

Generación automática de políticas de diálogo a partir de máquinas de estados para chatbots

Trabajo Terminal No. ____ - ____

*Alumnos: Medina Luqueño Ana Ximena, Zenón Vela Juan José**

Directores: Duchanoy Martínez Carlos Alberto, Suárez Castañón Miguel Santiago

**e-mail: juanjozenon@hotmail.com*

Resumen – En este trabajo terminal se desarrollará un sistema que interprete una lógica de negocios destinada a un chatbot, descrita a través de una máquina de estados finita. Posterior a la interpretación, el sistema generará, con base en dicha lógica de negocios, una política de diálogo de manera automática a través del enfoque de “relleno de espacios”. De esta forma, se acelerará el desarrollo de chatbots al ahorrar el proceso de describir manualmente la política de diálogo. Se integrará por medio de un API externo un procesador de lenguaje natural de libre acceso como módulo del entendimiento del lenguaje que detectará en qué momento se activa una política de diálogo determinada.

Palabras clave – Chatbots, Flujos Conversacionales, Máquinas de Estado, Política de Diálogo.

1. Introducción

Un chatbot, agente conversacional, o sistema de diálogo, es un programa de computadora cuyo propósito es generar respuestas similares a las de los humanos durante una conversación. El diseño de un sistema que sea capaz de lo anterior ha sido uno de los temas más investigados en el campo de la interacción humano-máquina. Los chatbots son diseñados, principalmente, para tener conversaciones con humanos haciendo uso del lenguaje natural y, usualmente, se enfocan en realizar tareas específicas para las que fueron entrenados, dentro de una amplia variedad de operaciones como organizar archivos en una computadora, buscar en la web, agendar citas, entre otros [1].

Los chatbots están siendo ocupados por proveedores de servicio como una forma de ofrecer un servicio al cliente rápido y eficiente. A través de los chatbots, los proveedores de servicio son capaces de proveer de respuestas inmediatas a las peticiones de clientes a cualquier hora del día, cualquier día de la semana. Los chatbots pueden ser usados como la primera opción de soporte en un chat, o ser ofrecidos como alternativa a chatear con humanos [2].

En un chatbot podemos identificar 6 componentes principales [3] como lo muestra la **Figura 1**:

- 1. Decodificador de entrada:** Reconoce la entrada y la convierte en texto (únicamente presente en sistemas de diálogo que no tienen texto como entrada).
- 2. Entendimiento del lenguaje:** Convierte la secuencia de palabras en una representación semántica, o en otras palabras: en un lenguaje que puedan entender los chatbots, que puede ser usada por el gestor de diálogo.
- 3. Gestor de diálogo:** Toma una representación semántica del texto de entrada, lo interpreta según el contexto y crea una representación semántica de la respuesta del sistema.
- 4. Componentes de dominio específico:** Dependiendo de las necesidades del chatbot se requieren interfaces a software externo como una base de datos o un sistema experto.
- 5. Generador de la respuesta:** Construye el mensaje que se transmitirá al usuario, toma las decisiones sobre qué información y cómo debe presentarse, así como la elección de las palabras y la estructura sintáctica.
- 6. Codificador de salida:** Convierte el texto en algún otro medio como audio o imágenes (únicamente presente en sistemas que no utilizan texto como salida).

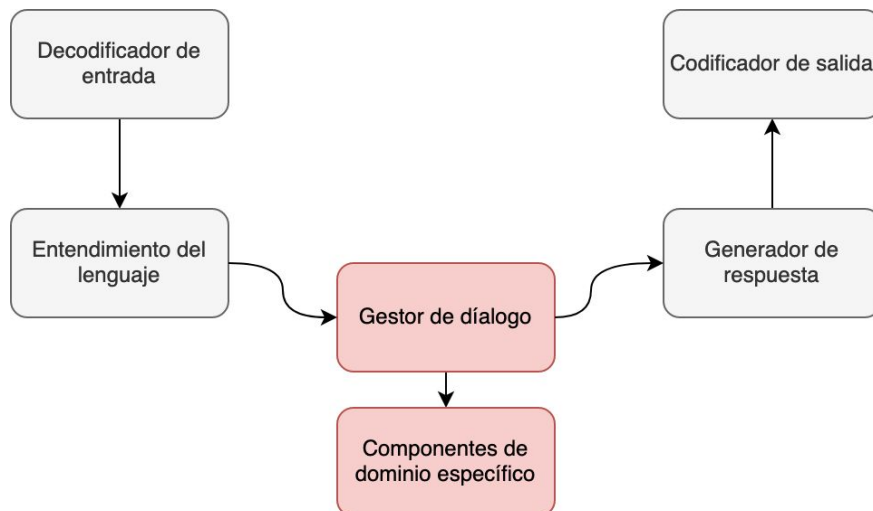


Figura 1. Partes de un sistema de diálogo.

Para el alcance de este Trabajo Terminal, el componente de interés es el gestor de diálogo, resaltado de color rojo en la **Figura 1**. El gestor de diálogo es la parte de un chatbot encargada de ejecutar la lógica del negocio o tarea dependiendo de la representación semántica de la entrada del usuario, es decir, toma una entrada ya procesada por el módulo del entendimiento del lenguaje.

Para poder describir el problema a abordar y plantear su solución, es necesario que se expliquen de forma breve conceptos importantes dentro del módulo del entendimiento de lenguaje, el mismo que será implementado a través de un procesador de lenguaje natural que se encuentre disponible para su libre uso y acceso.

El primer paso que realizan los chatbots cuando reciben una entrada es la interpretación del lenguaje natural. El Procesamiento de Lenguaje Natural (NLP por sus siglas en inglés), es la principal herramienta utilizada dentro del módulo de entendimiento del lenguaje de los chatbots hoy en día. El NLP permite el análisis del lenguaje a través de intenciones, entidades y algunas otras categorías [1].

Una intención o *intent* es la intención del usuario identificada de la entrada que el chatbot ha recibido del usuario. Dependiendo del chatbot, se pueden identificar intenciones de dominio específico para las cuales el chatbot necesitaría entrenamiento en dicho dominio [4]. Por ejemplo, un chatbot para reservación en restaurantes tendría intenciones como reservar mesa, cancelar reservación, o modificar reservación; mientras que un chatbot para solicitar información de trámites en una oficina, tendría intenciones como mostrar todos los trámites, solicitar folio, solicitar aclaración, o consultar estado de un trámite.

Una entidad o *entity* es una variable del lenguaje natural [1], que representa un parámetro de entrada necesario para la ejecución de la intención. Siguiendo con el ejemplo anterior, la hora y nombre del cliente serían entidades de la intención “reservar una mesa en un restaurante”, y el folio y problema serían entidades de la intención “solicitar aclaración” [5].

Analizando el lenguaje natural en estas categorías es como se permite que el gestor de diálogo siga el enfoque del “relleno de espacios”, bajo el cual, la mayoría de los chatbots hoy en día están desarrollados.

El “relleno de espacios” consiste en diseñar el flujo de la conversación de forma que el chatbot haga las preguntas adecuadas para conocer todas las entidades de una intención y así poder ejecutar alguna acción asociada. Para lograrlo, el módulo de entendimiento del lenguaje natural analizará la entrada del usuario por cada turno de la conversación para que después el gestor de diálogo tomando en cuenta la intención y las entidades reconocidas hasta el momento, busque hacer preguntas al usuario que lo lleven a conocer o rellenar los espacios o entidades faltantes.

Para explicar mejor el enfoque del relleno de espacios y algunos de los conceptos mencionados anteriormente, se utilizarán dos ejemplos ilustrativos de una representación de un comando o entrada de texto que un chatbot podría recibir.

Enunciado	Quiero	pedir	una	pizza	hawaiana	mediana	de	masa	rellena	de	queso
Espacios-entidades	-	-	-	-	Sabor	Tamaño	-	-	Masa		
Intención	Ordenar pizza										
Dominio	Restaurantes / Servicio de alimentos / Pizzerías										

Tabla 1. Ejemplo ilustrativo de representación de lenguaje natural

En **Tabla 1** se presenta el primer ejemplo que describe un caso ideal, donde el enunciado de entrada permite identificar la intención y todas sus entidades. Esto no siempre ocurre de esta forma, el usuario pudo haber dado únicamente la intención, o la intención y el tamaño de la pizza, o la intención y la masa de la pizza. En cualquiera de estos casos, el chatbot debe realizar preguntas con el objetivo de reunir toda la información necesaria (sabor, tamaño y masa para este ejemplo). Debe hacerlo de una forma que se contemplen los posibles flujos que pueda presentar la conversación. Un flujo de conversación es el distinto orden en que el chatbot pueda hacer las preguntas y/o que el usuario pueda proveer la información.

Ahora, no sólo en el dominio de servicios de alimentos las entidades de la intención pueden darse en distinto orden. La **Figura 2** ilustra cómo una conversación para reservar un boleto de avión puede darse de formas distintas y que el gestor de diálogo debe reconocer las entidades ya ingresadas por el usuario así como las que aún faltan por conocer.

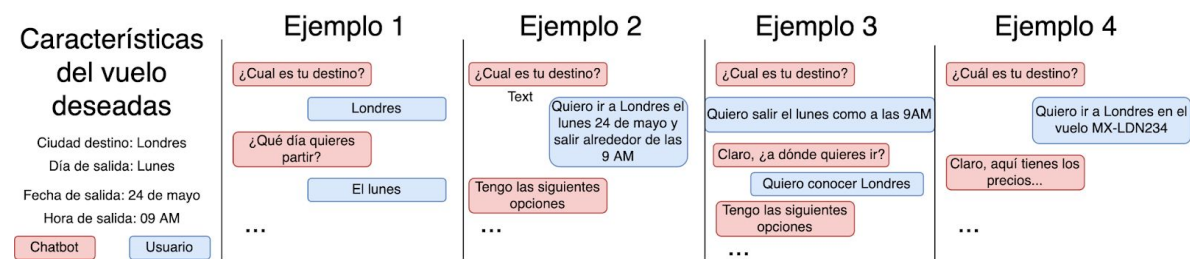


Figura 2. Ejemplo de flujos de una conversación para la reserva de un vuelo dentro del enfoque de relleno de espacios

El proceso que se sigue para construir un gestor de diálogo que siga el enfoque de “relleno de espacios” es el que se describe a continuación:

1. Definir la lógica de negocios a través de estructuras de control (normalmente máquinas de estados finitas).
2. Definir una interfaz conversacional o política de diálogo que sea equivalente a la estructura de control del **paso 1**, la cual se encargará de determinar la siguiente acción del chatbot con base en la intención y las entidades correspondientes al punto específico de la conversación tomando en cuenta los posibles flujos de conversación.

De forma breve, podemos definir una **política de diálogo** como el proceso de generación de la siguiente acción o respuesta de un chatbot, dependiendo de la entrada del usuario [6]. Esta es la que determina las preguntas que deben hacerse al usuario según la información ya obtenida hasta cierto momento de una conversación, lo que fue mostrado en la **Figura 2**.

El enfoque de “relleno de espacios” aunque ha probado ser confiable para el desarrollo de chatbots, sin embargo, requiere de mucho tiempo para su desarrollo ya que en muchas ocasiones se tiene que describir o codificar todos los flujos de conversación manualmente y así el tiempo de desarrollo aumenta conforme crece el número de intenciones y la cantidad de entidades para una determinada intención.

Lo anterior representa un problema porque, para construir el gestor de diálogo se necesitan crear dos estructuras lógicas equivalentes: una representando la lógica de negocios de forma general y la otra representando la política de diálogo con base en la misma lógica de negocios.

El presente trabajo terminal abordará el problema anterior proponiendo una forma de acelerar la construcción de la política de diálogo al agilizar la definición de los flujos de conversación, pidiéndole al usuario únicamente generar una sola estructura lógica para la lógica de negocios y así ahorrándole la conversión a política de diálogo.

Para ello, se creará un sistema que genere de forma automática políticas de diálogo. Éstas se crearán a partir de máquinas de estados finitas que definan la lógica de negocios a implementar a través de una interfaz gráfica web que permita definir de forma visual dicha lógica de negocios. Es decir, el sistema pretende agilizar el desarrollo de chatbots mediante la automatización del proceso de conversión de una lógica de negocios a una política de diálogo.

Algunos sistemas relacionados que se han desarrollado son:

Con base en una consulta tanto de directorios internos del instituto y del estado del arte en diversas fuentes, se han encontrado los siguientes sistemas que abordan problemas similares y se listan algunos a continuación en la **Tabla 2**.

SOFTWARE	CARACTERÍSTICAS	PRECIO EN EL MERCADO	DEFICIENCIAS
Kite [5]	Sistema que construye plantillas de chatbots con base en interacciones con aplicaciones móviles. Este proceso abarca la extracción automática de las reglas de negocio, la generación de la política de diálogo y un programa “plantilla” editable del chatbot.	No disponible.	No está abierto al público.
Caligraphi [7]	Modelador gráfico de diálogos para sistemas de diálogo por voz.	No está disponible.	El proyecto data del año 2011 pero hoy en día no existe.
IBM Cloud – Watson Assistant [8]	Aplicación web para el desarrollo de chatbots. Permite definir lógicas de negocios como árboles de decisión	Versión gratuita (limitada), versión estándar de \$0.0025USD por mensaje o cotizaciones especiales.	Permite la generación de políticas de diálogo de forma gráfica pero manualmente.
Dialogflow [9]	Aplicación web para desarrollo de chatbots que permite definir flujos conversacionales como árboles de decisión.	Gratuito hasta un número limitado de solicitudes.	No hace una generación automática de la política de diálogo, es necesario crear el árbol de decisión manualmente.
Tesis de UPIICSA– “Construcción de un prototipo de programa personalizado de tipo Chatbot en ambiente java con un lenguaje natural” [10]	Prototipo de un chatbot que pretende realizar consultas a bases de datos del IPN con base en una entrada en lenguaje natural.	No está implementado.	No se implementó. El trabajo sólo funciona para un dominio específico (bases de datos del IPN) y por lo tanto cambiar el dominio requeriría de un nuevo desarrollo por completo.

TT de ESCOM – CATTBot para la divulgación de información [11]	Chatbot para consulta de información de trámites escolares de la CATT en la ESCOM a través de lenguaje natural.	No se diseñó para estar dentro de un esquema de cobro.	Ya no se encuentra funcionando. Es un chatbot de dominio específico, para una plataforma específica, no es reutilizable para otros dominios.
Chatfuel [12]	Plataforma para desarrollo de chatbots sin utilizar código, para implementarlos en Facebook Messenger.	Gratis hasta 1000 suscriptores por chatbot, con más suscriptores y usuarios los precios empiezan en \$15 USD.	Únicamente produce chatbots para Facebook Messenger con tecnologías propias no transparentes, dejando de lado otras redes sociales y plataformas donde se pueden implementar chatbots.
TT de ESCOM – SaaS para construcción de chatbots	Aplicación para desarrollo gráfico de chatbots sin necesidad de usar códigos, permitiendo elegir el procesador de lenguaje natural a utilizar.	Herramienta desarrollada para la experimentación y ayuda de otros proyectos en el instituto, por lo tanto, no tiene costo.	A pesar de ser un sistema funcional y completo, la construcción de la política de diálogo se realiza de forma manual gráficamente.

Tabla 2. Resumen de productos similares

2. Objetivo

General: Desarrollar un sistema para generar de forma automática políticas de diálogo para ser utilizadas por un chatbot, con base en la lógica de negocios descrita a través de una máquina de estados finita definida en una interfaz gráfica web, para acelerar el proceso del desarrollo de un chatbot de dominio específico al evitar tener que definir dos estructuras lógicas equivalentes.

Particulares:

- Realizar una investigación de temas relacionados al trabajo terminal.
- Analizar las plataformas, técnicas, métodos y algoritmos que puedan ser empleados para el diseño y desarrollo del sistema.
- Diseñar un método para interpretar la máquina de estados finita que describe la lógica de negocios.
- Implementar el método de interpretación de máquinas de estados finitas.
- Desarrollar la interfaz gráfica web que permita definir gráficamente la lógica de negocios en forma de máquina de estados finitas.
- Diseñar un método para generar flujos conversacionales con base en la interpretación de la lógica de negocios.
- Implementar el método de generación de flujos conversacionales.
- Diseñar un método de generación de políticas de diálogo que funcionen bajo el enfoque de relleno de espacios.
- Implementar la generación de políticas de diálogo.
- Diseñar un módulo de integración de la llamada API de un procesador de lenguaje natural de libre acceso, para detectar la activación de intenciones.
- Desarrollar el módulo de integración del procesador de lenguaje natural.
- Diseñar un método para probar las políticas de diálogo generadas.
- Probar el desempeño de las políticas de diálogo haciendo uso del método diseñado.

3. Justificación

Como ya hemos mencionado, el uso de chatbots por parte de proveedores de servicio como un medio de ofrecer un servicio al cliente rápido y eficiente cada vez es más común ya que presenta una alternativa rápida

de respuestas instantáneas disponible en cualquier horario. Esto, entre otros factores como una mejor relación costo-eficiencia, además de la posibilidad de mejorar la experiencia del cliente, ha motivado a los proveedores de servicio a la adopción de chatbots. Existe evidencia de que algunos chatbots han sido apreciados por consumidores por sus respuestas inmediatas y ayuda accesible [2].

Aún con este incremento en la popularidad de los chatbots, el proceso de desarrollo para el método del “relleno de espacios” se ha mantenido casi igual, diseñándose manualmente los flujos de conversación para las políticas de diálogo, incluso en los servicios más populares para el desarrollo de chatbots como Dialogflow, IBM-Watson, entre otros, donde a pesar de ofrecer la funcionalidad de diseñar el diálogo de forma gráfica e intuitiva, sigue requiriendo de un esfuerzo proporcional a la cantidad de intenciones y entidades.

Este trabajo terminal surge de la necesidad de acelerar y disminuir el esfuerzo requerido para la implementación de un chatbot, facilitando el proceso específicamente entre la parte de la definición de la tarea como máquina de estados finitas y la construcción o generación de la política de diálogo que desarrolle dicha tarea de forma que se contemplen los posibles flujos que la conversación puede tomar que, en muchos casos, representa una gran parte del tiempo total de la construcción de un chatbot.

Dentro del conocimiento de los alumnos y directores que proponemos este trabajo terminal y después de una investigación en algunas fuentes, únicamente se identificó, hasta el momento, un sistema similar que resuelva la problemática específica de generar una política de diálogo: Kite, mencionado en la **Tabla 1**. Sin embargo, este sistema no se encuentra disponible al público en este momento, por lo cual consideramos que resolveríamos una problemática poco atacada.

Debido a que el sistema propuesto pretende acelerar la implementación de chatbots y que la popularidad de estos se encuentra en constante crecimiento, este sistema tendría un atractivo para cualquier empresa o desarrollador que se encuentre con la tarea de implementar un chatbot enfocado a automatizar servicios a clientes.

4. Productos o Resultados esperados

Al término de este trabajo terminal se habrá desarrollado un sistema que funcione de la siguiente manera:

1. El usuario defina la lógica de negocios a través de una interfaz gráfica web la cual producirá una estructura lógica que el sistema pueda interpretar.
2. Interprete la lógica de negocios a partir de la representación de la máquina de estados generada en el punto anterior.
3. Genere una política de diálogo que implemente la lógica de negocios.

A continuación, la **Figura 3** describe el sistema a realizar.

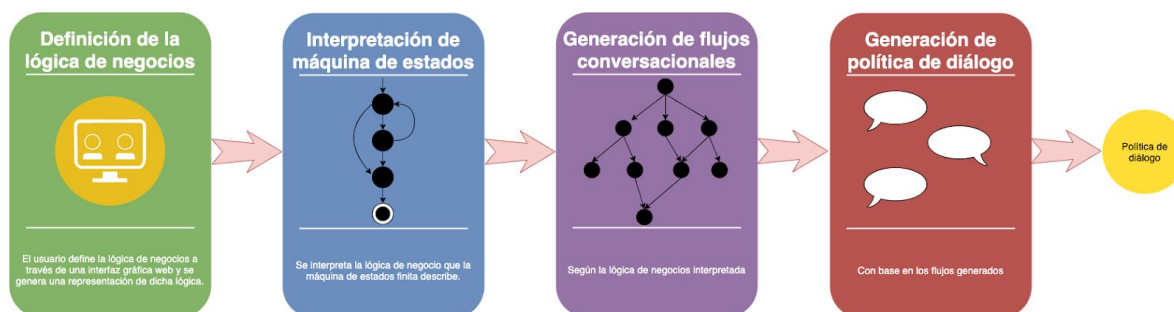


Figura 3. Arquitectura de la generación de políticas de diálogo

- **Análisis de la entrada:** Se interpreta la máquina de estados finita donde se encuentra descrita la lógica del negocio.
- **Generación de flujos de conversación:** A partir de la lógica del negocio interpretada, se generan flujos de conversación.
- **Generación de política de diálogo:** A partir de los flujos de conversación, se genera una política de diálogo que implementa la lógica del negocio.

Para que una política generada por el sistema pueda ejecutarse, se hará uso de un procesador de lenguaje natural de libre acceso mediante una llamada API. Este procesador de lenguaje natural permitirá identificar cuándo una política es activada mediante la clasificación de intenciones; así como para analizar la entrada para identificar entidades.

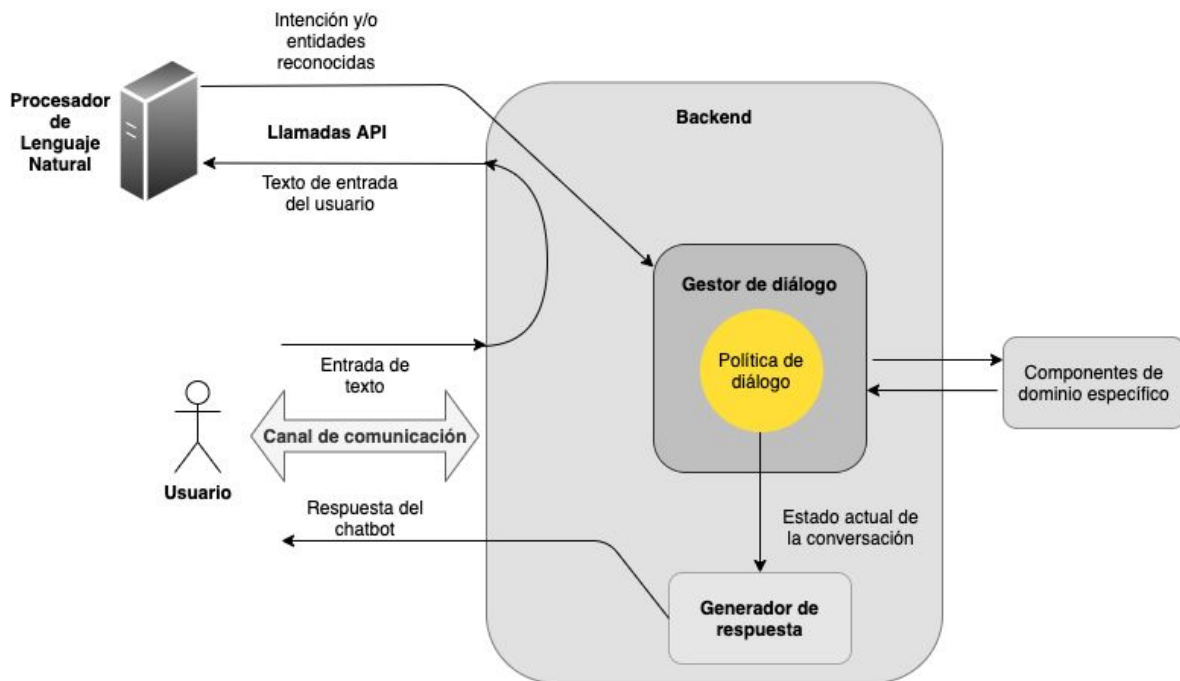


Figura 4. Diagrama de la integración de una política de diálogo generada por el sistema propuesto

En la **Figura 4** mostramos cómo interactúa el usuario final (el público del desarrollador de chatbots) con un chatbot que integre una política de diálogo generada por el sistema. El backend se encarga de recibir las entradas del usuario, hacer las llamadas API al módulo externo del entendimiento del lenguaje, enviar las intenciones y entidades reconocidas a la política de diálogo, y entregar la respuesta generada a través del mismo canal de comunicación. El módulo externo de entendimiento de lenguaje recibe, por parte del backend, las entradas del usuario y se encarga de categorizarla en intenciones y entidades. El gestor de diálogo toma como entrada la respuesta de la llamada API y activa la política de diálogo, después le pasa las entidades que regresó la llamada API como parámetros a la política de diálogo. La política de diálogo se encarga de reconocer el flujo de la conversación y reconocer en qué estado se encuentra la misma según las entidades que reciba desde la respuesta de la llamada API; así como indicar este estado de la conversación al generador de la respuesta. El gestor de diálogo también interactúa con los componentes de dominio específico, como pueden ser bases de datos del negocio, APIs y cualquier otro recurso de consulta o escritura que el chatbot necesite. Por último, el generador de respuesta se encarga de entregar la respuesta del sistema a través del canal de comunicación.

Para que la arquitectura anterior esté completa, el desarrollador del chatbot será el encargado de contar con un servidor donde ejecutar la política de diálogo y el módulo de procesamiento de lenguaje natural, así como de realizar la integración del canal de comunicación sobre el cual se desea que funcione el chatbot, y también de

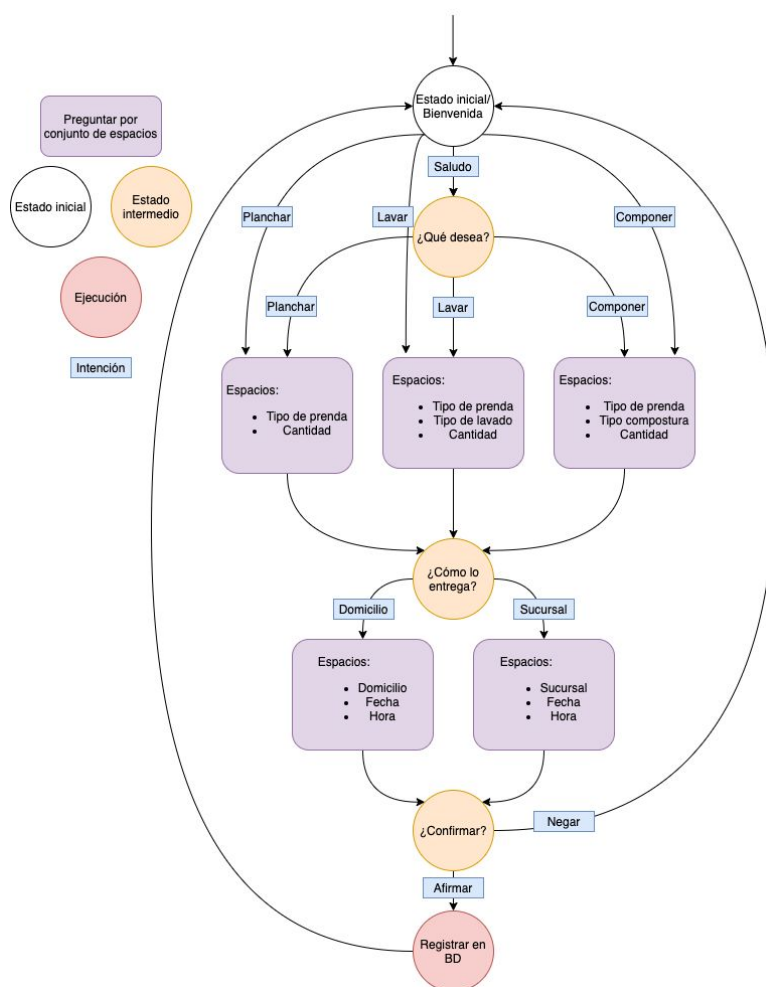
desarrollar los componentes de dominio específico, que pueden ser, por ejemplo API de un servicio que haga registros o consultas en una base de datos.

De forma que los productos esperados son:

1. Interfaz gráfica web para definir de forma visual la lógica de negocios y que genere una representación de dicha lógica.
2. Módulo de interpretación de la representación de las máquinas de estados finitas del punto 1 para extracción de lógica de negocio.
3. Generador de flujo conversacional basado en la lógica de negocios.
4. Módulo para integrar la llamada API de un procesador de lenguaje natural externo que se encargue del reconocimiento de intenciones y entidades.
5. Generador de políticas de diálogo con base en las tareas descritas en la lógica de negocio.
6. Reporte de pruebas con base en el caso de estudio presentado en la siguiente subsección.
7. Manual de usuario.
8. Reporte técnico con el caso de estudio presentado en la siguiente subsección.

4.1 Caso de estudio

Para comprobar que el sistema funcione correctamente se desarrollará un chatbot de muestra para reservaciones de servicio en una tintorería, descrito por la siguiente máquina de estados finita descrita con una nomenclatura propia:



Donde:

- Círculo blanco: Es el estado inicial del chatbot, da el mensaje de bienvenida o el primer mensaje.
- Rectángulo morado: Representa un estado que describe un conjunto de espacios que corresponden a un tipo de entidad. Este estado es una abreviación de los flujos conversacionales que pueden ocurrir para que el chatbot obtenga las entidades requeridas.
- Rectángulo azul: Representa la identificación de la intención en la entrada del usuario especificada en su etiqueta.
- Círculo naranja: Describe un estado intermedio que requiere de la identificación de una intención para continuar con la conversación.
- Círculo rojo: Describe un estado que ejecutará alguna acción y puede usar las entidades recolectadas durante la conversación.

El chatbot para el caso de estudio seguirá el enfoque de “relleno de espacios” y deberá poder registrar reservaciones de servicio para una tintorería, donde los servicios pueden ser lavado, planchado y compostura. Debe tomar en cuenta las consideraciones sobre los posibles flujos conversacionales para pedir al usuario un conjunto de entidades que corresponden a una intención, tal como se describió en la introducción para el ejemplo de la intención “Ordenar pizza” y “Reservar vuelo”.

Para hacer un análisis que muestre que el sistema reduce el tiempo y facilita el desarrollo de un chatbot, se realizará una comparativa con algunos de los sistemas descritos en la **Tabla 2** desarrollando el mismo chatbot propuesto para este caso de estudio, mencionando las tareas más repetitivas y la alternativa que nuestro sistema ofrece.

5. Metodología

Durante el desarrollo de este sistema como trabajo terminal se seguirá la metodología SCRUM, permite un desarrollo y gestión del proyecto de forma controlada, flexible y eficaz, al ser una metodología de tipo ágil y colaborativa, basada en las entregas parciales regulares del sistema. Además de que el uso de SCRUM es frecuente en sistemas donde los requerimientos están en constante cambio, podrá influir positivamente en el desarrollo de un proyecto donde el diseño específico del mismo requiere de una gran cantidad de análisis de información y procesos como es este trabajo terminal.

En esta metodología existen 3 roles principales: Product Owner, SCRUM Master y Equipo de Desarrollo. El Product Owner es el encargado de validar que el trabajo realizado por el equipo corresponda a los objetivos del proyecto y quien prioriza las tareas a realizar, generalmente es quien tiene el mayor conocimiento sobre el tema en el equipo. El SCRUM Master es quien se encarga de que el equipo entienda y esté siguiendo la metodología. El equipo de desarrollo es el encargado de realizar y estimar el tiempo que tomarán las tareas priorizadas por el Product Owner.

Para este Trabajo Terminal, los roles de Product Owner y SCRUM Master serán adquiridos por los directores. El rol del equipo de desarrollo será tomado por los alumnos que lo proponen. Generalmente, aunque no siempre, los equipos de trabajo no tienen sub-equipos o especialistas, por lo que en la mayoría de los casos se tienen responsabilidades compartidas.

En esta tecnología existe un artefacto o herramienta llamada Product Backlog o simplemente Backlog donde se reúnen las características y requerimientos del sistema representados como tareas a realizar. En SCRUM se realizan sprints o iteraciones de 2 a 4 semanas que son consideradas como el núcleo de la metodología. A partir del Backlog se seleccionan las tareas que se deben trabajar durante el sprint, a este conjunto de las tareas a trabajar durante el sprint se le conoce como el Sprint Backlog.

El equipo de desarrollo se encarga de realizar las tareas del Sprint Backlog y se realiza una junta interna diaria para evaluar el progreso y posteriormente dar a conocer al resto del equipo si hay algún pendiente que los bloquee para continuar. Al término de cada Sprint se realizan un conjunto de reuniones, la primera es el Sprint Review con todos los integrantes del proyecto, donde se habla del estado del proyecto y se define cualquier tarea nueva que deba agregarse al Backlog. La segunda es el Sprint Planning, donde se deciden las tareas a

incluir en el siguiente Sprint Backlog. Y por último la Sprint Retrospective, donde se hace una evaluación sobre la forma en que el equipo ha aplicado la metodología[13].

La **Figura 5**. Metodología SCRUM pretende describir de forma gráfica la metodología a usar durante el proyecto, donde lo más destacable es que al final de cada iteración o *sprint*, los integrantes del equipo realizan las reuniones correspondientes que permitan el progreso del proyecto.



Figura 5. Metodología SCRUM

Durante el desarrollo de este Trabajo Terminal se realizarán Sprints de 2 semanas de duración, de forma que el Product Backlog tendrá que completarse en 16 Sprint aproximadamente. Las Sprint Review se harán al término de cada Sprint pero las Sprint Retrospective se realizarán cada 2 meses.

6. Cronogramas

Nombre de la alumna: Medina Luqueño Ana Ximena

TT No.: -

Título del TT: Generación automática de políticas de diálogo a partir de máquinas de estado para chatbots.

[illegible]

específicas para cumplir los puntos anteriores.									
Sprint Planning 11 y 12 enfocados a: <ul style="list-style-type: none"> Implementación del método de generación de políticas de diálogo. Diseño del método para probar políticas de diálogo. Pruebas de las políticas de diálogo. Definición de tareas específicas para cumplir los puntos anteriores. 									
Sprint Planning 13 y 14 enfocados a: <ul style="list-style-type: none"> Pruebas de las políticas de diálogo. Reporte de pruebas de las políticas de diálogo. Definición de tareas específicas para cumplir los puntos anteriores. 									
Sprint Planning 15 y 16 enfocados a: <ul style="list-style-type: none"> Reporte de pruebas de las políticas de diálogo. Definición de tareas específicas para cumplir el punto anterior. 									
Realizar tareas del Sprint Backlog correspondiente.									
Sprint review (Cada 2 semanas)									
Sprint Retrospective									
Pruebas del sistema, caso de estudio y reingeniería.									
Redacción del reporte técnico y manual de usuario.									
Evaluación de TT II									

Nombre del alumno: Zenón Vela Juan José

TT No.: ____ - ____

Título del TT: Generación automática de políticas de diálogo a partir de máquinas de estado para chatbots.

Actividad	Año
-----------	-----

[illegible]

y reingeniería.									
Redacción del reporte técnico y manual de usuario.									
Evaluación de TT II									

7. Referencias

- [1] K. Ramesh, S. Ravishankaran, A. Joshi y K. Chandrasekaran, «A Survey of Design Techniques for Conversational Agents,» *Springer Nature Singapore Pte Ltd*, 2017.
- [2] K. Kvale, O. A. Sell, S. Hodnebrog y A. Følstad, «Improving Conversations: Lessons Learnt from Manual Analysis of Chatbot Dialogues,» 2020.
- [3] S. Arora, K. Batra y S. Singh, «Dialogue System: A Brief Review,» 2013.
- [4] A. Singh, K. Ramasubramian y S. Shivam, Building an Enterprise Chatbot, Apress, 2019.
- [5] T. J.-J. Li y O. Riva, «Kite: Building Conversational Bots from Mobile Apps,» 2018.
- [6] H. Chen, X. Liu, D. Yin y J. Tang, «A Survey on Dialogue Systems: Recent Advances and New Frontiers,» 2018.
- [7] G. Bertrand, F. Nothdurft, F. Honold y F. Schüssel, «CALIGRAPHI - Creation of Adaptive dialogues using a Graphical Interface,» 2011.
- [8] IBM, «Documentos de IBM Cloud / Watson Assistant,» [En línea]. Available: <https://cloud.ibm.com/docs/services/assistant?topic=assistant-index>. [Último acceso: 27 Febrero 2020].
- [9] Google Inc., «Documentación de DialogFlow,» [En línea]. Available: <https://cloud.google.com/dialogflow/docs>. [Último acceso: 27 Febrero 2020].
- [10] M. A. Limón Pérez, «Construcción de un prototipo de programa personalizado de tipo chatbot en ambiente java con un lenguaje natural,» 2016.
- [11] J. Morales Torres, «CATTBot para la divulgación de información,» 2018.
- [12] chatfuel, «chatfuel,» [En línea]. Available: <https://chatfuel.com/>. [Último acceso: 02 Marzo 2020].
- [13] K. Schwaber y J. Sutherland, «La guía de SCRUM,» 2013.

8. Alumnos y directores

Medina Luqueño Ana Ximena.- Alumna de la carrera de Ingeniería en Sistemas Computacionales en la ESCOM. Boleta: 2016630248, Tel. 551243 8926, email: ximena.medluq@gmail.com

Firma: _____

Zenón Vela Juan José.- Alumno de la carrera de Ingeniería en Sistemas Computacionales en la ESCOM. Boleta: 2016630415, Tel 5510138705, email: juanjoze0424@gmail.com

Firma: _____

Duchanoy Martínez Carlos Alberto.- Recibió el grado de Ingeniero en Mecatrónica por parte de la UPIITA del Instituto Politécnico Nacional, el grado de Maestro en Ciencias en Ingeniería de Cómputo y Doctor en Ciencias en Computación por parte del Centro de Investigación en Computación. Ha trabajado en la industria privada como Director de Investigación y Desarrollo en la empresa GusChat enfocado en el desarrollo de tecnologías para la comprensión del lenguaje natural para su uso en robots conversacionales. Actualmente es Catedrático CONACYT asignado como profesor visitante en el Centro de Investigación en Computación. Su área de experiencia incluye Inteligencia Artificial, Redes Neuronales Profundas, Optimización y Diseño Mecatrónico. Tel. 5512566602, email: duchduchanoy@gmail.com


Firma: _____

Suárez Castañón Miguel Santiago.- Dr. En Ciencias de la Computación en el Instituto Politécnico Nacional en 2005, M. en C. de la UNAM en 2001, Ing. En Cibernética y Ciencias de la Computación en la Universidad La Salle AC en 1991, Profesor en la ESCOM del IPN (Sección de Posgrado e Investigación) desde 2009, Áreas de Interés: Ingeniería de Software. Tel: 57-29-60-00 Ext. 52043, Email: sasuares@prodigy.net.mx.

Firma: _____

CARÁCTER: Confidencial
FUNDAMENTO LEGAL: Artículo 11 Fracc. V y Artículos 108, 113 y 117 de la Ley Federal de Transparencia y Acceso a la Información Pública.
PARTES CONFIDENCIALES: Número de boleta y teléfono.

9. Acuses de recibo del presente protocolo

Juan ZV 
Revisión Protocolo 2020-A072
Para: ulises.velez@gmail.com, Cc: Ana Luqueño


ayer, 22:03
[Detalles](#) **JZ**

Buenas noches profesor Ulises,

Le mandamos la nueva versión donde abordamos los detalles que platicamos hoy por la tarde.

Cualquier otro comentario le pedimos que nos lo haga saber. De otra forma le pedimos que nos responda con un correo de acuse de recibido y poder cumplir con la fecha límite de entrega del día de mañana.

Muchos saludos
Ana Ximena Medina Luqueño
Juan José Zenón Vela


ProtocoloA072-10septi...re2.pdf

Ulises Velez Saldaña
Re: Revisión Protocolo 2020-A072
Para: Juan ZV, Cc: Ana Luqueño

12:12
[Detalles](#) **US**

RECIBIDO.
[Ver más de Juan ZV](#)

Acuse de recibo Sinodal M. en C. Velez Saldaña Ulises

Miriam Pescador Rojas
RE: Evaluación propuesta Trabajo Terminal 2020-A072
Para: Juan ZV, ulises.velez@gmail.com, tareasimg@gmail.com, Cc: Ana Luqueño

ayer, 1:33
[Detalles](#) **MR**

Acuso de recibido y no requiero que realicen más modificaciones.

Saludos cordiales

De: Juan ZV <juanzo0424@gmail.com>
Enviado: miércoles, 9 de septiembre de 2020 23:03
[Ver más de Juan ZV](#)

Acuse de recibo Sinodal M. en C. Pescador Rojas Miriam

Iván Mosso
Re: Evaluación propuesta Trabajo Terminal 2020-A072
Para: Miriam Pescador Rojas, Cc: Juan ZV, ulises.velez@gmail.com, Ana Luqueño

ayer, 10:24
[Detalles](#) **IM**

RECIBIDO. Sin problema de mi parte.
[Ver más de Miriam Pescador Rojas](#)

Acuse de recibo Sinodal M. en C. Mosso García Iván Giovanni