Avances en Interacción Humano-Computadora, 2022, Año 7, No. 1, pp. 17-20
http://doi.org/10.47756/aihc.y7i1.121.g149

Dialog modeling of domain specific cognitive assistants

J. Guadalupe Ramos Díaz*, Juan Jesús Ruiz Lagunas, Adrián Núñez Vieyra, José Manuel Cuin Jacuinde, Juan Carlos Olivares Rojas

Published: 30 November 2022

Abstract
Intelligent virtual assistants of software are becoming more common. Frequently, the communication interface is dialogue which is expressed in natural language. When an assistant consists of questions that are associated with a single answer, the abstraction of the dialogue is straightforward, however, when there are conversations with multiple bifurcations in the flow of communication, the representation and documentation is complex. In this sense, there is a lack of formalities that allow a dialogue to be accurately represented and based on them, to proceed with its programming in an industrial framework. In this work we propose the use of finite state machines to model complex dialogues. Thus, we introduce a formal model for the representation of dialogue and its mapping to a real assistant, using an industrial framework. We believe that our approach will make it easier to design and to program assistants with complex interaction, enabling better adoption of them.

Keywords:
Virtual assistant; Dialogue modeling; Dialogue; Chatbot; Formal methods; Formal modeling; Dialog design.

1 Introducción
El uso de asistentes virtuales como interfaz de los sistemas computacionales modernos es cada día más factible e incluso podría decirse que es deseable en muchas ramas de la industria. Ciertamente, el procesamiento computacional del lenguaje hablado ha sido estudiado desde hace muchos años. Sin embargo, el interés reciente puesto por las grandes compañías de tecnología de información en el mundo, como Google [3], ha dado un impulso definitivo, dando lugar a *frameworks* maduros para la construcción de asistentes.

De manera adicional, se ha pretendido dotar de mayores capacidades de autonomía a los asistentes virtuales, a través de la programación de acciones de respuesta al diálogo expresado por el ser humano, dando lugar al término "asistente cognitivo" [6], en [5] se puede encontrar un análisis detallado de este género de software. Los asistentes cognitivos representan una nueva ola de transformaciones en la industria, toda vez que su adopción cambiará la forma en la que los seres humanos, no sólo realizan interacción con la tecnología, sino también la manera en la que el ser humano delega tareas a los asistentes.

Una parte fundamental de los asistentes cognitivos es el diálogo. En este sentido ha habido un gran esfuerzo por la creación de asistentes para la comprensión general, de cualquier tópico, del habla humana. En el mundo comercial destacan Siri de Apple, Cortana de Microsoft, Alexa de Amazon y Google Assistant.

No obstante, para pequeñas y medianas organizaciones con una misión definida, por ejemplo, para ventas de un tipo de producto, no parece necesario que se disponga de un asistente que pueda entender amplias formas de expresión hablada. En este caso particular se percibe, más bien, la necesidad de un asistente que sea capaz de atender procesos específicos de la empresa. Es decir, se requiere de un asistente que entienda un conjunto particular de órdenes y con capacidades de respuesta a tales mandatos. A este tipo de asistentes les llamaremos: Asistentes Cognitivos de Dominio Específico (ACDE)---en contraparte con Asistentes Cognitivos de Propósito General---, emulando el concepto de Lenguajes de Dominio Específico del campo de lenguajes de programación [2].

De esta manera, un ACDE es un asistente cognitivo que atiende un vocabulario y un conjunto de frases que son específicas de un campo de aplicación. Es importante categorizar este tipo de asistentes ya que existen herramientas muy maduras para construirlos y se echa en falta una metodología que permita diseñar el diálogo que se necesita que ejecute el software. Es decir, se requieren modelos formales para documentar el diálogo que desempeñará un asistente y al mismo tiempo, el diagrama o bosquejo resultante debe servir como herramienta de especificación y comunicación en un equipo de desarrollo de software.

Para el desarrollo del diálogo de asistentes cognitivos de dominio específico se pueden emplear una gran variedad de *frameworks* industriales como son: Dialogflow, Chatfuel, Microsoft Bot Framework, Wit.ai, IBM Watson, Pandorabots, Botpress, Botkit, RASA Stack, ChatterBot, entre otros. Antes de emprender el esfuerzo por desarrollar un asistente deberían revisarse las ventajas de la gama de generadores de asistentes en función de lo que se pretenda construir. No obstante, encontramos en la herramienta Dialogflow de Google [3] una serie de elementos que permiten trasladar el diálogo del dominio de aplicación a un autómata finito determinista y a partir del formalismo llevar a cabo

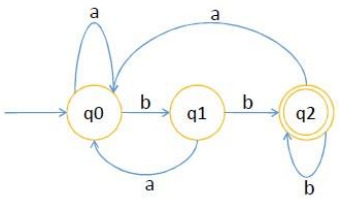
Ramos Díaz, J. Guadalupe., Ruiz Lagunas, Juan Jesús., Núñez Vieyra, Adrián., Cuin Jacuinde, José Manuel., Olivares Rojas, Juan Carlos.
Tecnológico Nacional de México / Instituto Tecnológico de Morelia
Morelia, Michoacán, México.
Email: {juan.rl, adrian.nv,jose.cj,juan.or}@morelia.tecnm.mx
Correspondence: jose.rd@morelia.tecnm.mx

la captura en el *framework*, por lo que consideramos que es una opción conveniente para el desarrollo de ACDEs.

En el presente trabajo planteamos un procedimiento para llevar a cabo el diseño del componente de diálogo de los asistentes cognitivos con base en máquinas de estados finitos (autómatas finitos deterministas), para ello, en la Sección 2 presentamos la definición formal de autómatas finitos deterministas, posteriormente, en la Sección 3 planteamos las fases de construcción de asistentes, hacemos énfasis en la etapa que tiene que ver con reconocimiento de oraciones que transmiten órdenes, enseguida citamos los componentes de Dialogflow que nos son útiles para construir un asistente con diálogo, a continuación, definimos los autómatas finitos deterministas de diálogo e incluimos un ejemplo. Finalmente, en la Sección 4 presentamos nuestras conclusiones.

2 Autómatas finidos deterministas: AFD

Una máquina de estado finito (autómata finito) es un modelo formal que permite representar el conjunto de estados por los que puede transitar un cómputo. Si además el autómata es un sistema determinista, entonces podemos afirmar que se trata de un autómata finito determinista (AFD).



	a	b
q0	q0	q1
q1	q0	q2
q2	q0	q2

Figura 1 Autómata finito determinista

Formalmente un AFD es una 5-tupla $A=(Q, \Sigma, q_0, \delta, F)$ [4] donde

- Q es un conjunto finito de estados.
- Σ es un alfabeto.
- $q_0 \in Q$ es el estado inicial.
- $\delta : Q \times \Sigma -> Q$ es una función de transición.
- $F \subseteq Q$ es un conjunto de estados finales o de aceptación.

Cuando un autómata lee un símbolo del alfabeto, considerando que se tiene un estado actual se produce una transición. Por ejemplo, $\delta(q_1, b)$ causa que el estado actual del autómata q_1, se modifique hacia q_2. Así pues, una cadena de entrada producirá un conjunto de transiciones. En la Figura 1 se ilustra un ejemplo de un AFD que reconoce como cadenas de éxito aquellas que terminan con "bb" sobre el alfabeto {a,b}. Podemos observar que el AFD se puede representar como una figura gráfica o bien, con una representación tabular indicando todas las combinaciones de transiciones

3 Diseño de diálogo de asistentes virtuales

En esta sección se revisan los aspectos de diseño del asistente, se introduce en particular, el método formal de modelado de diálogo que proponemos.

3.1 Fases de construcción de asistentes

La construcción de un asistente virtual se compone, a grandes rasgos, de los siguientes elementos:

1. Reconocimiento del habla y conversión de voz a texto.
2. Reconocimiento léxico, sintáctico y semántico de los elementos (tokens) de una oración.
3. Reconocimiento de frases de dialogo y su lugar dentro de un proceso de comunicación.
4. Acciones de respuesta automáticas (emulando el modelo de Agente Inteligente).

La primera parte, que tiene que ver con reconocimiento del habla ha sido sustancialmente resuelta de manera satisfactoria y existen diferentes APIs que permiten automatizar el reconocimiento de voz y su conversión a texto (Android Developers).

En cuanto al segundo elemento, para el reconocimiento léxico, sintáctico y semántico de una oración textual, es decir, para recoger el significado de una oración se han desarrollado y probado diversas técnicas formales, desde las clásicas: vectoriales, probabilísticas, estadísticas hasta las que hoy en día se consideran definitivas, es decir, las que emplean aprendizaje automático profundo [1].

Si bien podrían desarrollarse bibliotecas de código a partir de técnicas formales primitivas, también hay que decir que las herramientas industriales para creación de asistentes han hecho un gran esfuerzo que simplifica el trabajo. Así pues, el reconocimiento del significado de una oración se puede determinar mediante las capacidades de aprendizaje automático de los *frameworks* para desarrollo de asistentes. De hecho, Dialogflow de Google pone a disposición del usuario programador sus capacidades de *machine learning* para detectar adecuadamente el significado de una oración.

En este trabajo nos centramos en el tercer punto del desarrollo de asistentes, el que tiene que ver con la integración armónica de un conjunto de oraciones para conducir a un proceso de conversación con un asistente. Las aplicaciones de los asistentes virtuales conversacionales son muchas, entre otras podrían citarse: diálogos de venta de productos, orientación de funcionamiento de procesos, atención de centros de telefonía, etc.

3.2 Reconocimiento de fragmentos de diálogo del asistente

Para la fase de reconocimiento del diálogo de un asistente virtual proponemos los siguientes pasos:

- Requisitos: determinación de las oraciones que transmiten información u órdenes de la persona hacia el asistente.
- Determinación de respuestas de diálogo del asistente hacia el usuario.
- Integración del modelo de diálogo del asistente.
- Implementación.

La propuesta del presente artículo se centra específicamente en el uso de autómatas finitos para llevar a cabo la integración del modelo de diálogo del asistente.

3.3 Elementos del *framework*

A continuación, se describen dos elementos de Dialogflow que consideramos, son importantes para probar nuestra propuesta de modelado de diálogo:

1) *Intents*: Es el nombre técnico que Dialogflow da a las intenciones de comunicación, es decir, a las órdenes de un usuario hacia un asistente. Un *intent* es un modelo de orden en lenguaje natural que hay que construir. El *intent* puede embeber palabras clave (*keywords*) y sus variaciones léxicas, sinónimos y cantidades; en esta parte Dialogflow también explota las características de *machine learning* de Google para reconocer automáticamente variaciones de acepciones de palabras. Es posible detectar una oración incluso cuando se cambia de posición las palabras definidas en el modelo de la orden. El programador debe configurar a través del *framework* de Dialogflow cada *intent* introduciendo versiones alternas de la frase para que los algoritmos de aprendizaje automático de Google puedan ser entrenados y reconocer expresiones verbales de usuarios del mundo real.

2) *Contexts*: Los contextos son mecanismos que fungen como estados previos obligatorios para que se pueda reconocer un *intent* (llamados *context* de entrada) y, por otro lado, cuando un *intent* es ejecutado pude llevar a cabo la definición de un *context* de salida al terminar su tarea, que podría ser condición para que se activen otros *intents*, de esta manera un *context* puede verse como una estafeta que entrega un *intent* hacia otro *intent*.

3.4 Autómata finito determinista de diálogo

En esta sección adaptamos la definición de AFD para describir el Autómata Finito Determinista de Diálogo: AFDD como una herramienta formal para modelar el diálogo de asistentes cognitivos.

Definimos formalmente un AFDD como una 5-tupla $A_d = (Q_d, \Sigma_d, q_0, \delta_d, F_d)$ tal que:

- Q_d es un conjunto finito de estados. Cada estado representa un momento del diálogo entre un ser humano y un asistente. En términos de Dialogflow, un estado representa un punto lógico de una conversación al que se llegó derivado de una detección de un *intent* y al mismo tiempo representa una situación o punto de partida para continuar con la conversación.
- Σ_d es el universo de oraciones del asistente. Cada oración que va a ser capaz de detectar el asistente forma parte de Σ_d.
- $q_0 \in Q_d$ es el estado inicial, en el cual el asistente está alerta para recibir señales del usuario para iniciar la conversación.
- $\delta_d : Q_d \times \Sigma_d \rightarrow Q_d$ es una función de transición. Cuando se está en un estado y el asistente reconoce un *intent*, es decir, reconoce una expresión en lenguaje natural emitida por el usuario, entonces, se producirá el cambio hacia otro estado. En otras palabras, la función de transición se activa cuando el usuario expresa órdenes al asistente.
- $F_d \subseteq Q_d$ es un conjunto de estados finales o de aceptación. En términos del diálogo los estados finales serán aquellos que representan la conclusión de una operación, tal como la venta concluida de un producto.

3.5 Modelado de diálogo

Para ilustrar la forma de hacer el modelado de diálogo introducimos un caso de ejemplo: Se considera una persona que desea comprar hamburguesas, el asistente interactúa con la persona, primero ofreciendo producto, después el cliente hace su pedido. El asistente dá la oportunidad de reiterar. Una vez que se decide no reiterar el asistente pide datos de entrega, mismos que recibe y finaliza informando las características del pedido y los datos de entrega que recibió. En la Figura 2, se ilustra un diálogo real, se antecede la palabra bot a las afirmaciones que proceden del asistente.

humano: Hola, ¿qué tal?

bot: ¡Hola! ¡Hola! Hay burros y hamburguesas, ¿Qué te damos?
humano: Dame por favor tres hamburguesas

bot: Ya anotamos su petición de tres hamburguesas. ¿deseas algo más?
humano: Si por favor

bot: Muy, bien reitero que tenemos burros y hamburguesas, ¿qué te damos?
humano: Quiero cinco burguers

bot: Ya anotamos su petición de cinco burguers. ¿deseas algo más?
humano: No ya no

bot: De acuerdo, consideramos tu pedido como completo, ¿deseas confirmar envío?
humano: Si

bot: De acuerdo pedido confirmado, procedemos a solicitarte datos de envío, por favor indica tu nombre en el formato: Soy Pedro Hernández...
humano: José Ramos

bot: Digita el domicilio
humano: Avenida Tecnológico No 1500, Colonia lomas de santiaguito, Morelia, Michoacán, CP 58000

bot: Teclea el teléfono por favor
humano: 443 312 15 70

bot: Gracias por tu confianza, José Ramos, el pedido se enviará al domicilio Colonia lomas de santiaguito, Morelia, Michoacán, CP 58000, si hay algún detalle te marcaremos al 4433121570

Figura 2 Un diálogo para comprar 'burritos' y hamburguesas.

El dialogo puede ser comprobado de manera verbal, con micrófono o textual en la liga: https://bit.ly/3YYWuZC

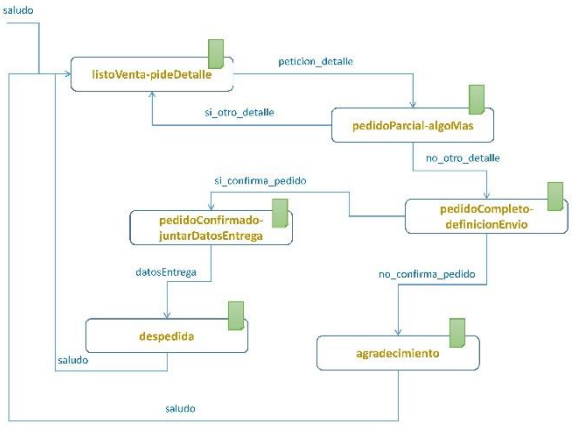


Figura 3 AFDD del diálogo de la Figura 2.

En nuestra propuesta el diálogo se representa por un AFDD. Cada intención de comunicación será modelada como una transición del autómata, mientras que los contextos de entrada y salida serán modelados como estados del autómata, veamos:

El *intent* "saludo" representa una intención de saludo del cliente hacia el asistente, pero al mismo tiempo espera que como respuesta el asistente indique su función y que inicie el proceso del diálogo. Al configurar el *intent* "saludo" debemos, por un lado, programar una respuesta automática, por ejemplo: "!Hola! Hay

burros y hamburguesas, ¿Qué te damos?" para que sea desplegada cuando el asistente reconozca un saludo, pero por otro, considerando el hecho de que ya reconoció el saludo, entonces debemos establecer un contexto de salida, en la ilustración "listoVenta-pideDetalle".

Al establecer el contexto de salida, cuando la persona hable y el asistente escuche, éste procurará reconocer los *intents* que tienen establecido el contexto de entrada "listoVenta-pideDetalle", esto es, el *intent* "peticion_detalle". De esta manera los estados van indicando qué parte del diálogo es susceptible de ser reconocida.

El nombre que damos en la ilustración a los estados tiene como fin marcar por un lado la conclusión de una acción de comunicación, pero por otro indicar el inicio de otra acción, por ejemplo, en la Figura 3: "pedidoParcial-algoMas" significa que se registró un pedido parcial, pero, al mismo tiempo el asistente estará esperando que el usuario pida algo más.

El diálogo que se muestra en la ilustración del autómata (Figura 3) es el que ha sido implementado en el caso de ejemplo. El AFDD en cuestión ilustra la segmentación del proceso de diálogo.

Enseguida, en la figura 4, se presenta una impresión de pantalla del asistente.

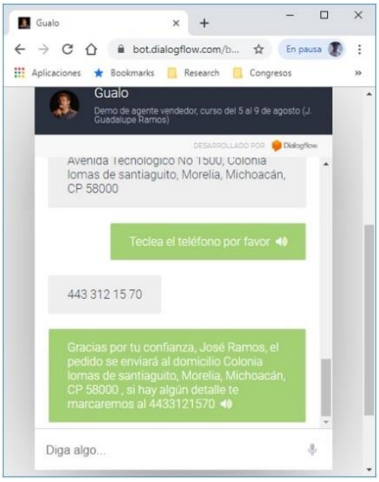


Figura 4 Captura de pantalla del asistente.

El usuario del asistente no puede ver el flujo del diálogo, únicamente puede observar el instante de interacción con la persona.

4 Conclusiones y trabajo futuro

Creemos que la forma que se introdujo de abordaje de diálogos complejos puede ser útil para llevar a cabo la tarea de diseño de asistentes conversacionales, si bien, ya hay muchos *frameworks* para realizar asistentes, pensamos que el diálogo, por sí mismo, debe tener su propia estrategia de creación, que permita explicar y desarrollar estas herramientas. En este sentido, la adaptación que hemos hecho de autómatas finitos deterministas para representar diálogos mediante AFDDs, como documento de ingeniería de producto, sirve para poder tener una interacción con el inversionista que contrató el desarrollo del asistente, durante el proceso de diseño y desarrollo, ya que materializa de manera gráfica la conversación que se desea automatizar.

Como trabajo futuro consideramos realizar ejemplos que permitan ilustrar el formalismo AFDD como parte de una metodología de diseño de diálogo. Por otro lado, creemos que se puede construir una herramienta visual [2] para dibujar los autómatas y que a partir de ella se imprima la especificación del diálogo que a su vez podría ser empleada como insumo para crear el asistente en un *framework* industrial como aquí se ilustró con Dialogflow. La idea de una herramienta visual genérica, independiente de un *framework*, es una necesidad real, por ejemplo, la versión avanzada de [3] posee un entorno de diseño basado en máquinas de estado, pero, hasta la fecha en que se escribió este trabajo, no se puede acceder sin ser usuario de la versión CX.

5 Agradecimientos

Este documento es fruto del proyecto con registro del Tecnológico Nacional de México: 14208.22-P "Modelo de software para consumo masivo de datos abiertos mediante bots y web semántica". Agradecemos el apoyo prestado por el TecNM.

6 Referencias

[1] Bhagwat, V. A. Deep learning for chatbots, 2018, San Jose State University. https://doi.org/10.31979/etd.9hrt-u93z

[2] Caprio, G. *Domain-specific languages & DSL workbench*, Dr. Dobbs, 2006, http://www.ddj.com/windows/184429825

[3] Google, *Dialogflow: Build natural and rich conversational experiences*, 2020, https://dialogflow.com/

[4] Kelley, D. Teoría de autómatas y lenguajes formales. *Pearson Educación*, 1995

[5] Maier, T., Menold, J. and McComb, C. Towards an ontology of cognitive assistants, *Proceedings of the Design Society: International Conference on Engineering Design*, vol. 1, no. 1, pp. 2637–2646, 2019. https://doi.org/10.1017/dsi.2019.270

[6] NSF, Ed., Intelligent Cognitive Assistants Workshops, Workshop Summary and Recommendations, ICA-1 Report 2016, May 8-10, 2016, Washington, D.C., USA, ser. Intelligent Cognitive Assistants Workshops, vol. 1. Joint Semiconductor Research Corporation and NSF, 2016.