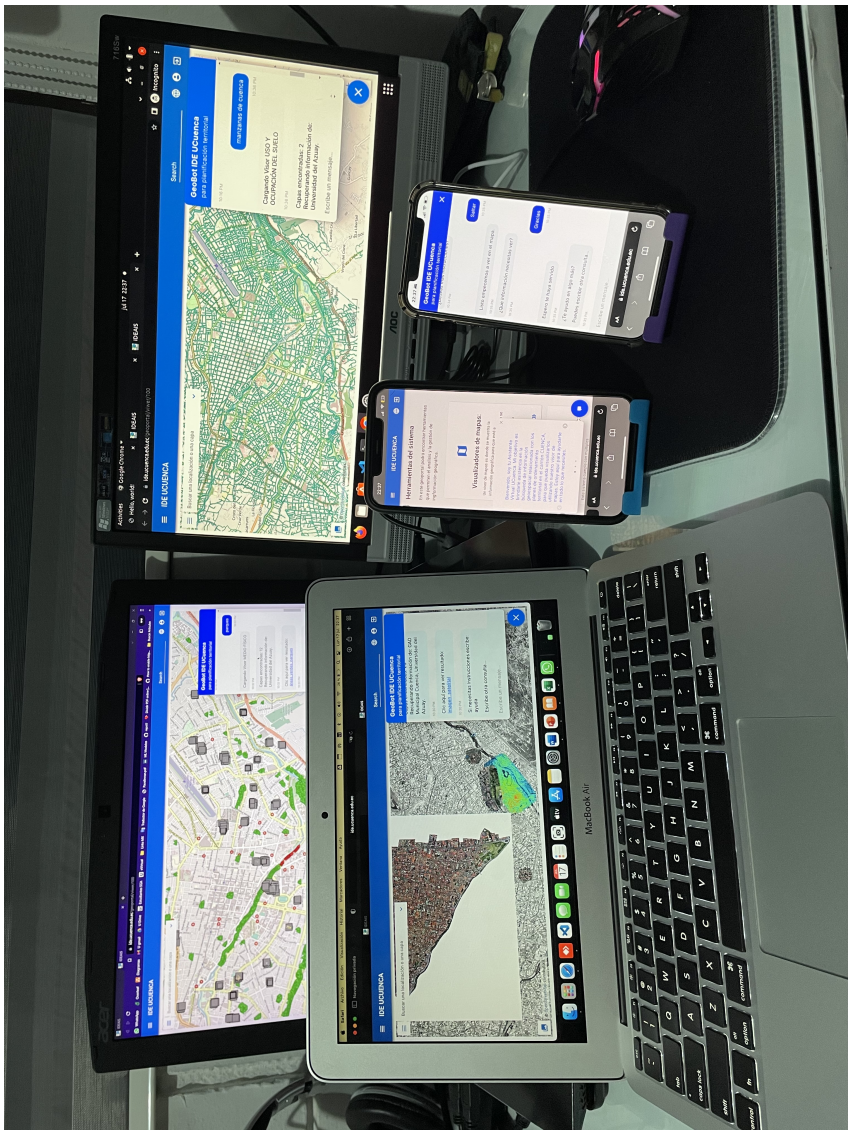
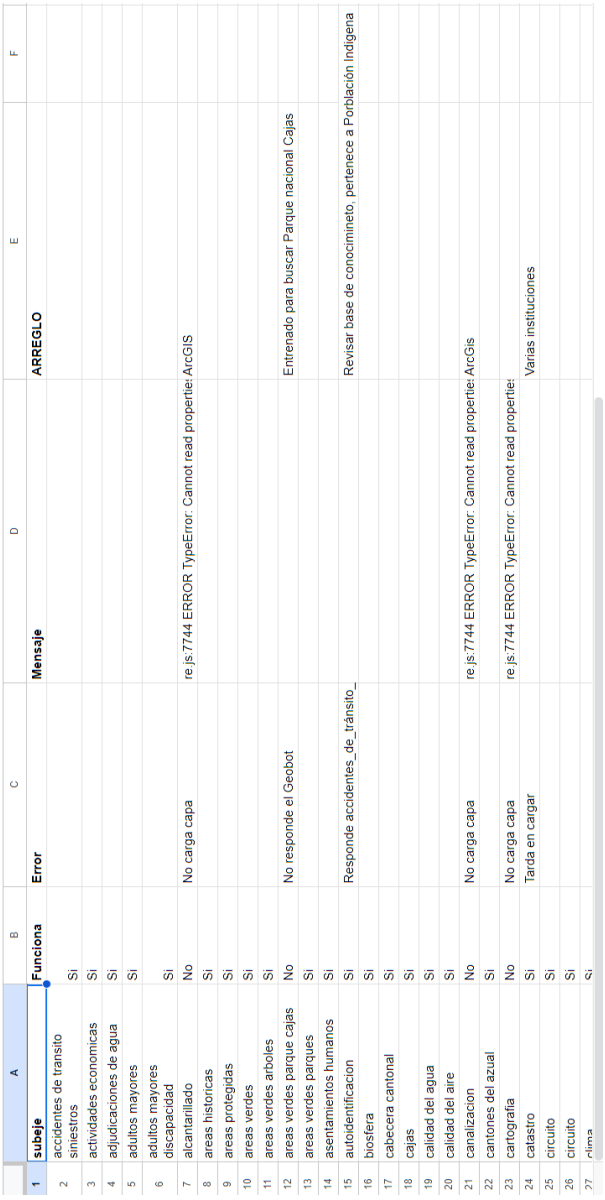


Fig E.1: Fotografía del Asistente Virtual desplegado en varios dispositivos.

Fig E.2: Herramienta IDE del GAD municipal de Cuenca.

E.3. matriz de registro de pruebas unitarias.xlsx



	A	B	C	D	E	F
1	subeje	Funciona	Error	Mensaje	ARREGLO	
2	accidentes de transito siniestros	Si				
3	actividades economicas	Si				
4	adjudicaciones de agua	Si				
5	adultos mayores	Si				
6	adultos mayores discapacidad	Si				
7	alcantarillado	No	No carga capa	re:js:7744 ERROR TypeError: Cannot read properties ArcGIS		
8	areas historicas	Si				
9	areas protegidas	Si				
10	areas verdes	Si				
11	areas verdes arboles	Si				
12	areas verdes parque cajas	No	No responde el Geobot		Entrenado para buscar Parque nacional Cajas	
13	areas verdes parques	Si				
14	asentamientos humanos	Si				
15	autoidentificacion	Si	Responde accidentes_de_tránsito_		Revisar base de conocimineto, pertenece a Población Indigena	
16	biosfera	Si				
17	cabecera cantonal	Si				
18	cajas	Si				
19	calidad del agua	Si				
20	calidad del aire	Si				
21	canalizacion	No	No carga capa	re:js:7744 ERROR TypeError: Cannot read properties ArcGis		
22	cantones del azual	Si				
23	cartografia	No	No carga capa	re:js:7744 ERROR TypeError: Cannot read properties		
24	catastro	Si	Tarda en cargar		Varias instituciones	
25	circuito	Si				
26	circuito	Si				
27	clima	Si				

Fig E.3: Matriz de registro de pruebas unitarias

Este archivo se creó para llevar un registro de las pruebas unitarias. Este archivo está disponible en Repositorio de matrices Juan Bustamante, con el nombre *pruebas unitarias.xlsx*

E.4. Aplicación de la encuesta y retroalimentación de HCI

Fig E.4: Aplicación de Encuestas a técnicos en ordenamiento territorial.