Herramientas para la Computación en la Nube Dirigida a Inteligencia Artificial

# Tema 7. Diseño de bots conversacionales con IA en la nube

# Índice

Ideas clave

7.1. Introducción y objetivos

7.2. Diseño de bots conversacionales con Microsoft Azure AI Bot Services

7.3. Proceso de configuración y creación de mensajes para bots conversacionales

7.4. Integración de bots conversacionales

7.5. Beneficios para el negocio y casos de uso de bots conversacionales

7.6. Cuaderno de ejercicios

7.7. Referencias bibliográficas

A fondo

Bot Framework Composer

Introduction to Microsoft Copilot Studio

Test

## 7.1. Introducción y objetivos

U n **chatbot** constituye una interfaz de usuario basada en conversaciones, cuyo mayor beneficio es que presentan una curva de aprendizaje prácticamente nula para quien los utiliza. Se utilizan en diferentes ámbitos de negocio para captar clientes, proporcionarles servicios de soporte, mejorar la eficiencia de los empleados o, de forma más general, para proporcionar información y automatizar actividades.

En este tema, se presentan dos plataformas de creación de chatbots a través de servicios de computación en la nube: **Azure Bot Framework** (o Azure AI Bot Service) y **Microsoft Copilot Studio** (previamente conocida como Power Virtual Agents). La primera es la plataforma ***legacy,*** creada con anterioridad a la irrupción de la IA generativa; no obstante, ha ido incorporando progresivamente capacidades avanzadas de comprensión y generación de lenguaje natural. La segunda ha sido desarrollada recientemente, e incorpora el paradigma de **IA generativa** en su totalidad. Ambas tienen nichos de aplicación bien definidos y son relevantes en el mercado.

Para cada una de ellas se describen sus características y los conceptos en torno a los cuales se articulan sus funcionalidades. También se incluyen ejemplos de creación de chatbots para ambas, así como los procedimientos que permiten desplegarlos en recursos de computación en la nube. Asimismo, se describen las ventajas de utilización de **chatbots** y los casos de uso más populares en cada sector.

# Ideas clave

Por último, este tema se plantea los siguientes objetivos para el alumno:

- Diseñar el *business case* que justifica la inversión en tecnología en la nube para la creación de chatbots.
- Ser capaces de diseñar y construir chatbots con Azure Bot Framework Composer y Microsoft Copilot Studio.
- Disponer de criterio suficiente para determinar en qué casos es más recomendable un chatbot determinista, basado en reglas, frente a uno basado en IA generativa, o viceversa.

## 7.2. Diseño de bots conversacionales con Microsoft Azure AI Bot Services

**Introducción**

Microsoft proporciona dos entornos para la creación de bots: Bot Framework Composer y Microsoft Copilot Studio (que sustituye a Power Virtual Agents). A continuación, se describen las características de cada uno de ellos.

**Bot Framework Composer**

Bot Framework Composer es un **entorno de desarrollo (IDE)** ***open-source***, que permite crear, probar y publicar chatbots. Está disponible para los principales sistemas operativos y puede descargarse desde la web de Microsoft.

---

Para descargar Bot Framework Composer, ingresa al siguiente artículo: JonathanFingold, emgrol, stevemunk, ianl-terawe, erhopf, Kaiwb y zxyanliu (2022, octubre 6). Install Bot Framework Composer. *Microsoft Learn.* https://learn.microsoft.com/es-es/composer/install-composer?tabs=macos

---

El IDE proporciona un **lienzo** o canvas en el que pueden diseñarse los principales elementos de la experiencia, como los flujos de la conversación o el cambio de contexto. Otras tareas más complejas, como la integración con servicios vía REST, deben realizarse a través de código fuente, con el SDK de Bot Framework.

En la siguiente Figura 1, se muestra el aspecto de la interfaz de usuario de Composer:

# Ideas clave

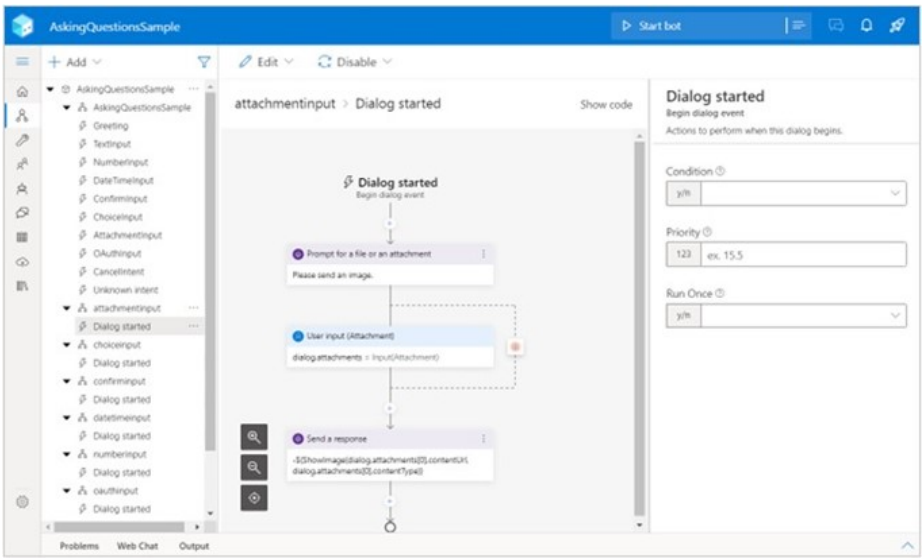



Figura 1. Canvas de Bot Framework Composer. Fuente: JonathanFingold, iaanw, eric-urban, emgrol, erhopf, stevemunk, WashingtonKayaker, Quirinevwm, zxyanliu, benbrown, 2022.

A través de Composer, pueden llevarse a cabo las siguientes tareas del proceso de generación de chatbots:

- Crear un chatbot a partir de una **plantilla.**
- Generar los **contenidos** del chatbot, incluyendo variaciones de estilo, en diferentes idiomas.
- Incorporar habilidades de comprensión de **lenguaje natural.**
- Incorporar la capacidad para responder a preguntas **(FAQ).**
- Integrar el chatbot con **servicios externos.**
- Probar el **funcionamiento** del chatbot.
- Publicar el chatbot a través de **Azure App Service o Azure Functions.**

Para llevar a cabo las tareas anteriores, es importante comprender las características de Composer que se describen a continuación.

### Plantillas

Las plantillas de Composer simplifican la puesta en marcha del proceso de creación de un chatbot a partir de plantillas *(templates)* que cubren los escenarios más habituales. Es posible añadir, eliminar o personalizar las capacidades incluidas en estos ejemplos. En la Tabla 1 se incluye la lista de plantillas disponibles:

| Plantilla | Descripción |
|---|---|
| **Empty bot** | Plantilla básica con un saludo y un sistema para capturar las preguntas que no sabe contestar. |
| **Core bot with Language** | Añade a la anterior la posibilidad pedir ayuda o finalizar la conversación. Incorpora elementos básicos de comprensión de lenguaje natural. |
| **Core bot with QnA Maker** | Introduce la capacidad de responder a preguntas incluidas en una base de conocimiento mediante QnA Maker (QnA Maker será discontinuado en 2025). |
| **Core Assistant Bot** | Añade un *tour* de funcionalidades, detección de la intención del usuario, recepción de *feedback*, detección de errores y posibilidad de charlar de manera informal (*small talk*). |
| **Enterprise Assistant Bot** | Incluye las habilidades de los bots Calendar y People. |
| **Enterprise Calendar Bot** | Bot especializado en el manejo de la agenda del usuario a través de Microsoft 365, incluyendo la posibilidad de agendar reuniones en *slots* disponibles para todos los asistentes. |
| **Enterprise People Bot** | Bot especializado en la búsqueda de personas en una organización a través de Microsoft Entra ID. |

Tabla 1. Plantillas de bots en Composer. Fuente: elaboración propia.

### Diálogos

Un bot implementa uno o varios diálogos. Cada diálogo representa una funcionalidad del bot y define cómo reaccionará a las peticiones del usuario. Por tanto, incluye el texto de respuesta, así como lógica de negocio y llamadas a APIs necesarias para generarlo.

# Ideas clave

Existen dos **tipos** de diálogos:

- **El diálogo principal,** que es el que inicia la conversación con el usuario.
- **Los diálogos secundarios,** o *child* (hijo), que están vinculados al diálogo principal y se especializan en alguna de las funcionalidades del bot.

Cada diálogo tiene los siguientes **elementos:**

- ***Recognizers.*** Identifican el significado *(intent)* del mensaje del usuario y extraen sus principales entidades.
- ***Triggers.*** Son llamados cuando sucede algún evento relevante en la conversación. Por ejemplo, detección del significado de un mensaje. Se encargan de manejar dicho evento para decidir qué hacer a continuación.
- ***Actions.*** Son las acciones de respuesta desencadenadas por un *trigger.* Por ejemplo: iniciar la escritura de una respuesta, hacer un cálculo o consultar una base de conocimiento.
- **Generadores de lenguaje.** Cuando la acción de respuesta consiste en escribir un mensaje, se encargan de componer dicho mensaje a partir de plantillas y variables.

**Memoria**

La memoria se refiere a la información intercambiada en el histórico de la conversación, necesaria para que el bot disponga de contexto suficiente para continuarla. Esta se implementa a través de **propiedades,** que tienen un nombre, un alcance y un valor. El alcance especifica cuándo está disponible y durante cuánto tiempo debe guardarse.

### Habilidades

Una habilidad o *skill* es un bot que no interactúa directamente con el usuario, sino que proporciona una **funcionalidad** determinada **a otros bots.** Esta funcionalidad se describe en fichero JSON, denominado *manifest,* que también incluye la especificación de entradas y salidas.

### Generación de lenguaje

Para la generación de respuestas en diferentes puntos de la conversación, se utilizan **plantillas parametrizables** (no deben confundirse con las plantillas de bots descritas previamente). A continuación, se incluye una plantilla de ejemplo:

```
> this is a comment

# nameTemplate

- Hello ${user.name}, how are you?

- Good morning ${user.name}. It's nice to see you again.

- Good day ${user.name}. What can I do for you today?
```

### Comprensión de lenguaje

Para la **comprensión del lenguaje,** se utilizan tres componentes:

- ***Utterances,*** que son ejemplos de frases que puede utilizar el usuario.
- ***Intents,*** que son los significados o finalidad asociada a esas frases.
- ***Entities,*** que son los «parámetros» asociados a un *intent*, por ejemplo, lugares, personas o fechas.

A partir del *intent* y sus *entities,* el bot debe ser capaz de generar una respuesta útil para el usuario.

**Experiencia conversacional de usuario (CUX)**

Se refiere una **colección de consejos y mejores prácticas** que ha elaborado Microsoft para mejorar la experiencia de usuario con el uso de bots. A continuación, se incluyen algunas de ellas:

- **Planificación del bot.** Cuál es la finalidad, qué problemas resuelve, cuál es el perfil de usuario o en qué plataformas se ejecutará el bot.
- **Personalidad del bot.** El hecho de que el bot muestre algún tipo de personalidad mejora la experiencia del usuario.
- Mantener los **diálogos breves.**
- Incluir un diálogo para **pedir ayuda** o **finalizar la conversación.**
- Añadir **variaciones** a las respuestas.
- Anticipar **ambigüedades** en los mensajes del usuario y manejarlos conforme a los escenarios que el bot debe resolver.

**Extensiones**

Las extensiones en Composer son **módulos Javascript** que permiten **personalizar** el funcionamiento de varios aspectos del entorno, dotándole a los bots de funcionalidades no disponibles por defecto. Por ejemplo: almacenar contenidos en una base de datos, requerir autenticación del usuario, conceder acceso al bot en función del rol del usuario o publicar contenido en servicios externos.

**Conexiones**

Las conexiones se utilizan para conectar los bots con **servicios de mensajería** de Azure o externos. En el caso de Azure, están disponibles los siguientes: Web Chat (el servicio por defecto), Microsoft Teams o Direct Line Speech, que sustituye la interfaz de texto por una de voz, en la que el usuario puede hablar al bot y escuchar

sus respuestas. También están soportados muchos servicios populares de mensajería, como Twilio, Slack o Facebook.

**Generación de audio**

Los bots de Composer pueden responder mediante **texto** o mediante **voz.** En el segundo caso, el desarrollador debe definir un **fichero SMML** *(speech synthesis markup language),* que especifica aspectos como el tono, la pronunciación o la velocidad del habla.

**Orquestación**

El orquestador de Bot Framework utiliza modelos preentrenados para facilitar la comprensión de los ***intents* del usuario,** es decir, cuál es el significado de su mensaje y qué es lo que espera como respuesta.

A partir de un conjunto reducido de peticiones y respuestas ***(few shot learning),*** el orquestador es capaz de interpretar correctamente peticiones similares del usuario y asociarlas con la respuesta correspondiente. Adicionalmente, el orquestador puede extender el **soporte a idiomas diferentes** a los utilizados en los ejemplos y también reconocer múltiples *intent*s en una petición de usuario.

El procedimiento para **configurar el orquestador** comprende los siguientes pasos:

- **1.** Crear ejemplos de *utterances* con la etiqueta de *intent* asociada en un fichero con extensión .lu .
- **2.** Descargar el modelo preentrenado que se utilizará como base para incorporar los ejemplos anteriores:

```
bf orchestrator:basemodel:get --out ./models
```

- **3.** Combinar el modelo con el fichero .lu para generar un fichero .blu (denominado snapshot ):

```
bf orchestrator:create --model <base model folder> --in <label file/folder>
--out <generated folder> --hierarchical
```

- **4.** Probar la precisión del snapshot en la identificación de *intents:*

```
bf orchestrator:test --in <snapshot file> --model <base model file> --out
<report folder> --test <test data file>
```

- **5.** Integrar el recognizer resultante en el bot.

**Paquetes**

Bot Framework soporta la creación e instalación de paquetes que encapsulan **nuevas funcionalidades** para bots, tales como: acciones, *triggers,* diálogos, mecanismos de comprensión del lenguaje y ficheros para generación de respuestas.

Existen **dos tipos** de paquetes:

- **NuGet,** para bots escritos en lenguaje C#/.NET .
- **npm,** para bots escritos en Node.js o Javascript.

Los paquetes pueden ser consumidos a través de un *feed* público o privado, o directamente instalados desde el sistema de ficheros de un PC.

En la Figura 2 se muestra cómo se integran los paquetes en la arquitectura de Bot Framework:

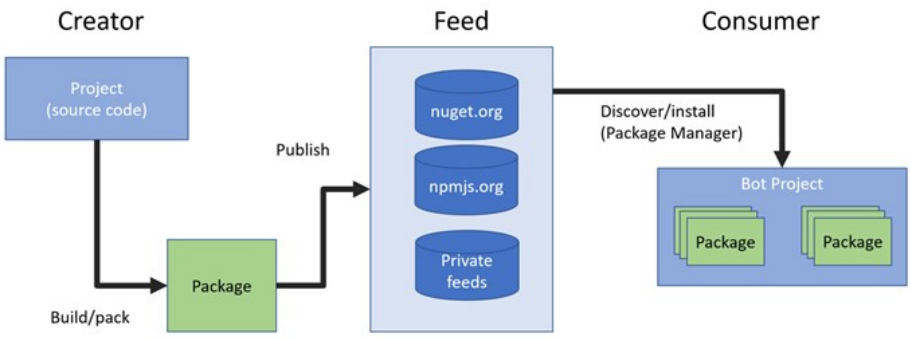



Figura 2. Ejemplos de paquetes e integración en la arquitectura de un chatbot. Fuente: JonathanFingold, iaanw, eric-urban, emgrol, erhopf, stevemunk, WashingtonKayaker, Quirinevwm, zxyanliu, benbrown, 2022.

***Hand-offs***

Los bots creados en Composer pueden **transferir** la conversación a un **experto humano** cuando no sean capaces de ofrecer respuestas al usuario.

La configuración del *hand-off* requiere de dos pasos: identificar en qué punto de la conversación debe realizarse el *hand-off* y disponer del paquete (ver sección anterior) que permite efectuar el escalado a una plataforma externa. Los paquetes de *hand- off* disponibles son **LivePerson** y **ServiceNow.**

**Código fuente**

La funcionalidad no disponible por defecto en Composer puede ser incorporada a través de código fuente. Para ello, pueden utilizarse los lenguajes de programación C# o Javascript. La nueva funcionalidad se denomina ***bot component.*** Para posibilitar el desarrollo, Bot Framework proporciona el Adaptive Runtime, que debe incorporarse como dependencia al proyecto. Este *runtime* es necesario para acceder a la SDK.

**Microsoft Copilot Studio**

Un *copilot* es una interfaz conversacional basada en IA y, concretamente, en *large language models* (LLMs). Copilot Studio es la herramienta proporciona por Microsoft para desarrollar *copilots.*

---

Copilot Studio puede utilizarse desde el sitio web https://copilotstudio.microsoft.com o desde Microsoft Teams.

---

En la Figura 3 se muestra una captura de Microsoft Copilot Studio para la creación de chatbots:

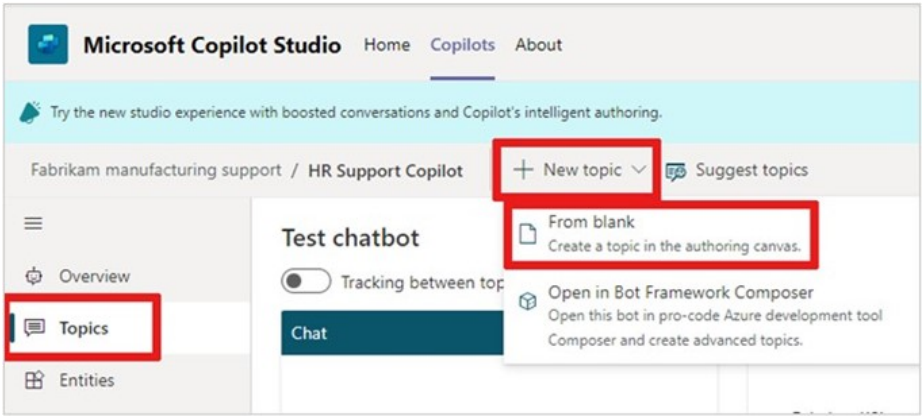



Figura 3. Pantalla de creación de bots en Microsoft Copilot Studio. Fuente: Iaanw, sad-cl0wn, bolducs, GitikaG, peterswimm, svielse, 2024b.

En las siguientes secciones se describen los aspectos más relevantes de este entorno, como paso previo a la implementación práctica de bots.

**Diferencias con Bot Framework Composer**

En Copilot Studio, los conceptos de *trigger* y *dialog* vistos para Composer se combinan en un único concepto, denominado ***topic.*** Cada *topic* es un flujo o ruta posible de la conversación, que se activa a partir de determinadas frases del usuario que actúan como *trigger.* Por otro lado, las *actions* vistas en Composer se denominan **nodos** *(nodes)* en Copilot Studio.

**Comprensión de lenguaje natural**

La comprensión del lenguaje natural en los bots de Copilot Studio se realiza con los **modelos GPT** de OpenAI. De esta forma, el bot puede comprender la petición, utilizar el conocimiento del que dispone para generar una respuesta (o identificar la fuente en la que puede encontrar la respuesta y obtenerla sobre la marcha), y traducir dicha respuesta a lenguaje natural para presentársela al usuario.

**Creación de *topics***

La labor de creación de un bot en Copilot Studio se centra en la creación de *topics.* Cuando el usuario realiza una petición de algo que no está ninguno de los *topics:*

- El bot le pide al usuario que **reformule** la petición.
- Si aun así no es capaz de gestionarla, lanza un proceso de **escalado** a través del *topic* Escalate para que la conversación sea atendida por un humano.
- Como alternativa, es posible especificar un ***fallback topic,*** es decir, una respuesta por defecto.

Por otro lado, la creación de los *topics* puede automatizarse a través de la funcionalidad «Create with Copilot». En este caso, el creador del bot solo debe describir la temática asociado al *topic,* y Copilot Studio se encarga de generar dicho *topic,* incluyendo el contenido necesario, los nodos (que realizan preguntas, generan respuestas o introducen lógica condicional) y la interrelación entre ellos.

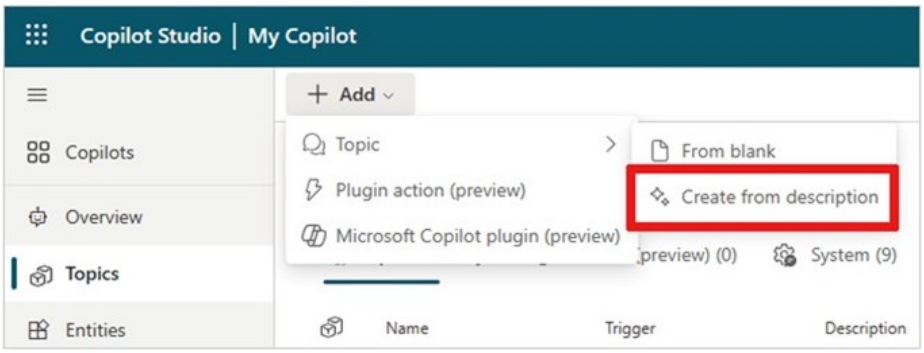



Figura 4. Creación de *topics* mediante «Create with copilot». Fuente: KendalBond007, cyrilanderson, sad-cl0wn, 2024.

**Respuestas generativas**

La opción **«Boost conversations»** permite reaccionar a escenarios en los que ningún topic cubre la necesidad del usuario. En ese caso, se especifica una fuente externa, y el bot trata de generar una respuesta apropiada a partir de los contenidos encontrados en dicha fuente, de forma dinámica. Los tipos de fuente soportados son:

- Un **sitio web** accesible a través de una URL.
- Una ruta a **OneDrive, Sharepoint o Dataverse.**
- **Bing Search**, el buscador de Internet de Microsoft.

En la Figura 5 se muestra el cuadro de diálogo en el que se puede configurar un sitio web externo:

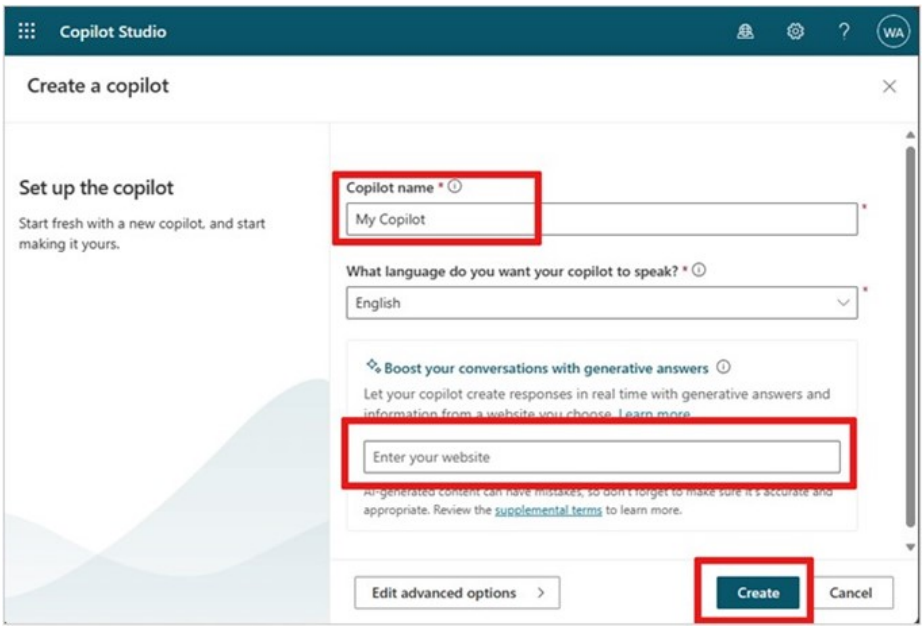



Figura 5. Configuración del bot para extraer información de una fuente externa. Fuente: KendalBond007, mainguy70, bolducs, raemone, cyrilanderson, Eric-tw-convAI, sophie-roy3, 2024.

Para lograr esta funcionalidad, se utiliza el concepto de **respuestas generativas.** Mediante IA generativa, el bot es capaz de entender el contenido de la fuente y realizar un resumen de los fragmentos relevantes para proporcionarlo como respuesta al usuario, en lenguaje natural.

Para acotar el alcance de las respuestas proporcionadas, se puede utilizar la opción de «Content moderation». Cuando se fija el valor a «High», el bot solo proporciona respuestas precisas. En los valores «Medium» y «Low», el bot es capaz de contestar a más preguntas, pero las respuestas podrían ser inexactas. También se pueden especificar instrucciones al bot para controlar su comportamiento. En la Tabla 2 se incluyen algunos ejemplos:

| Categoría | Instrucción |
|---|---|
| Alcance | Si el usuario te pregunta algo que no sabes, simplemente di que no lo sabes. |
| | Si alguna de las preguntas alude a nuestra competencia (marca A, marca B, marca C), responde con los productos y servicios de nuestra empresa, obviando la referencia a estas marcas. |
| Formato | Estructura la respuesta en una serie de pasos. |
| | Proporciona ejemplos útiles con la respuesta. |
| Tono | Tu tono debe ser amistoso, cordial e informal. |
| | Saluda siempre al usuario. |
| | Utiliza emojis y símbolos de exclamación cuando corresponda. |
| | Termina las conversaciones con un mensaje positivo. |
| | Ofrece tu ayuda de forma proactiva. |

Tabla 2. Ejemplos de instrucciones para configurar el comportamiento del bot. Fuente: elaboración propia.

### Creación de *plugins*

Las funcionalidades por defecto que ofrece Copilot Studio pueden ampliarse a través de *plugins.* Estos pueden combinarse para soportar una tarea completa, que va más allá de proporcionar una respuesta. Por ejemplo, el equipo comercial puede utilizar un copilot para automatizar la creación de nuevas oportunidades. Para ello, se integran los siguientes *plugins:*

- Un *plugin* que dispara un **flujo de Power Automate** —herramienta de creación de automatizaciones de Microsoft— para crear el *lead* en la plataforma corporativa de gestión de *leads* (Salesforce).
- Otro *plugin* resume las conversaciones que ha tenido el usuario con el contacto asociado al *lead,* y las incorpora al registro del *lead* en Salesforce.

Existen dos tipos de *plugins:* **conversacionales,** que son similares a los *topics* de un *copilot* convencional, y los **A I *plugins,*** que pueden pertenecer a una de las siguientes categorías:

| Categoría | Instrucción |
|---|---|
| ***Prompts* de AI Builder** | Permite a los usuarios generar *prompts* en lenguaje natural para desencadenar acciones, entre ellas la generación de respuestas. |
| **Flujos de Power Automate** | Permite activar flujos (*flows*) de Power Automate para recuperar información de los entornos a los que tiene acceso el usuario, o bien lanzar una acción determinada. |
| **Conectores de Power Platform** | Se utilizan conectores de Power Platform para acceder a fuentes externas mediante sus respectivas APIs. |
| **Acciones de OpenAI** | Es la versión de los *prompts* de AI Builder creada por OpenAI. |

Tabla 3. AI *plugins.* Fuente: elaboración propia.

En la actualidad existen más de 1200 conectores de Power Platform, cuya lista puede consultarse en: https://learn.microsoft.com/en-us/connectors/connector-reference/

**Protección frente a fugas de información**

Los *copilots* pueden introducir un **riesgo** en la organización, dado que tienen la posibilidad de acceder a múltiples fuentes de información que pueden incluir datos sensibles. Para minimizar este riesgo, se proporcionan los mecanismos de ***data loss prevention* (DLP).** Estas políticas pueden aplicarse a los *copilots* para evitar que los desarrolladores activen determinados canales (Teams, Facebook, Omnichannel o Direct line) o bien que la autenticación del usuario sea obligatoria para usar el *copilot.*

## 7.3. Proceso de configuración y creación de mensajes para bots conversacionales

**Bot Framework Composer**

En el siguiente ejemplo se crea un bot que proporciona información meteorológica. El usuario puede solicitar esta información para un código postal, y el bot extrae los datos relevantes de un servicio externo. Para resolver este escenario, se ponen en práctica varios de los conceptos descritos en secciones anteriores.

**Paso 1: inicialización del bot**

Los pasos que deben seguirse son los siguientes:

- **1.** Abrir la aplicación de Composer y hacer clic en «Create new».
- **2.** Seleccionar como lenguaje de programación « C #» y la plantilla del Empty bot —cuya descripción aparece en la lista de plantillas de la sección anterior—.
- **3.** Rellenar los datos básicos del bot: nombre, entorno en el que se ejecutará (Azure Web App) y la carpeta destino de los recursos generados durante el proceso. Hacer clic en «Create» para crear el proyecto.
- **4.** Por defecto se crean dos *triggers:* «Greeting», para saludar al usuario cuando accede al bot, y «Unknown intent», cuando ninguno de los flujos de conversación encaja con la petición del usuario.

**Paso 2: procesamiento de mensajes del usuario**

Los pasos que deben seguirse son los siguientes:

- **1.** Seleccionar el *trigger* «Greeting». Se muestra el canvas con un diagrama de bloques. Hacer clic en el bloque «Send a response» y sustituir el texto de la sección «Responses» por un texto de saludo personalizado.

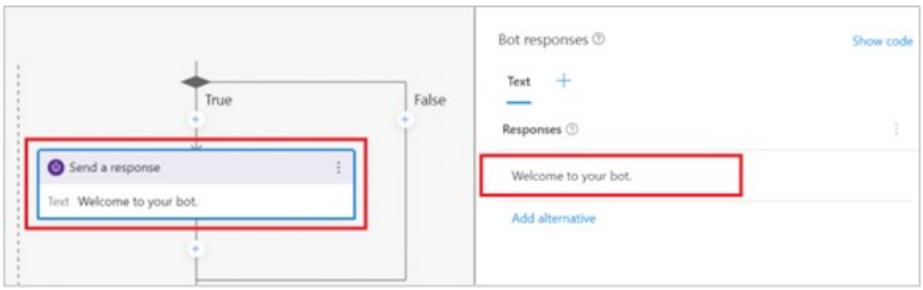



Figura 6. Modificación del *trigger* Greeting. Fuente: JonathanFingold, damabe, emgrol, v-alarioza, stevemunk, MarcFromTerawe, 2023.

- **2.** Crear un nuevo *dialo*g para encapsular la funcionalidad de recuperación de los datos meteorológicos. Para ello, seleccionar el nombre del bot y hacer clic en «Add a dialog». Proporcionar un nombre y descripción.

- **3.** Seleccionar el *trigger* «BeginDialog», hacer clic en el símbolo «+» del canvas y seleccionar «Send a response». En el campo de texto, escribir «Vamos a buscar la información del tiempo». Esta es la información que se muestra al usuario cuando se activa este diálogo.

- **4.** A continuación, seleccionar de nuevo el nombre del bot, seleccionar «Recognizer/Dispatch type» y hacer clic en «Change». Seleccionar «Regular expression» en la lista de opciones. En efecto, vamos a utilizar expresiones regular para detectar el propósito *(intent)* del usuario.

- **5.** Seleccionar el nombre del bot y hacer clic en «Add new trigger». Especificar «Intent recognized» como tipo, un nombre para el *trigger* y la expresión regular («tiempo»). De esta forma, cuando el usuario escriba un mensaje que contenga la palabra «tiempo», se activará el *trigger.*

- **6.** Por último, seleccionar el *trigger* y hacer clic en el símbolo «+», y a continuación en «Dialog Management – Begin a new dialog». Seleccionar el diálogo creado en el paso 2 de la lista «Dialog name». De esta forma, la activación de *trigger* invoca el inicio del diálogo.

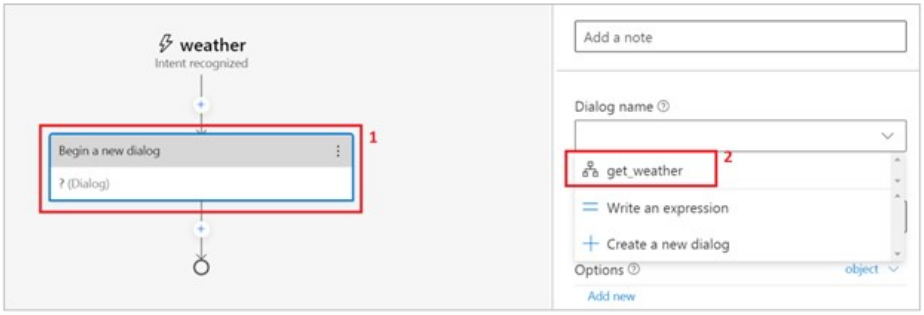



Figura 7. Vinculación de un trigger con un *dialog* en Bot Framework. Fuente: JonathanFingold, damabe, emgrol, v-alarioza, stevemunk, MarcFromTerawe, 2023.

**Paso 3: generación de respuesta**

Los pasos que deben seguirse son los siguientes:

- **1.** Seleccionar «Begin dialog» del diálogo creado en el paso anterior, hacer clic en «+» en el canvas y elegir «Ask a question – Text». En la sección «Bot response» de las propiedades, hacer clic en «Add alternative» y escribir el texto «¿Cuál es tu código postal?» en la sección «Text». En este paso, recogemos el código postal que permite determinar la ubicación sobre la que se desea conocer la información meteorológica.

- 2. Seleccionar «User input» e introducir el valor `user.codigo_postal` en Property. En «Output format» utilizar la expresión `=trim(this.value)` para eliminar los espacios que pueda haber delante o detrás del código postal.

- **3.** Seleccionar nuevamente «Begin dialog» del diálogo creado en el paso anterior, hacer clic en «+» y elegir «Access external resources – Send an HTTP request». Rellenar el campo URL con el siguiente valor: https://api.openweathermap.org/data/2.5/weather?zip=${user.postalcode},us&appid=YOUR_API_KEY, reemplazando el valor de la API key por el obtenido durante el registro en Weather API de OpenWeather.

En este paso, construimos la llamada HTTP al servicio externo que proporciona las predicciones meteorológicas, especificando la URL e incluyendo como parámetro el código postal solicitado previamente al usuario.

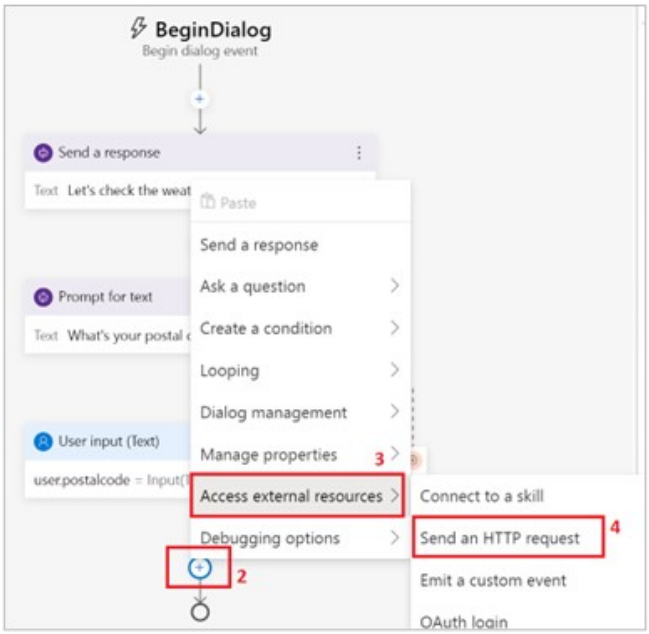



Figura 8. Creación de una HTTP request para recuperar información de una fuente externa. Fuente: JonathanFingold, damabe, emgrol, v-alarioza, stevemunk, MarcFromTerawe, 2023.

- **4.** Escribir el valor dialog.api_response en «Result property» para almacenar en esta variable el resultado de la petición HTTP. Fijar el valor JSON para el «Result type».

- **5.** En el canvas, hacer clic en «+» después de «Send an HTTP request» y seleccionar «Manage properties – Set properties». Crear una nueva propiedad, dialog.temp , con el valor =dialog.api_response.content.main.temp . Ahora disponemos del valor de temperatura para incluir en la respuesta.

- **6.** En el canvas, hacer clic en «+» después de «Set properties» y seleccionar «Send a response». En el panel de propiedades, en «Responses», hacer clic en «Add alternative» y escribir el texto « La temperatura es de {dialog.temp} grados ».

Una vez completados estos pasos, el bot incorpora la funcionalidad deseada. En la sección 7.4 se describe el procedimiento para ponerlo en producción.

**Microsoft Copilot Studio**

A continuación, se describen los pasos a seguir para crear un *copilot* en Copilot Studio. Se proporcionan varios ejemplos de uso del *copilot,* si bien la capacidad que ofrece Copilot Studio de generar *topics* automáticamente mediante una descripción permite al alumno experimentar con escenarios propios.

Los **pasos** son los siguientes:

- **1.** Entrar en el sitio web de Microsoft Copilot Studio () y hacer clic en «Create a copilot». Especificar el nombre del *copilot* y la URL del sitio web de *fallback,* es decir, aquella a la que puede acudir el *copilot* cuando desconozca la respuesta a la pregunta del usuario.

- **2.** Una vez creado el *copilot,* hacer clic en «Topics». A continuación, hacer clic en «Add – Topic – Create from description». Proporcionar un nombre para el nuevo *topic.*

- **3.** En el cuadro «Create a topic to…», describir la funcionalidad que debe proporcionar el *topic.* A continuación, se incluyen algunos ejemplos sencillos:

  - Pedir el nombre, edad y fecha de nacimiento al usuario, y devolver esta información como respuesta.

  - Pedir una hamburguesa a domicilio, eligiendo los ingredientes, la bebida y el acompañamiento (patatas fritas o ensalada). En este caso, el sitio web de *fallback* puede utilizarse para localizar las opciones disponibles en el menú.

  - Pedir al usuario su dirección postal, incluyendo el número de la calle, el código postal y la ciudad.

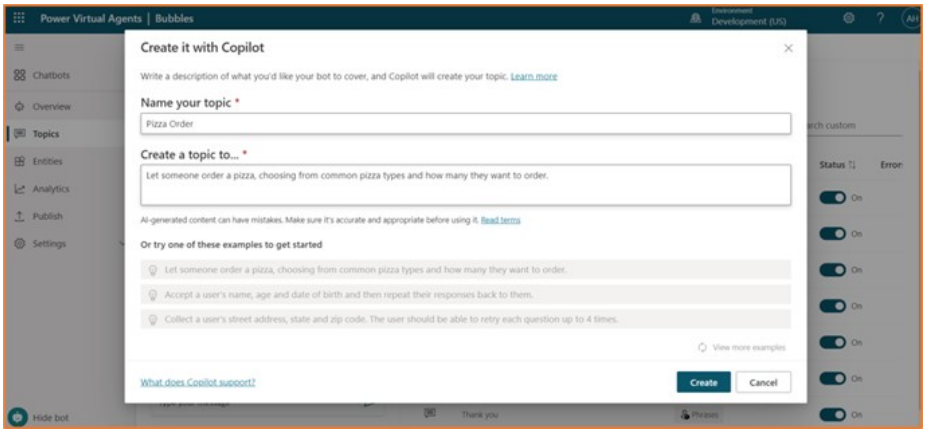



Figura 9. Cuadro de diálogo «Create it with Copilot» para crear un nuevo *topic*. Fuente: Elijah, 2023.

Una vez completados los pasos anteriores, el *copilot* ha sido creado con éxito. El siguiente paso es probarlo. Para ello, hay que seleccionar el *topic* recién creado y hacer clic en «Test copilot».

En el cuadro «Type your message» se pueden escribir los *prompts* del usuario. Es habitual realizar las siguientes categorías de pruebas:

- Un mensaje relacionado con la funcionalidad que proporciona el *topic.*
- Un mensaje que no se puede resolver a través del *topic,* pero sí rescatando información del sitio web de *fallback.*
- Un mensaje que no puede ser resuelto por ninguno de los dos métodos anteriores para verificar que el *copilot* responde, indicando que no puede ayudar al usuario con su petición.

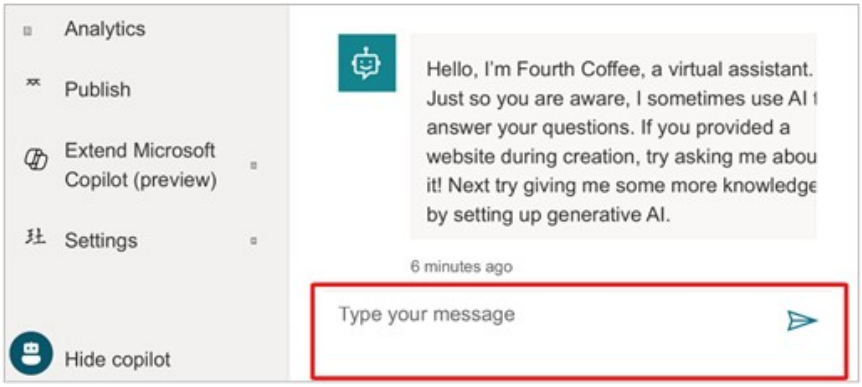
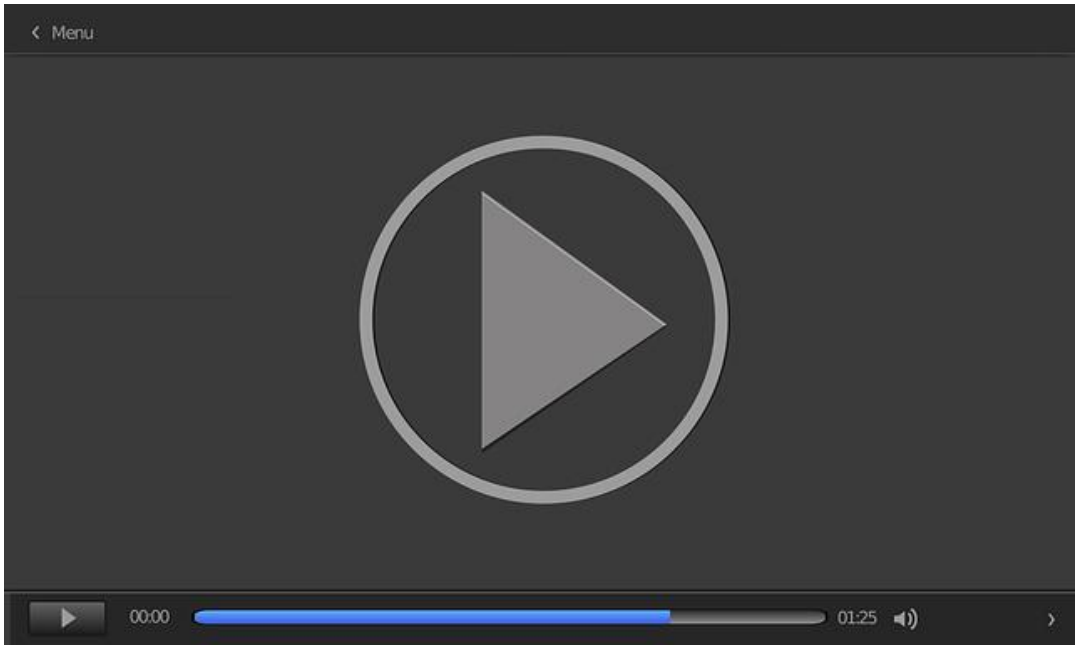



Figura 10. Pantalla de pruebas del *copilot* en Copilot Studio. Fuente: Iaanw, sad-cl0wn, bolducs, peterswimm, 2024b.

Los chatbots basados en IA generativa son susceptibles de ser mejorados a través de técnicas de ingeniería de *prompts.*

En el siguiente vídeo, *Personalización del comportamiento de un chatbot mediante ingeniería de prompts en OpenAI,* se muestran ejemplos de algunas de estas técnicas.

# Ideas clave




Accede al vídeo:

https://unir.cloud.panopto.eu/Panopto/Pages/Embed.aspx?id=1ea602a4-0556-4ce3-b60b-b16501226a60

## 7.4. Integración de bots conversacionales

**Bot Framework Composer**

La publicación de un chatbot creado en Composer puede hacerse a través de una Azure Web App. De esta forma, Azure provisiona de forma automática la infraestructura necesaria para hospedar el chatbot.

Los **pasos** que deben seguirse son los siguientes:

- **1.** En Composer, hacer clic en «Publish». Seleccionar el bot de la lista.
- **2.** En la sección «Publish target», seleccionar «Manage profiles». En «Publishing profile», hacer clic en «Add new». A continuación, escribir un nombre para el perfil de publicación y seleccionar «Publish bot to Azure».
- **3.** El siguiente paso es autenticarse en Azure para provisionar los recursos de infraestructura necesarios.
- **4.** Una vez autenticado, seleccionar «Create new resources». Especificar la suscripción de Azure en la que se facturará el consumo, el grupo de recursos y la región en los que se creará la infraestructura, el nombre de la VM y, finalmente, su sistema operativo. La sección de sistema operativo por defecto se basa en el *runtime* del proyecto (Node.js o .NET).
- **5.** Al hacer clic en «Create», se inicia el proceso de provisión.
- **6.** Una vez completado, repetir el paso 1. Al seleccionar «Publish target», ahora aparece el perfil creado en los pasos previos. Seleccionar este perfil y hacer clic en «Publish selected bots».

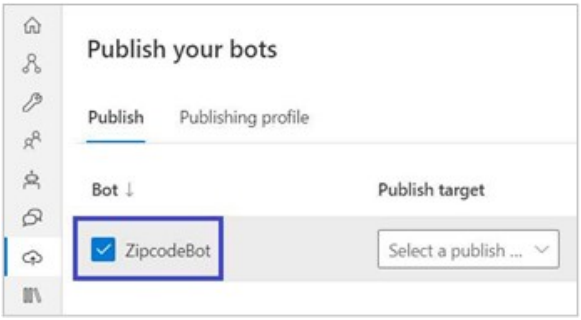



Figura 11. Publicación de bot desde Composer. Fuente: Freitas, s. f.

**Microsoft Copilot Studio**

La publicación de un *copilot* puede hacerse en una aplicación web o en Teams. Cada vez que se actualiza algún aspecto del *copilot,* se debe repetir el proceso para disponibilizar la nueva versión.

En el caso de publicación en una aplicación web, están disponibles las siguientes **opciones de autenticación:**

- **Sin autenticación.** Cualquiera puede utilizar el chatbot sin necesidad de autenticación previa.
- **Solo Teams y Power Apps.** Permite utilizar la sesión del usuario en estas dos plataformas para evitar una configuración expresa.
- **Manual.** En este caso, debe configurarse un método de autenticación mediante Azure Active Directory o OAuth2.

Para publicar el chatbot, solo es necesario tenerlo seleccionado para edición y hacer clic en «Publish». Por defecto, el chatbot se despliega en un sitio web de pruebas en el que se puede confirmar que los resultados son los esperados.

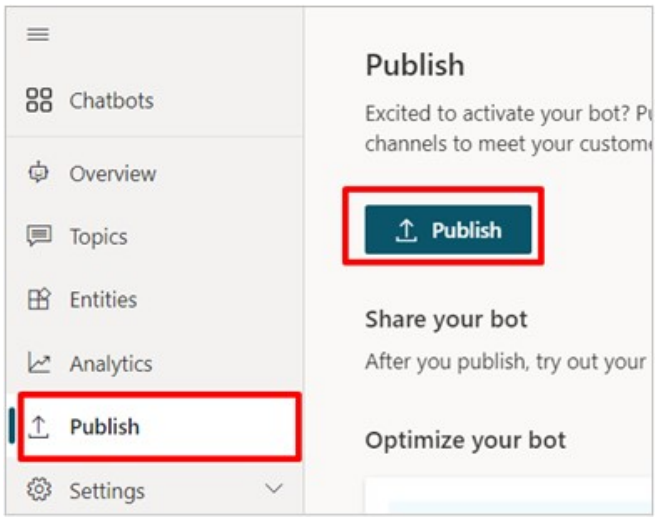



Figura 12. Publicación de un *copilot* en Copilot Studio. Fuente: Iaanw, cyrilanderson, lesliehirst, bolducs, 2024.

A continuación, es posible configurar los canales en los que el chatbot está disponible. Para ello, desde «Settings» se debe hacer clic en «Channels».

Se incluyen las siguientes opciones:

- **Microsoft Teams.**
- **Sitio web de pruebas** —activado por defecto cuando se publica con las opciones de autenticación manual o sin autenticación—.
- **Sitio web personalizado.** Este canal es el que permite integrar el chatbot en el sitio web de la organización.
- **Aplicación para móviles.**
- **Facebook.**
- **Los canales soportados por Azure Bot Service:** Skype, Slack, Telegram, Twilio, Line, Kik, GroupMe, Direct Line speech o correo electrónico.

## 7.5. Beneficios para el negocio y casos de uso de bots conversacionales

**Beneficios para el negocio**

Los chatbots proporcionan los siguientes beneficios a las organizaciones:

- **Disponibilidad.** Los chatbots atienden al usuario inmediatamente, evitando que tenga que esperar a que esté un agente (humano) disponible. De esa forma, se reduce la carga de los agentes y se incrementa la satisfacción del cliente.
- **Gestión de la capacidad.** Los chatbots pueden atender a múltiples usuarios de forma simultánea frente a la limitación de los agentes (humanos), que solo pueden atender a un usuario a la vez.
- **Flexibilidad.** Cada agente (humano) que se incorpora al equipo debe ser formado antes de poder atender peticiones de usuarios. Sin embargo, un chatbot solo debe ser programado en una ocasión.
- **Mejora de la experiencia del usuario.** Los agentes (humanos) pueden cometer errores o proporcionar información imprecisa al usuario por múltiples circunstancias. En cambio, el chatbot muestra siempre respuestas consistentes y precisas.
- **Bajo coste.** El coste de un chatbot en el medio/largo plazo es inferior al de mantener un equipo de agentes (humanos).
- **Automatización de tareas.** Los humanos suelen cometer errores frecuentes cuando se les encomiendan tareas repetitivas; por ejemplo, la labor de respuesta a consultas de clientes. En cambio, el chatbot está especialmente diseñado para asumir estas tareas repetitivas, sin cometer errores, y dejar que el agente se encargue de los escenarios que no pueden ser resueltos de forma automática.

- **Canal de venta.** El chatbot puede convertirse en un canal de venta que funciona en 24x7, sin supervisión humana. Siempre que sea programado correctamente, el chatbot puede ayudar a un cliente a completar todos los pasos de un proceso de venta.

- **Asistente personal.** Los usuarios valoran positivamente la asistencia que puede prestar un chatbot en la realización de sus tareas diarias o bien utilizarlo como fuente de recomendaciones (de viajes, ropa, contenidos, etc.).

Sin embargo, los chatbots también presentan algunas **desventajas:**

- **Incapacidad de entender al usuario.** La frustración que genera en los usuarios una interacción fallida con un chatbot puede afectar muy negativamente a su experiencia con el servicio.

- **Interfaces complejas.** Antes de la aparición de la IA generativa, la interacción con un chatbot era compleja, puesto que el éxito de esta dependía en gran medida de la habilidad del usuario para seguir el flujo conversacional del chatbot. Sin embargo, la aparición de sistemas como ChatGPT ha eliminado esta barrera.

- **Coste de actualizaciones.** Mantener un chatbot actualizado para reflejar los cambios en los procesos de negocio o la información de una compañía implica realizar nuevas inversiones, que pueden ser costosas.

- **Incapacidad de sacar conclusiones.** Por defecto, un chatbot no está programado para detectar incidencias repetidas o sugerir cambios necesarios en el procedimiento de atención a usuarios. En cambio, un agente (humano) puede observar estas circunstancias e informar a su supervisor para que se tomen medidas.

**Casos de uso**

En la Tabla 4 se muestran los casos de uso más populares en cada sector:

| Sector | Caso de uso |
|---|---|
| ***Utilities***<br><br>**Gas, electricidad, telecomunicaciones.** | Los clientes de estas compañías suelen formular preguntas relacionadas con los siguientes temas:<br>▸ La contratación de nuevos servicios.<br>▸ Cambios de tarifa<br>▸ Motivos de un aumento repentino o continuado de la facturación<br>▸ Interrupción del servicio<br>El chatbot debe disponer de una base de conocimiento amplia y actualizada para resolver las dudas del usuario en cada una de estas áreas. En función de su complejidad, puede limitarse a informar al usuario de los pasos a seguir o bien ejecutar él mismo las peticiones como si se tratase de un agente humano (por ejemplo, completar un cambio de tarifa sin supervisión humana). |
| **Seguros** | Los clientes de estas compañías formulan preguntas relacionadas con las coberturas de su póliza, el procedimiento para poner un parte o el estado de tramitación de un parte. También pueden utilizarse para atraer a nuevos clientes, realizando simulaciones de pólizas y cálculo de primas. |
| **Agencias de viajes** | Los chatbots de este sector se especializan en proporcionar detalles de vuelos o estancias, o bien en facilitar el proceso de contratación de un viaje. Este proceso es complejo, dado que a menudo involucra múltiples pasos relacionados, y la recuperación de información en tiempo real para proporcionar tarifas actualizadas. La capacidad de un bot de realizar tareas repetitivas de forma sistemática es fundamental para evitar errores durante la contratación de un viaje o la asistencia al viajero. |
| **Salud** | Los chatbots se utilizan principalmente para gestionar citas de los pacientes con los doctores. También pueden utilizarse para resolver cuestiones generales de salud, aunque en este caso se debe actuar con cautela para evitar casos de desinformación. |
| **Servicios financieros** | En el ámbito de servicios financieros, los *chatbots* pueden realizar múltiples tareas de gestión de forma automatizada:<br>▸ Emisión de tarjetas<br>▸ Reinicialización (*reset*) de contraseñas<br>▸ Pago de facturas<br>▸ Emisión de cheques<br>▸ Cumplimiento de criterios para concesión de créditos |

Tabla 4. Casos de uso de chatbots. Fuente: elaboración propia.

## 7.6. Cuaderno de ejercicios

**1.** ¿Cuáles son los pasos (alto nivel) del procedimiento que permite desplegar un chatbot en Facebook Messenger?

Los pasos que deben seguirse son los siguientes:

a. Establecer la configuración de la *app* de Facebook a la que estará asociada el *copilot.*

b. Activar el acceso por API a dicha *app.*

c. Añadir Facebook Messenger a la *app.*

d. Configurar las páginas de Facebook a las que se añadirá el bot.

e. Configurar el canal de Facebook en Copilot Studio.

f. Conectar la *app* de Facebook con Copilot Studio.

g. Enviar la *app* al proceso de revisión de Facebook.

h. Publicar la *app* y la página en Facebook.

---

Extraído de: laanw, boducs, cyrilanderson (2024, mayo 6). Add a copilot to Facebook. *Microsoft Learn.* https://learn.microsoft.com/en-us/microsoft-copilot-studio/publication-add-bot-to-facebook

---

**2.** En relación con la analítica de uso de un copilot, ¿cuáles son los posibles estados de una sesión y a qué circunstancias están asociados?

Los estados posibles son los siguientes:

- *Unengaged.* Se refiere al inicio de la sesión, cuando el *copilot* ha enviado de forma proactiva un mensaje al usuario.
- *Engaged.* Este estado se adquiere cuando se activa uno de los *topics.*
- *Escalated.* Este estado se adquiere cuando se activa el *topic* Escalate o se ejecuta el nodo Transfer to agent.
- *Resolved.* Este estado se adquiere cuando se activan los *topics* de Confirmed success o End of conversation. En el segundo caso, el usuario debe confirmar la finalización exitosa de la conversación.
- *Abandoned.* Cuando no se alcanza el estado *resolved* o *escalated* en el plazo de tiempo máximo fijado.

---

Extraído de: laanw, kimberleybrown, bolducs y cyrilanderson (2024, febrero 4). Analyze copilot performance and usage in Copilot Studio. *Microsoft Learn.* https://learn.microsoft.com/en-us/microsoft-copilot-studio/analytics-summary

---

**3.** ¿Cuáles son los indicadores de rendimiento que aparecen en el área «Resumen» de la sección «Analytics» de un *copilot,* y cuál es su significado?

Los indicadores que se muestran son los siguientes:

- Número total de sesiones. Se refiere al número de sesiones de usuario registradas en el período de tiempo especificado.
- *Ratio de engagement.* Porcentaje de sesiones que alcanzaron el estado *engaged,* es decir, en las que hubo alguna interacción con el usuario.
- Ratio de resolución. Porcentaje de sesiones que alcanzaron el estado *resolved,* es decir, que finalizaron satisfactoriamente.
- Ratio de escalado. Porcentaje de sesiones que alcanzaron el estado *escalated,* es decir, que requirieron la atención de un agente (humano).
- Ratio de abandono. Porcentaje de sesiones que fueron abandonadas por el usuario sin una resolución satisfactoria.
- Satisfacción de cliente. Promedio de puntuación otorgada por los usuarios que aceptaron cumplimentar la encuesta de satisfacción.

**4.** ¿Cuáles son los escenarios considerados en la plantilla Enterprise Assistant Bot de Azure Bot Framework Composer, y qué tareas puede realizar el bot en cada uno de ellos?

Los escenarios considerados son los siguientes. Para cada uno de ellos, se mencionan las funcionalidades incluidas.

- *Core assistant:*
  - Saludar a usuarios nuevos y recurrentes.
  - Ofrecer un *tour* de funcionalidades.
  - Posibilidad de pedir ayuda (por parte del usuario).
  - Abandonar una conversación.
  - Recibir *feedback.*
  - Manejar errores.
  - Repetir una respuesta.
- *Enterprise calendar:*
  - Agenda del día: completa, primer/último evento del día.
  - Localizar, crear o actualizar un evento.
  - Responder a un evento.
  - Huecos libres.
- *Enterprise people:*
  - Información de contacto de un compañero.

- Información del supervisor, colaboradores, con quién trabaja o quiénes están en su mismo puesto.
- Contactar con un compañero vía teléfono, correo o mensaje.

---

Extraído de: JonathanFingold, eric-urban, emgrol y erhopf (2022, noviembre 22). Enterprise Assistant Bot template for Composer. *Microsoft Learn.* https://learn.microsoft.com/en-us/composer/templates/enterprise-assistant-overview

---

**5.** En el momento de la provisión de un chatbot creado con Azure Bot Framework, ¿cuáles son los recursos que deben crearse obligatoriamente en Azure para que el chatbot esté operativo?

Los recursos requeridos son los siguientes:

- **Registro de una aplicación,** para que el chatbot pueda comunicarse con los servicios de Azure.
- **App Service,** que provee la infraestructura necesaria para la ejecución del chatbot en entorno web, móvil o vía API.
- **Azure Bot,** que incluye los componentes del propio chatbot.

---

Extraído de: JonathanFingold, eric-urban-emgrol. v-alarioza y MarcFromTerawe (2024, enero 16). How to provision Azure resources. *Microsoft Learn.* https://learn.microsoft.com/en-us/composer/how-to-provision-azure-resources

---

## 7.7. Referencias bibliográficas

Elijah, R. (2023, julio 4). 6 Power Platform copilot and AI features you can try out today. *Rishona Elijah.* https://rishonapowerplatform.com/2023/07/04/6-power-platform-copilot-and-ai-features-you-can-try-out-today/

Freitas, E. (s. f.). Publishing the Bot. En *Azure Bot Service Succinctly.* Syncfusion. https://www.syncfusion.com/succinctly-free-ebooks/azure-bot-service-succinctly/publishing-the-bot

Iaanw, boducs, cyrilanderson (2024, junio 5). Add a copilot to Facebook. Microsoft Learn. https://learn.microsoft.com/en-us/microsoft-copilot-studio/publication-add-bot-to-facebook

Iaanw, cyrilanderson, lesliehirst y bolducs (2024a, mayo 21). Quickstart guide for building copilots with generative AI. *Microsoft Learn.* https://learn.microsoft.com/en-us/microsoft-copilot-studio/nlu-gpt-quickstart

Iaanw, kimberleybrown, bolducs y cyrilanderson (2024, febrero 4). Analyze copilot performance and usage in Copilot Studio. *Microsoft Learn.* https://learn.microsoft.com/en-us/microsoft-copilot-studio/analytics-summary

Iaanw, kimberleybrown, bolducs, cyrilanderson y peterswimm (2024, junio 4). Key concepts - Publish your copilot. *Microsoft Learn.* https://learn.microsoft.com/en-us/microsoft-copilot-studio/publication-fundamentals-publish-channels?tabs=web

Iaanw, sad-cl0wn, bolducs y peterswimm (2024, mayo 21). Test your copilot in Copilot Studio. *Microsoft Learn.* https://learn.microsoft.com/en-us/microsoft-copilot-studio/authoring-test-bot?tabs=webApp

Iaanw, sad-cl0wn, bolducs, GitikaG, peterswimm y svielse (2024, junio 6). Quickstart: Create and deploy a Copilot Studio copilot. *Microsoft Learn.* https://learn.microsoft.com/en-us/microsoft-copilot-studio/fundamentals-get-started?tabs=web

JonathanFingold, damabe, emgrol, v-alarioza, stevemunk y MarcFromTerawe (2023, julio 19). Tutorial: Create a weather bot with Composer. *Microsoft Learn.* https://learn.microsoft.com/en-us/composer/tutorial-create-weather-bot

JonathanFingold, emgrol, stevemunk, ianl-terawe, erhopf, Kaiwb y zxyanliu (2022, octubre 6). Install Bot Framework Composer. *Microsoft Learn.* https://learn.microsoft.com/es-es/composer/install-composer?tabs=macos

JonathanFingold, eric-urban, emgrol y erhopf (2022, noviembre 22). Enterprise Assistant Bot template for Composer. *Microsoft Learn.* https://learn.microsoft.com/en-us/composer/templates/enterprise-assistant-overview

JonathanFingold, eric-urban-emgrol. v-alarioza y MarcFromTerawe (2024, enero 16). How to provision Azure resources. *Microsoft Learn.* https://learn.microsoft.com/en-us/composer/how-to-provision-azure-resources

JonathanFingold, iaanw, eric-urban, emgrol, erhopf, stevemunk, WashingtonKayaker, Quirinevwm, zxyanliu y benbrown (2022, noviembre 22). Best practices for building bots with Composer. *Microsoft Learn.* https://learn.microsoft.com/en-us/composer/concept-best-practices?tabs=v2x

# Ideas clave

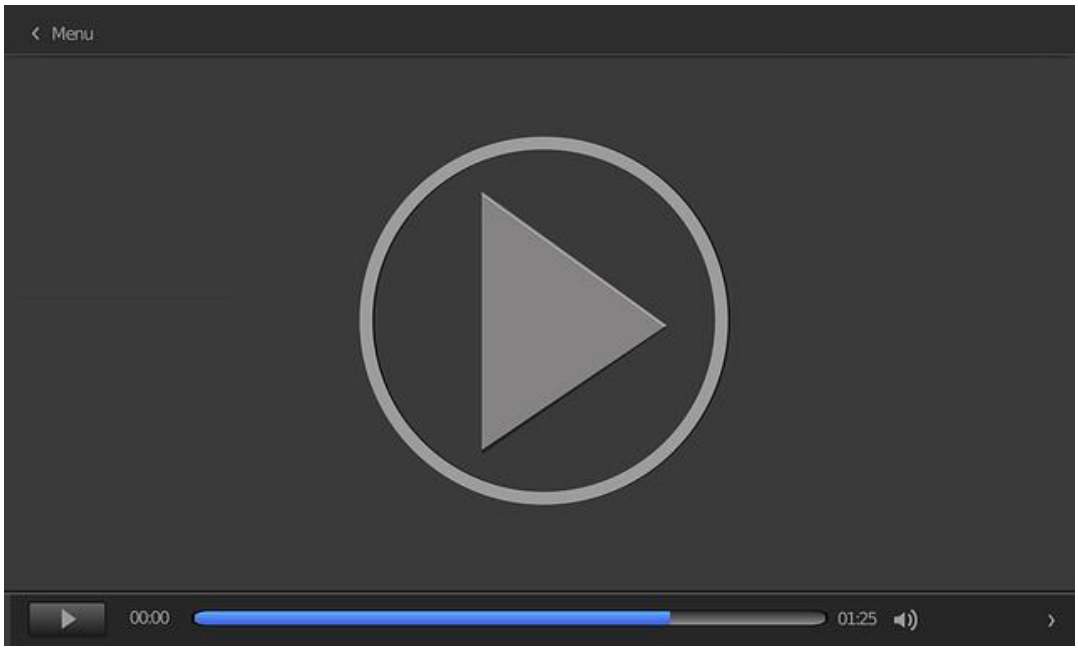
KendalBond007, cyrilanderson y sad-cl0wn (2024, mayo 21). Create and edit topics with Copilot Studio. *Microsoft Learn.* https://learn.microsoft.com/en-us/microsoft-copilot-studio/nlu-authoring

KendalBond007, mainguy70, bolducs, raemone, cyrilanderson, Eric-tw-convAI y sophie-roy3 (2024, mayo 21). Generative answers. *Microsoft Learn.* https://learn.microsoft.com/en-us/microsoft-copilot-studio/nlu-boost-conversations

Página de Microsoft Copilot Studio. https://copilotstudio.microsoft.com/environments/Default-22c8b4a4-d926-43b2-bcc7-87b998590b47/bots/~previous

## Bot Framework Composer

Microsoft Developer (2020, enero 28). *Bot Framework Composer* [Vídeo]. YouTube. https://www.youtube.com/watch?v=Aiv95e2-Yn0



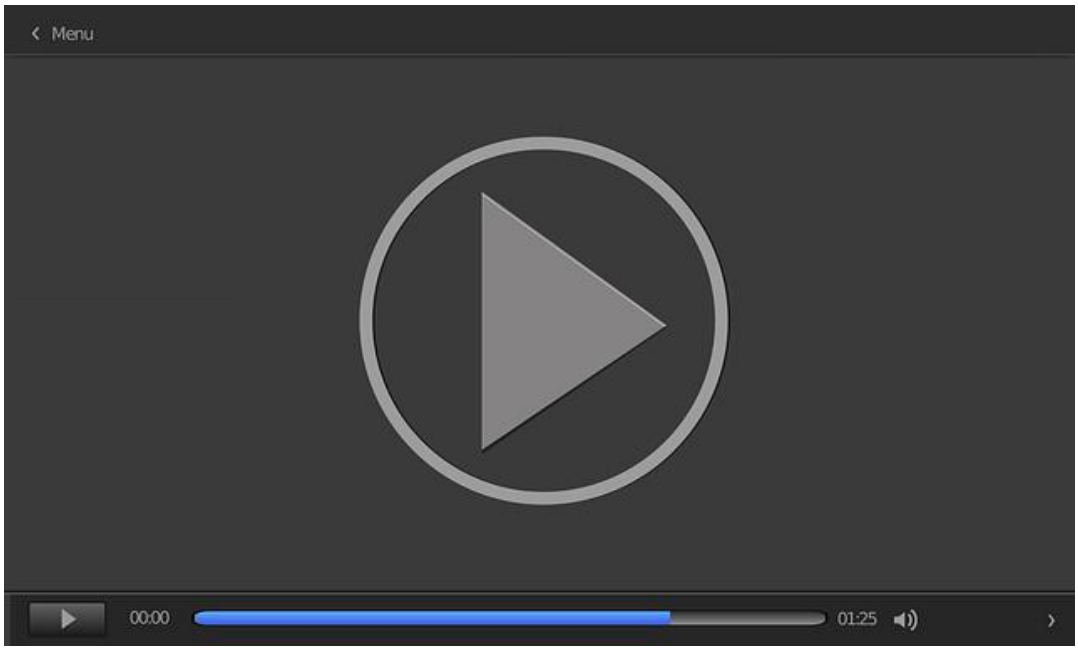

Accede al vídeo:

https://www.youtube.com/embed/Aiv95e2-Yn0

En este vídeo se muestra un ejemplo práctico de creación de un chatbot basado en Azure Bot Framework Composer. El alumno puede reproducir los pasos del tutorial en su propio PC para adquirir más experiencia en el uso de esta plataforma.

## Introduction to Microsoft Copilot Studio

Microsoft Community Learning (2024, abril 10). *Introduction to Microsoft Copilot Studio* [Vídeo]. YouTube. https://www.youtube.com/watch?v=8gdIDiJt4Uw




Accede al vídeo:

https://www.youtube.com/embed/8gdIDiJt4Uw

En este vídeo se presentan las características de Copilot Studio y se muestran ejemplos prácticos de uso de sus funcionalidades para crear *copilots.* El alumno puede de esta forma ampliar los conceptos descritos en este tema y ganar experiencia práctica en el uso de esta herramienta.

# Test

1. En la terminología de Azure Bot Framework, ¿qué es un diálogo?

   A. Un diálogo es el componente visual a través del cual se registran las conversaciones entre el chatbot y el usuario.

   B. Un diálogo representa una funcionalidad del bot, y define cómo reaccionará a las peticiones del usuario.

   C. Un diálogo se refiere al conjunto de mensajes que intercambian el usuario y el chatbot.

   D. Un diálogo es un conjunto de respuestas posibles del chatbot a la petición de un usuario.

2. ¿Para qué sirve un *recognizer* en Azure Bot Framework?

   A. AAA.

   B. Un *recognizer* detecta el significado de la petición del usuario *(intent)* y extrae sus principales entidades *(entities).*

   C. Un recognizer es llamado cuando sucede algún evento relevante en la conversación.

   D. Un *recognizer* implementa acciones de respuesta a partir de la activación de un *trigger.*

**3.** ¿Cómo se realiza la implementación del concepto de memoria en un chatbot creado en Azure Bot Framework?

A. La memoria se implementa a través de variables y constantes, que tienen un nombre y un valor.

B. La memoria se implementa a través de propiedades *(properties),* que tienen un nombre, un alcance y un valor.

C. La memoria se implementa a través de propiedades *(properties),* que tienen un nombre y un valor.

D. La memoria se implementa a través de variables y constantes, que tienen un nombre, un alcance y un valor.

**4.** ¿Qué es y para qué sirve el Adaptive Runtime?

A. Es el entorno de ejecución en el que se hospedan los chatbots.

B. Es el entorno de ejecución que debe incorporarse como dependencia a un proyecto de chatbot para acceder a la SDK de Bot Framework.

C. Es el entorno de ejecución que debe incorporarse como dependencia a un proyecto de chatbot para acceder a la SDK de Microsoft Copilot Studio.

D. Es el entorno a través del cual se puede desplegar un chatbot.

**5.** ¿Para qué se utiliza un sitio web de *fallback* en Microsoft Copilot Studio?

A. Es la URL a la que accede el *copilot* cuando se produce un error durante el intercambio de mensajes en la conversación con el usuario.

B. Es la URL a la que accede el *copilot* cuando desconoce la respuesta a una petición del usuario para buscar información relevante que permita dar con dicha respuesta.

C. Es la URL a partir de la que el *copilot* genera todas sus respuestas, lo que evita tener que entrenar un modelo previamente.

D. Es la URL a través de la cual puede utilizarse el *copilot.*

**6.** ¿Qué es un perfil de publicación *(publishing profile)* en Azure Bot Framework?

A. Se refiere al nivel de autenticación necesario para utilizar un chatbot.

B. Se refiere a la configuración utilizada para hacer despliegues de chatbots, incluyendo la infraestructura de Azure en la que se hospedan.

C. Se refiere al nivel de autorización necesario para desplegar un chatbot.

D. Se refiere a la configuración utilizada para hacer pruebas de chatbots, como paso previo a su despliegue en producción.

**7.** ¿Qué aplicaciones tiene un chatbot en el sector de seguros?

A. El sector seguros no se beneficia de las ventajas de un chatbot.

B. Permite resolver dudas de los clientes, relacionadas con la cobertura de su póliza, el procedimiento para generar un parte o el estado de tramitación de un parte.

C. Se utilizan para que los agentes de seguros puedan determinar la probabilidad de venta de un seguro a un potencial cliente.

D. Se utilizan para atraer a nuevos clientes con mensajes promocionales, pero no tienen aplicaciones en los clientes existentes por la complejidad de los procesos de negocio en este sector.

**8.** ¿En qué lenguajes de programación puede escribirse el código fuente de los chatbots creados en Azure Bot Framework?

A. Python y Java.

B. C#/.NET, Node.js o Javascript.

C. C# exclusivamente.

D. Node.js o Java.

**9.** ¿Cuál de las siguientes no es una mejor práctica en el diseño de la experiencia conversacional?

A. Incluir un diálogo para pedir ayuda o finalizar la conversación.

B. Utilizar exactamente las mismas respuestas cada vez a una misma petición del usuario para no dar lugar a una interpretación incorrecta.

C. Mantener los diálogos breves.

D. Añadir variaciones a las respuestas

**10.** ¿Qué diferencia a un *skill* de un chatbot convencional?

A. Una habilidad o *skill* es el tipo de bot que interactúa directamente con el usuario.

B. Una habilidad o *skill* es un bot que no interactúa directamente con el usuario, sino que proporciona una funcionalidad determinada a otros bots.

C. Una habilidad o *skill* está programado en un lenguaje diferente que un bot convencional. A pesar de ellos, ambos son interoperables.

D. No existen diferencias, dado que se trata de términos utilizados para referirse al mismo tipo de tecnología conversacional.