

Google Cloud



# Google Cloud Dialogflow

**stratebi**  
open business intelligence

## 1. INTRODUCCIÓN

Google define **Dialogflow** como “una plataforma con comprensión del lenguaje natural que te facilita el diseño de una interfaz de usuario de conversación y su integración a tu aplicación para dispositivos móviles, aplicación web, dispositivo, bot, sistema de respuesta de voz interactiva y más”.

**Dialogflow** permite el análisis de varios tipos de entrada, como texto o audio, y puede responder con texto o con voz sintética.

## 2. CONCEPTOS

### 1. Agentes

Son módulos de comprensión del lenguaje natural que manejan los tipos de conversaciones requeridas con los usuarios finales.

**Dialogflow** tiene dos tipos de agentes:

- **CX** – es un tipo de agente avanzado, adecuado para crear agentes grandes o complejos; los componentes básicos del diseño de conversaciones son los flujos y las páginas, y se usan controladores de estado para controlar las rutas de conversación.
- **ES** – es el tipo de agente estándar, adecuado para crear agentes pequeños, medianos o moderadamente complejos; los componentes básicos del diseño de las conversaciones son los intents, y se usan los contextos para controlar las rutas de conversación.

En este documento se habla sobre el **Dialogflow ES**.

### 2. Intents

Un intent clasifica la intención del usuario final para un turno de conversación. Para manejar una conversación completa, se definen muchos intents para un agente y cuando un usuario escribe o dice algo Dialogflow hace coincidir la expresión con el mejor intent.

Un intent contiene los siguientes elementos:

- *Frases de entrenamiento* – son frases de ejemplo que podrían decir los usuarios finales; son las frases que usa Dialogflow para hacer coincidir las expresiones de los usuarios finales con los intents definidos en los agentes; no es necesario definir todos los ejemplos posibles.
- *Acción* – se define por cada intent y se usa para activar ciertas acciones en el sistema.

- *Parámetros* – cuando un intent coincide en el entorno de ejecución, Dialogflow proporciona los valores extraídos de la expresión del usuario final como parámetros; a diferencia de la entrada sin procesar del usuario final, los parámetros son datos estructurados que se pueden usar con facilidad para realizar alguna lógica o generar respuestas.
- *Respuestas* – proporcionan respuesta al usuario final, pueden ser de texto, de voz o visuales.

### 3. Entidades

Cada parámetro de un intent, llamado tipo de entidad, que dicta como se extraen los datos de una expresión.

**Dialogflow** proporciona entidades predefinidas para muchos tipos comunes de datos, como pueden ser: fechas, horas, colores, direcciones de correo electrónico. También se pueden definir entidades personalizadas.

### 4. Contextos

Sirven para controlar el flujo de una conversación y son de dos tipos:

- Contextos de entrada – controlan la detección de coincidencias con intents;
- Contextos de salida – controlan los contextos activos; cuando se detecta una coincidencia con un intent, se activan los contextos de salida configurados para ese intent.

### 5. Intents de seguimiento

Se usan para establecer contextos para grupos de intents de forma automática. Un intent de seguimiento es elemento secundario del intent superior asociado y solo coincide cuando el intent superior coincide en el turno anterior de conversación.

### 6. Fulfillment

De forma predeterminada, un agente responde al intent coincidente con una respuesta estática. Con *fulfillment* (entrega) se pueden proporcionar respuestas más dinámicas, llamando al servicio definido. Cada intent tiene una configuración que permite la entrega.

## 3. INTERFAZ DIALOGFLOW

### 1. Intents

Permite la creación y gestión de intents con toda la configuración necesaria.

### 2. Entities

Permite la creación y gestión de las entidades necesarias para el agente.

### 3. Knowledge

Permite definir y gestionar conectores de conocimiento que complementan los intents definidos. Estos analizan documentos de conocimiento (FAQs, artículos...) para encontrar respuestas automáticas. Es una función en Beta por lo que hay que activarla en la configuración del agente.

### 4. Fulfillment

Permite definir respuestas más dinámicas usando Google Cloud Functions.

## 5. Integrations

Permite ver y configurar las integraciones con otros servicios (Slack, Messenger ...).

The screenshot shows the Dialogflow Integrations interface. On the left, the 'Integrations' menu is active. The main panel features the Google Assistant logo and a 'Try it now' button. Below this, there are four integration options: Dialogflow Phone Gateway, Avaya, SignalWire, and Voximplant. The right sidebar displays the 'Agent' configuration for 'reservar-mesa', including user input, default response, contexts, intent, and action parameters.

## 6. Training

Permite ver y añadir a los intentos las frases que se han usado para probar el agente. También permite la subida de conjuntos de datos de entrenamiento.

The screenshot shows the Dialogflow Training interface. On the left, the 'Training' menu is active. The main panel displays a table with training data. The table has columns for Conversation, Requests, No match, and Date. The right sidebar displays the 'Agent' configuration for 'reservar-mesa', including user input, default response, contexts, intent, and action parameters.

Conversation	Requests	No match	Date
reserva para dos personas hoy a las 14h	2	0	Today
hola	1	0	Today
reserva para dos personas hoy a las 14h	1	0	Today

## 7. Validation

Permite ver información, advertencias y errores del agente.

The screenshot shows the Google Cloud Dialogflow Validation interface. The left sidebar contains navigation options: Intents, Entities, Knowledge, Fulfillment, Integrations, Training, Validation (selected), History, Analytics, Prebuilt Agents, Docs, Trial, Upgrade, Dialogflow CX, Support, and Account. The main panel is titled 'Validation' and shows 'Agent Issues' with a search bar and filters for ERROR, WARNING, and INFO. Below this, 'Intent Issues' and 'Entity Issues' are listed. The right sidebar shows the 'Agent' configuration, including user input, default response, contexts, intent, and action.

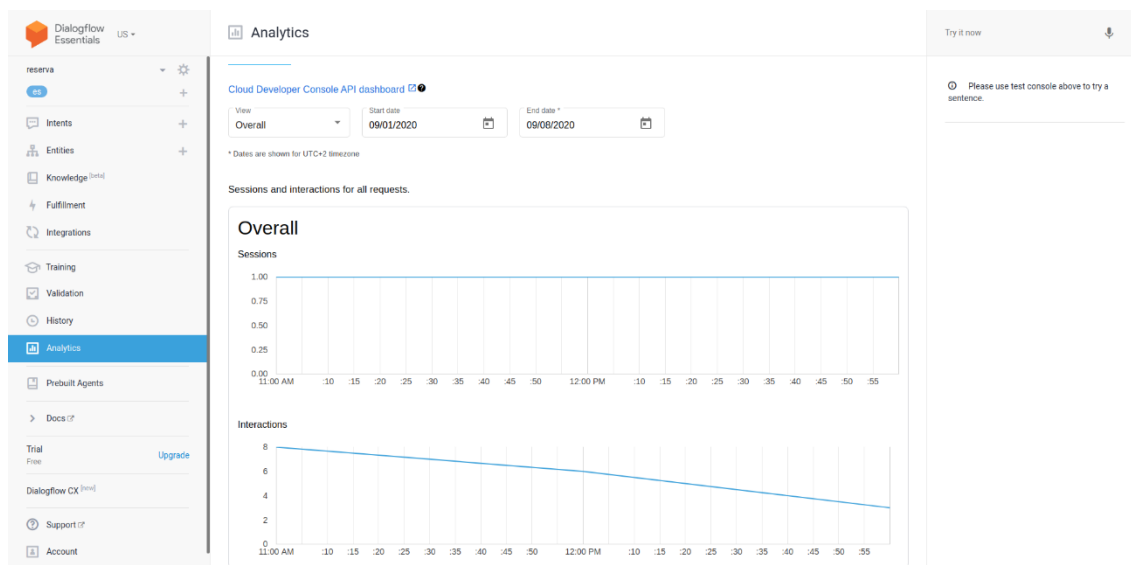
## 8. History

Permite ver todos los mensajes de todos los usuarios

The screenshot shows the Google Cloud Dialogflow History interface. The left sidebar contains navigation options: Intents, Entities, Knowledge, Fulfillment, Integrations, Training, Validation, History (selected), Analytics, Prebuilt Agents, Docs, Trial, Upgrade, Dialogflow CX, Support, and Account. The main panel is titled 'History' and shows a list of conversations with columns for Conversation, Date, and a refresh icon. The right sidebar shows the 'Agent' configuration, including user input, default response, contexts, intent, and action.

## 9. Analytics

Permite ver las métricas (interacciones, número de sesiones) del agente.



## 4. EJEMPLO CREACIÓN DE AGENTES

En este ejemplo crearemos un agente que se encarga de las reservas de un restaurante. Definiremos dos intents, uno para obtener el menú del día y otro para reservar una mesa. El intent de reserva tendrá intents de seguimiento para confirmar que el usuario desea continuar con la reserva y para obtener los datos de contacto.

Para crear agentes hay que navegar a la [Consola de Dialogflow ES](#) y pinchar en *Create Agent*

### 1. Creación del agente

Escribimos el nombre del agente, elegimos el idioma y la zona horaria, y creamos o elegimos un proyecto de *Google Cloud Platform*.

The screenshot shows the 'Create Agent' form in Dialogflow. The agent name is 'reserva'. There is a 'CREATE' button in the top right. The form includes fields for 'DEFAULT LANGUAGE' (Spanish - es), 'DEFAULT TIME ZONE' ((GMT+1:00) Europe/Madrid), and 'GOOGLE PROJECT' (test-dialogflow-288023). At the bottom, there is a section for 'AGENT TYPE' with a toggle for 'Set as Mega Agent'.



## 7. Intent obtener-menu

Creamos un intent para obtener el menú del día.

Añadimos las frases de entrenamiento.

Creamos la respuesta.

## 8. Intent reservar-mesa

Añadimos las frases de entrenamiento y definimos los parámetros *numero\_personas* y *fecha\_hora\_reserva*. Marcamos las casillas *REQUIRED* para especificar que los parámetros son necesarios. En *PROMPTS* definimos las preguntas que el agente hará a los usuarios si la solicitud no contiene los parámetros.

**Training phrases** ⓘ

Search training phrases 🔍 ^

” Add user expression

” reservar mesa 🗑️

” reserva para 3 personas hoy a las 13:25

” reserva

” quiero hacer una reserva para una persona para el próximo miércoles a las 14

” reserva de cuatro para el día 11 de septiembre a las 15:00

” quiero reservar una mesa de 4 personas para mañana a las 14h

**Action and parameters** ^

Enter action name

REQUIRED ⓘ	PARAMETER NAME ⓘ	ENTITY ⓘ	VALUE	IS LIST ⓘ	PROMPTS ⓘ
<input checked="" type="checkbox"/>	numero_personas	@sys.number	\$numero_personas	<input type="checkbox"/>	¿Cuántas person...
<input checked="" type="checkbox"/>	fecha_hora_reserva	@sys.date-time	\$fecha_hora_reserva	<input type="checkbox"/>	¿Qué día y a qu...
<input type="checkbox"/>	Enter name	Enter entity	Enter value	<input type="checkbox"/>	—

Definimos la respuesta usando los parámetros obtenidos.

**Responses** ⓘ ^

DEFAULT +

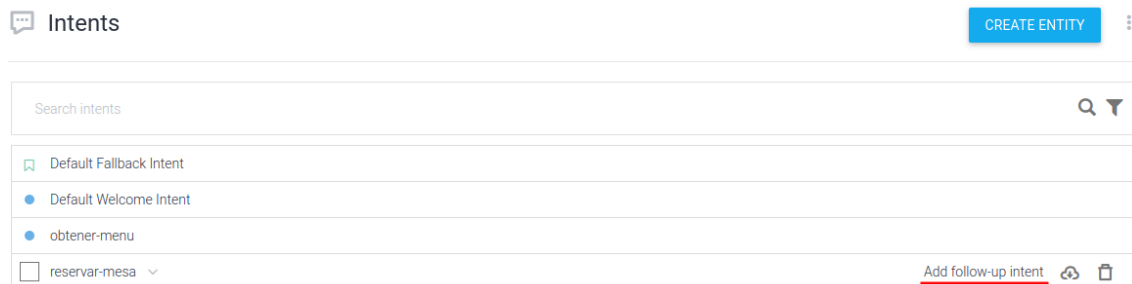
**Text Response** 🗑️

- 1 Reserva para \$numero\_personas personas, \$fecha\_hora\_reserva. ¿Deseas continuar?
- 2 Enter a text response variant

ADD RESPONSES

☐ Set this intent as end of conversation ⓘ

Creamos dos intents de seguimiento, afirmación y negación, para el intent *reservar-mesa* pinchando en *Add follow-up intent*. Se despliega un selector con varios tipos de intents. Anadimos *yes* y *no*.

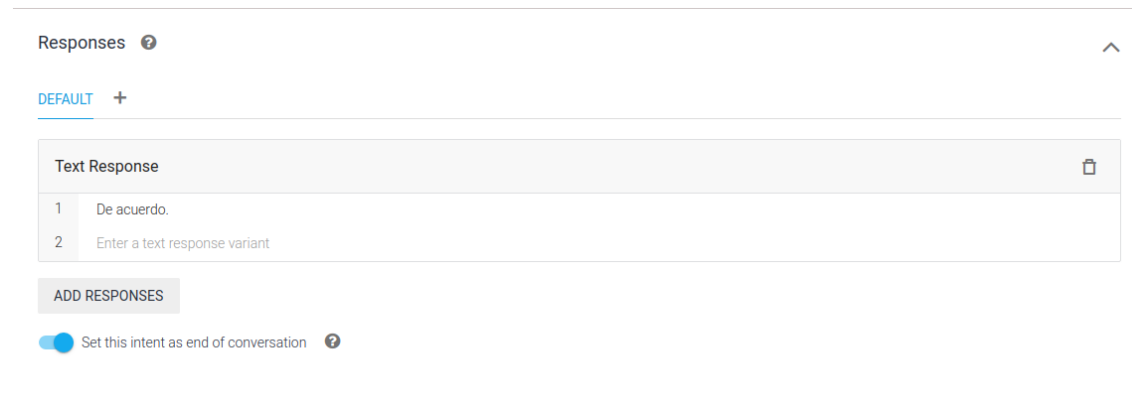


**Intents** CREATE ENTITY

Search intents

- Default Fallback Intent
- Default Welcome Intent
- obtener-menu
- reservar-mesa ▼ Add follow-up intent

Navegamos al intent *reservar-mesa - no*. Observamos que Dialogflow ha insertado algunas frases de entrenamiento. Definimos las respuestas y marcamos *Set this intent as end of conversation*.



**Responses**

DEFAULT +

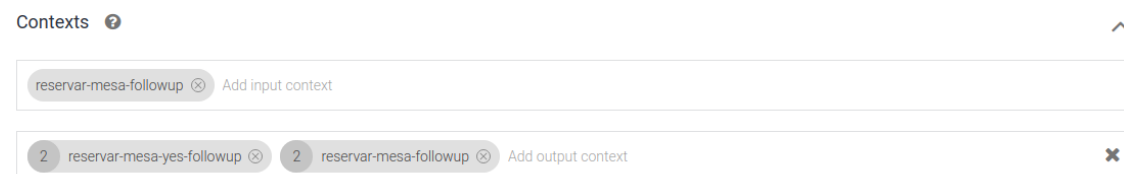
**Text Response**

- De acuerdo.
- Enter a text response variant

ADD RESPONSES

☒ Set this intent as end of conversation

Navegamos al intent *reservar-mesa - yes*. Observamos que Dialogflow ha insertado algunas frases de entrenamiento. Pasamos el contexto *reservar-mesa-followup* en el output para que el intent de seguimiento de este intent (*reservar-mesa - yes*) tenga los datos obtenidos en el intent *reservar-mesa*.



**Contexts**

reservar-mesa-followup ⊗ Add input context

2 reservar-mesa-yes-followup ⊗ 2 reservar-mesa-followup ⊗ Add output context

Definimos las respuestas.

Responses

DEFAULT +

Text Response

1 ¿Me puedes decir tu nombre y un número de teléfono?

2 Enter a text response variant

ADD RESPONSES

☐ Set this intent as end of conversation

Creamos un intent de seguimiento para el intent *reservar-mesa - yes*. Añadimos el contexto *reservar-mesa-followup* en el input.

Contexts

reservar-mesa-yes-followup

reservar-mesa-followup

Add input context

Add output context

Añadimos las frases de entrenamiento y definimos los parámetros *nombre* y *numero-telefono*. Igual que en el intent *reservar-mesa*, marcamos las casillas *REQUIRED* para especificar que los parámetros son necesarios y definimos los *PROMPTS*.

Training phrases

Search training phrases

Add user expression

mi movil es 745 600 168

mi numero 698831465

soy Gabriel, mi telefono es 759601647

mi nombre es Mihai

me llamo Ivan y mi numero de telefono es 685 149 880

reservar-mesa.reservar-mesa-yes.reservar-mesa-yes-datos-contacto

REQUIRED	PARAMETER NAME	ENTITY	VALUE	IS LIST	PROMPTS
<input checked="" type="checkbox"/>	nombre	@sys.person	\$nombre	<input type="checkbox"/>	¿Me dices un no...
<input checked="" type="checkbox"/>	numero-telefono	@sys.phone-number	\$numero-telefono	<input type="checkbox"/>	¿Me pasas tu nú...
<input type="checkbox"/>	Enter name	Enter entity	Enter value	<input type="checkbox"/>	—

Definimos la respuesta, usando los parámetros obtenidos y los del contexto *reservar-mesa-followup*. Marcamos *Set this intent as end of conversation*.

Responses ⓘ

DEFAULT +

Text Response ⓘ

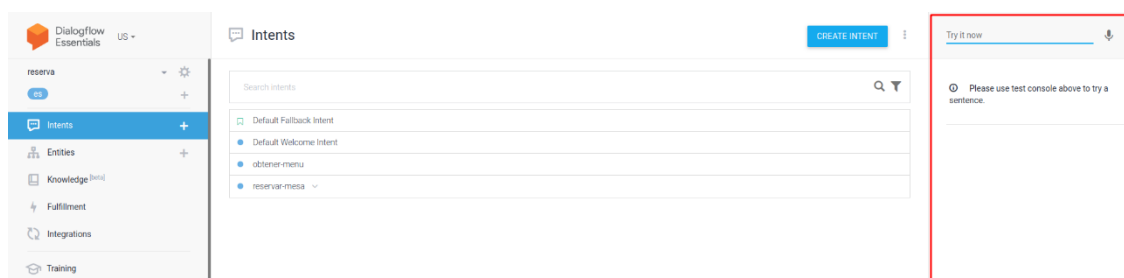
1	\$nombre, he hecho la reserva para #reservar-mesa-followup.numero_personas personas, #reservar-mesa-followup.fecha_hora_reserva. Gracias por elegirnos.
2	Enter a text response variant

ADD RESPONSES

☒ Set this intent as end of conversation ⓘ

## 9. Pruebas

Desde la interfaz de Dialogflow se puede probar el agente.



Probamos con la frase "Enséñame el menú" y observamos que la intención del usuario coincide con el intent *obtener-menu*.

Try it now

Agent

USER SAYS

COPY CURL

Enséñame el menú

DEFAULT RESPONSE

Primero a elegir entre Sopa de lentejas y Crema de garbanzos con sésamo. Segundo a elegir entre Pimientos rellenos de arroz y verduras y Pollo al grill con salsa de mostaza.

INTENT

obtener-menu

ACTION

Not available

DIAGNOSTIC INFO

Probamos con la frase "Reservame una mesa para cuatro personas, hoy a las 14 horas" y observamos que la intención del usuario coincide con el intent reservar-mesa y que se han obtenido también los parámetros.

**Agent**


---

USER SAYS

COPY CURL

Reservame una mesa para cuatro persona, hoy a las 14 horas

---

 DEFAULT RESPONSE

▼

Reserva para 4 personas, 2020-09-08T14:00:00.  
¿Deseas continuar?

---

CONTEXTS

RESET CONTEXTS

reservar-mesa-followup

---

INTENT

reservar-mesa

---

ACTION

Not available

PARAMETER	VALUE
fecha_hora_reserva	{ "date_time": "2020-09-08T14:00:00+02:00" }
numero_personas	4

---

DIAGNOSTIC INFO

Continuamos la conversación con la frase "sí". Observamos que el agente pregunta por el nombre y el número de teléfono.

**Agent**



---

USER SAYS

COPY CURL

si

---

 DEFAULT RESPONSE   
¿Me puedes decir tu nombre y un número de teléfono?

---

CONTEXTS

RESET CONTEXTS

reservar-mesa-yes-followup

reservar-mesa-followup

---

INTENT  
reservar-mesa - yes

---

ACTION  
reservar-mesa.reservar-mesa-yes

---

DIAGNOSTIC INFO

Le pasamos solo el nombre. Observamos que se ha obtenido el parámetro *nombre* y que el agente pregunta por el teléfono.



## Agent

USER SAYS

[COPY CURL](#)

Mihai



DEFAULT RESPONSE



¿Me pasas tu número de teléfono?

CONTEXTS

[RESET CONTEXTS](#)

711deb7c-44d7-4088-87cb-85007ee373b  
0\_id\_dialog\_context

reservar-mesa\_-\_yes\_-\_datos-contacto\_di  
alog\_context

reservar-mesa\_-\_yes\_-\_datos-contacto\_di  
alog\_params\_numero-telefono

reservar-mesa-followup

reservar-mesa-yes-followup

\_\_system\_counters\_\_

INTENT

reservar-mesa - yes - datos-contacto

ACTION

reservar-mesa.reservar-mesa-yes.reservar-  
mesa-yes-datos-contacto

PARAMETER

VALUE

nombre

{ "name": "Mihai" }

numero-telefono

Le pasamos el número de teléfono. Observamos la respuesta del agente y los parámetros.

## Agent

USER SAYS

[COPY CURL](#)

600600600



DEFAULT RESPONSE ▼

Mihai, he hecho la reserva para 4 personas,  
2020-09-08T14:00:00+02:00. Gracias por  
elegirnos.

CONTEXTS

[RESET CONTEXTS](#)

reservar-mesa-followup

reservar-mesa-yes-followup

INTENT

reservar-mesa - yes - datos-contacto

ACTION

reservar-mesa.reservar-mesa-yes.reservar-mesa-yes-datos-contacto

PARAMETER

VALUE

nombre

{ "name": "Mihai" }

numero-telefono

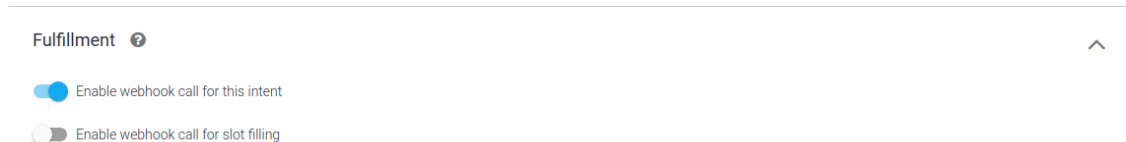
600600600

## DIAGNOSTIC INFO

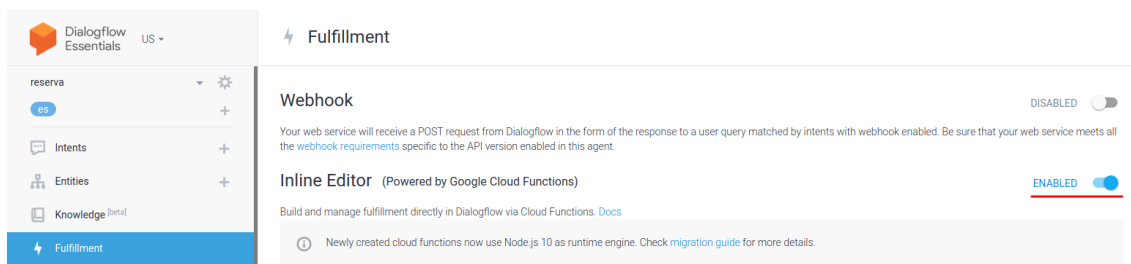
## 10. Fulfillment

Como se puede observar, el formato de la fecha (ISO 8601) no es amigable. Lo podemos cambiar usando *Fulfillment*.

Navegamos a *Intents* y habilitamos *Enable webhook call for this intent* para los intents *reservar-mesa* y *reservar-mesa - yes - datos-contacto*.



Luego, navegamos a *Fulfillment* y habilitamos *Inline Editor*. Esto nos permite crear funciones directamente en Dialogflow.



Hay dos ficheros: *index.js* y *package.json*. Los completamos con la siguiente información:

- *index.js* - definimos y registramos la función *respuesta\_reserva* para el intent *reserva-mesa*; esta función parsea la fecha y cambia el formato a uno más amigable.

```
'use strict';

const functions = require('firebase-functions');
const {WebhookClient} = require('dialogflow-fulfillment');
const {Card, Suggestion} = require('dialogflow-fulfillment');

process.env.DEBUG = 'dialogflow:debug'; // enables lib debugging statements

exports.dialogflowFirebaseFulfillment = functions.https.onRequest((request, response) => {
  const agent = new WebhookClient({ request, response });
  console.log('Dialogflow Request headers: ' +
    JSON.stringify(request.headers));
  console.log('Dialogflow Request body: ' + JSON.stringify(request.body));

  function respuesta_reserva(agent){
    const fecha = new Date(agent.parameters.fecha_hora_reserva.date_time);
    const opcionesFecha = {year: 'numeric', month: 'long', day: 'numeric'};
    const opcionesHora = {hour: '2-digit', minute: '2-digit', timeZone:
      'Europe/Madrid'};
  }
});
```

```
agent.add(`Reserva para ${agent.parameters.numero_personas} personas,
${fecha.toLocaleDateString('es-ES', opcionesFecha)} a las
${fecha.toLocaleTimeString('es-ES', opcionesHora)}.`);
}

// Run the proper function handler based on the matched Dialogflow intent
name
let intentMap = new Map();
intentMap.set('reservar-mesa', respuesta_reserva);
agent.handleRequest(intentMap);
});
```

- package.json - añadimos la dependencia *full-icu* para poder formatear la fecha en español.

```
{
  "name": "dialogflowFirebaseFulfillment",
  "description": "This is the default fulfillment for a Dialogflow agents
using Cloud Functions for Firebase",
  "version": "0.0.1",
  "private": true,
  "license": "Apache Version 2.0",
  "author": "Google Inc.",
  "engines": {
    "node": "10"
  },
  "scripts": {
    "start": "firebase serve --only functions:dialogflowFirebaseFulfillment",
    "deploy": "firebase deploy --only
functions:dialogflowFirebaseFulfillment"
  },
  "dependencies": {
    "actions-on-google": "^2.2.0",
    "firebase-admin": "^5.13.1",
    "firebase-functions": "^2.0.2",
    "dialogflow": "^0.6.0",
    "dialogflow-fulfillment": "^0.5.0",
    "full-icu": "^1.3.0"
  }
}
```

Hacemos clic en **Deploy** para desplegar el código.

## Webhook


DISABLED ☐

Your web service will receive a POST request from Dialogflow in the form of the response to a user query matched by intents with webhook enabled. Be sure that your web service meets all the [webhook requirements](#) specific to the API version enabled in this agent.

## Inline Editor (Powered by Google Cloud Functions)

ENABLED ☒

Build and manage fulfillment directly in Dialogflow via Cloud Functions. [Docs](#)

 Newly created cloud functions now use Node.js 10 as runtime engine. Check [migration guide](#) for more details.


```
index.js package.json
1 'use strict';
2
3 const functions = require('firebase-functions');
4 const {WebhookClient} = require('dialogflow-fulfillment');
5 const {Card, Suggestion} = require('dialogflow-fulfillment');
6
7 process.env.DEBUG = 'dialogflow:debug'; // enables lib debugging statements
8
9 exports.dialogflowFirebaseFulfillment = functions.https.onRequest((request, response) => {
10   const agent = new WebhookClient({ request, response });
11   console.log('Dialogflow Request headers: ' + JSON.stringify(request.headers));
12   console.log('Dialogflow Request body: ' + JSON.stringify(request.body));
13
14   function respuesta_reserva(agent){
15
```


[View execution logs in the Google Cloud Console](#)



Last deployed on 09/08/2020 12:46


**DEPLOY**

Por último, navegamos a la [lista de funciones en Google Cloud](#), hacemos clic en *dialogflowFirebaseFulfillment* y luego en **Edit**. Añadimos la variable *NODE\_ICU\_DATA* con valor *node\_modules/full\_icu* para especificar que queremos usar el conjunto de datos entero (*full-icu*) para la internacionalización. Pinchamos **Next** y luego **Deploy** para desplegarlo.


 Cloud Functions


 Edit function

 Configuration —  Code

 **dialogflowFirebaseFulfillment**  
us-central1


**Trigger**

 **HTTP**


Trigger URL 

<https://us-central1-test-dialogflow-288908.cloudfunctions.net/dialogflowFirebase...>


EDIT

VARIABLES, NETWORKING AND ADVANCED SETTINGS 

ADVANCED **ENVIRONMENT VARIABLES** CONNECTIONS

Environment variables 

Name	Value
NODE_ICU_DATA	node_modules/full-icu

 ADD VARIABLE

Probamos el agente con la frase “reserva para dos personas hoy a las 14h” y observamos como ha cambiado el formato de la fecha.

**Agent**

---

USER SAYS

COPY CURL

reserva para dos personas hoy a las 14h

---

 DEFAULT RESPONSE

▼

Reserva para 2 personas, 8 de septiembre de 2020 a las 14:00. ¿Deseas continuar?

---

CONTEXTS

RESET CONTEXTS

reservar-mesa-followup

---

INTENT

reservar-mesa

---

ACTION

*Not available*

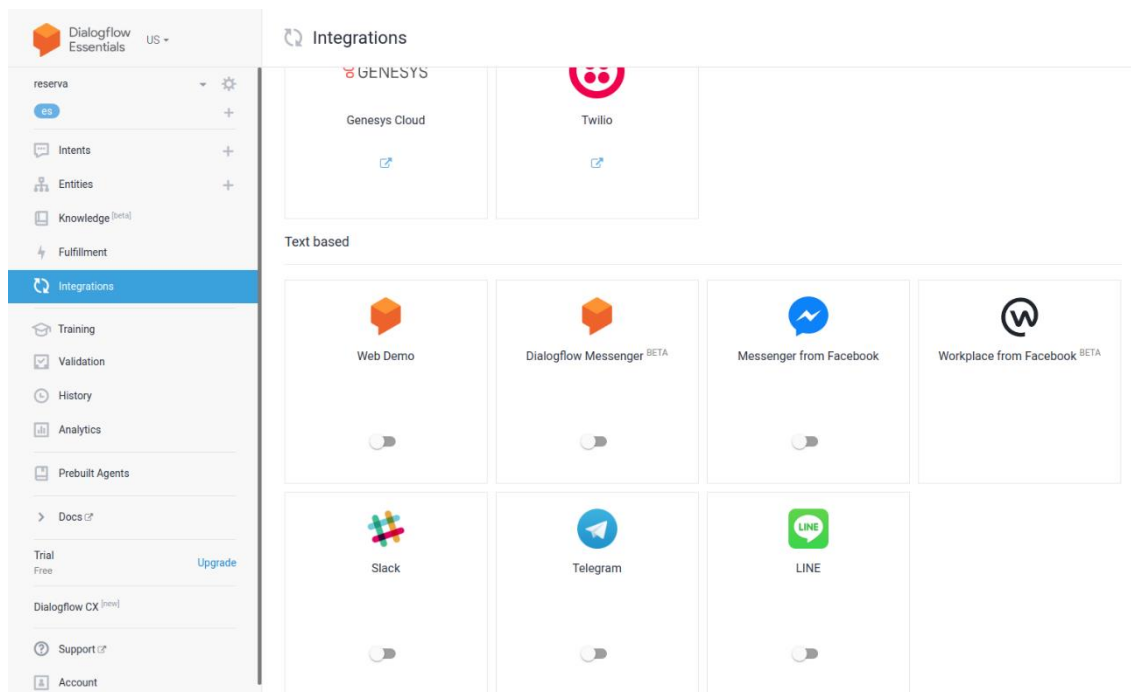
PARAMETER	VALUE
numero_personas	2
fecha_hora_reserva	{ "date_time": "2020-09-08T14:00:00+02:00" }

## 11. Interacciones

Se puede interactuar con el agente usando las integraciones que proporciona Google Dialogflow (Messenger, Telegram, Slack ...), pero también a través de la API.

## a. Integraciones

Las integraciones se encuentran en *Integrations*.



Cada integración tiene su propia configuración y se tiene que rellenar en la interfaz.



The screenshot shows the 'Integrations' panel in Dialogflow. The 'Facebook Messenger' integration is highlighted. The setup instructions are as follows:

**Facebook Messenger**  
An easier way to message.

Create and teach a conversational bot for Facebook Messenger.

After you design and test your Dialogflow agent, you can launch your Messenger bot

1. Get your Facebook Page Access Token and insert it in the field below.
2. Create your own Verify Token (can be any string).
3. Click 'START' below.
4. Use the Callback URL and Verify Token to create an event in the Facebook Messenger Webhook Setup.

[More in documentation.](#)

☐ Show old callback URL (more details [here](#))

Callback URL:

Verify Token:

Page Access Token:

Choose an environment to use with this integration.

Environment:

## b. API

Para poder usar la API tenemos que activar el [API de Dialogflow](#) en el proyecto.

### Register your application for Dialogflow API in Google Cloud Platform

Google Cloud Platform allows you to manage your application and monitor API usage.

#### Select a project where your application will be registered

You can use one project to manage all of your applications, or you can create a different project for each application.

test-dialogflow

También tenemos que crear una [cuenta de servicio](#) especificando el nombre y el ID de la cuenta.

Create service account

1 Service account details — 2 Grant this service account access to project (optional) — 3 Grant users access to this service account (optional)

**Service account details**

Service account name  
chatbot

Display name for this service account

Service account ID  
chatbot @test-dialogflow-288908.iam.gserviceaccount.com X ↺

Service account description  
Describe what this service account will do

CREATE CANCEL

Para la prueba le concedemos el Rol "Owner" aunque no es recomendable.

✓ Service account details — 2 Grant this service account access to project (optional) — 3 Grant users access to this service account (optional)

**Service account permissions (optional)**

Grant this service account access to test-dialogflow so that it has permission to complete specific actions on the resources in your project. [Learn more](#)

Role  
Owner

Condition  
[Add condition](#)

Full access to all resources.

+ ADD ANOTHER ROLE

CONTINUE CANCEL

Una vez creada la cuenta de servicio generamos la clave privada en formato JSON.

Service accounts + CREATE SERVICE ACCOUNT DELETE SHOW INFO PANEL

Service accounts for project "test-dialogflow"

A service account represents a Google Cloud service identity, such as code running on Compute Engine VMs, App Engine apps, or systems running outside Google. [Learn more about service accounts.](#)

Organization policies can be used to secure service accounts and block risky service account features, such as automatic IAM Grants, key creation/upload, or the creation of service accounts entirely. [Learn more about service account organization policies.](#)

Filter table	Email	Status	Name	Description	Key ID	Key creation date	Actions
	chatbot@test-dialogflow-288023.iam.gserviceaccount.com	✓	chatbot		No keys		Edit Disable <u>Create key</u> Delete

Esto activa la descarga automática de la clave privada.

Creamos un proyecto en Python e instalamos el paquete *dialogflow*.

```
pip install dialogflow
```

Creemos un fichero llamado cliente.py con el código de un pequeño cliente.

```
import os
import json
import uuid
import dialogflow_v2 as dialogflow

def establecer_conexion():
    sesion_cliente = dialogflow.SessionsClient()
    with open(os.environ['GOOGLE_APPLICATION_CREDENTIALS']) as credenciales:
        id_proyecto = json.load(credenciales)['project_id']

        sesion = sesion_cliente.session_path(id_proyecto, str(uuid.uuid4()))
        return sesion_cliente, sesion

if __name__ == '__main__':
    sesion_cliente, sesion = establecer_conexion()

    while True:
        text = input('Yo: ')
        if text == 'q':
            break

        text_input = dialogflow.types.TextInput(text=text, language_code='es-ES')
        query_input = dialogflow.types.QueryInput(text=text_input)
        respuesta = sesion_cliente.detect_intent(session=sesion,
        query_input=query_input)
        print(f'Chatbot: {respuesta.query_result.fulfillment_text}')
```

En el código se define la función *establecer\_conexion* que devuelve la sesión del cliente y la del usuario. En el *main* obtenemos la conexión, y en el bucle leemos desde la entrada estándar y enviamos el texto al agente. El agente devuelve la respuesta y la imprimimos.

Para la correcta función del cliente se tiene que pasar la ruta de la clave publica descargada.

```
GOOGLE_APPLICATION_CREDENTIALS=credenciales.json python cliente.py
```

Resultado:

```
Yo: menu del dia
Chatbot: Primero a elegir entre Sopa de lentejas y Crema de garbanzos con
sésamo. Segundo a elegir entre Pimientos rellenos de arroz y verduras y Pollo
al grill con salsa de mostaza.
```

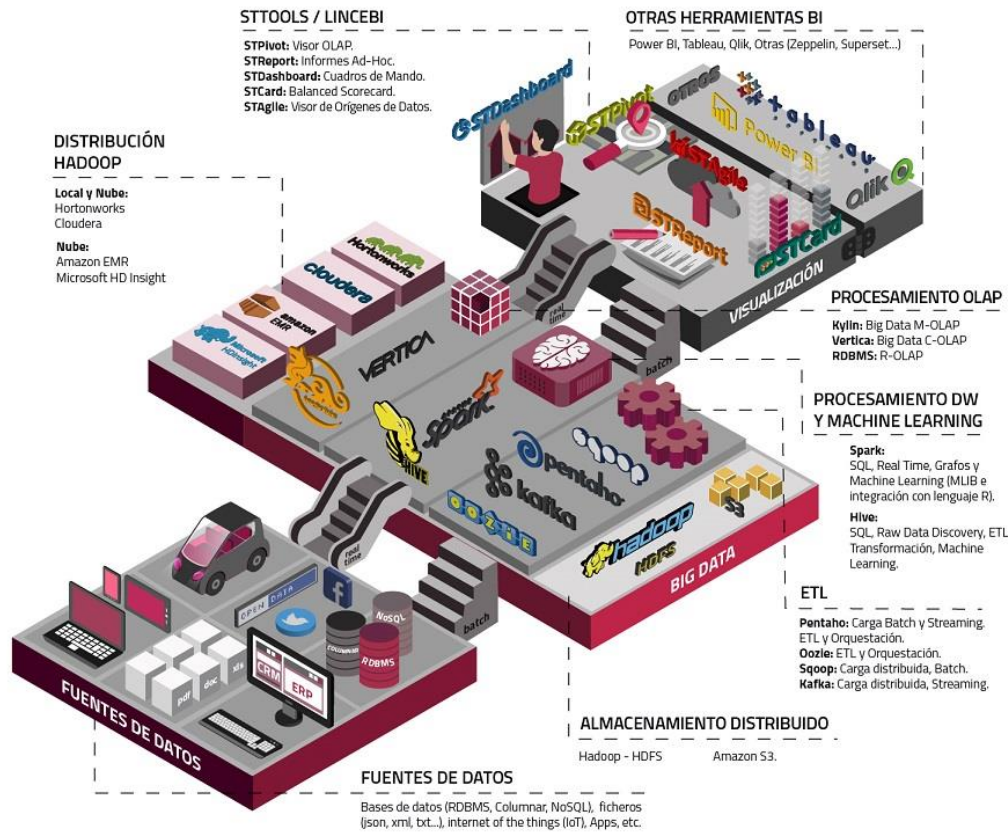
## 5. CONCLUSIONES

Google Dialogflow permite la creación de chatbots con comprensión del lenguaje natural. Puede analizar texto o audio y generar respuesta en formato texto o con voz sintética. Existen dos versiones, ES que sirve para crear agentes pequeños, medianos o modernamente complejos, y CX que sirve para crear agentes grandes o muy complejos. Son diferentes en cuanto al diseño, ES funciona con intents y contextos, CX con páginas y flujos. En la siguiente versión de este documento veremos Google Dialogflow CX.

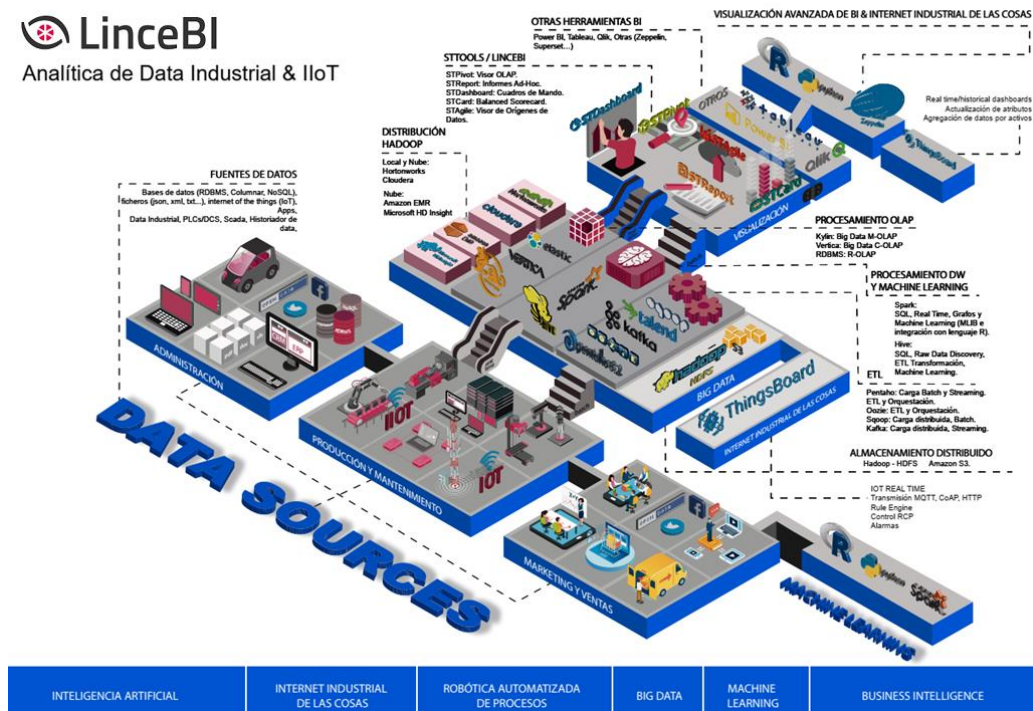
## 6. TECNOLOGÍAS

Recientemente, hemos sido nombrados Partners Certificados de Vertica, Talend, Microsoft, Snowflake, Kylligence, Pentaho, etc.





**LinceBI**  
Análítica de Data Industrial & IIoT



## 7. INFORMACIÓN SOBRE STRATEBI



**Stratebi** es una empresa española, con sede en Madrid y oficinas en Barcelona, Alicante y Sevilla, creada por un grupo de profesionales con amplia experiencia en sistemas de información, soluciones tecnológicas y procesos relacionados con soluciones de Open Source y de inteligencia de Negocio.

Esta experiencia, adquirida durante la participación en proyectos estratégicos en compañías de reconocido prestigio a nivel internacional, se ha puesto a disposición de nuestros clientes.

Somos **Partners Certificados en Microsoft PowerBI** con una dilatada experiencia

**Stratebi es la única empresa española que ha estado presente todos los Pentaho Developers celebrados en Europa** habiendo organizado el de España.

En Stratebi nos planteamos como **objetivo** dotar a las compañías e instituciones, de herramientas escalables y adaptadas a sus necesidades, que conformen una estrategia Business Intelligence capaz de rentabilizar la información disponible. Para ello, nos basamos en el desarrollo de soluciones de Inteligencia de Negocio, mediante tecnología Open Source.

Stratebi son **profesores y responsables de proyectos** del Master en Business Intelligence de la Universidad UOC, UCAM, EOI...

Los profesionales de Stratebi son los creadores y autores del primer weblog en español sobre el mundo del Business Intelligence, Data Warehouse, CRM, Dashboards, Scorecard y Open Source. Todobi.com

Stratebi es partner de las principales soluciones Analytics: Microsoft Power BI, Talend, Pentaho, Vertica, Snowflake, Kylogence, Cloudera...

**Todo Bi**, se ha convertido en una referencia para el conocimiento y divulgación del Business Intelligence en español.



## 8. OTROS

Trabajamos en los principales sectores y con algunas de las compañías y organizaciones más importantes de España.

### SECTOR PRIVADO



### SECTOR PÚBLICO



## 9. EJEMPLOS DE DESARROLLOS ANALYTICS

A continuación, se presentan **ejemplos de algunos screenshots** de cuadros de mando diseñados por Stratebi, con el fin de dar a conocer lo que se puede llegar a obtener, así como Demos Online en la web de Stratebi:

