**El Diseño Conversacional es un lenguaje de diseño basado en la conversación humana**.  
Para lograr un buen Diseño Conversacional se requieren disciplinas tales como:

* Diseño Interactivo
* Diseño Visual
* Diseño de animación
* Interfaces de voz
* Diseño de Audio
* Escritura UX
* ¿Qué es el diseño conversacional?  
  Es un lenguaje de diseño basado en la conversación humana.  
  Son buenos en entablar relaciones que se sientan más profundas y personales con nuestros usuarios.  
  Se involucran varias disciplinas: Interfaces de voz, diseño interactivo, la escritura de experiencia de usuario (ux writing).
* ¿Quién diseña las experiencias conversacionales?  
  El Diseñador de Experiencia de Usuario, que define el flujo, mejora la conversación, e incluye a todas las personas dentro del equipo de desarrolla. Es el que define las necesidades del usuario y las convierte en requerimientos.  
  Los usuarios están adoptando marcas que utilizan esta modalidad.
* ¿Por qué diseñar experiencias conversacionales?  
  Dentro del Google I/O del 2018 ya existían más de un millón de acciones disponibles dentro del Google Assistant. Hoy en día los usuarios realizan búsquedas a través de comandos de voz desde distintos dispositivos como Google Home, un celular o un smartwatch. Google Assistant no te limita a desplegar tus acciones únicamente dentro de su misma plataforma, también puedes extenderlas a Facebook Messenger, Slack o un desarrollo propio.
* ¿En qué dispositivos?  
  Headphones, speakers, teléfonos, smart tvs, etc.
* ¿En qué momento consumimos experiencias conversacionales?  
  Al despertar: buscamos conocer sobre noticias y la rutina del día  
  De camino al trabajo: Tratamos de ser más productivos aunque tengamos la vista y las manos ocupadas.  
  Descansando por la noche: Reproducimos música o series.
* ¿Qué cualidad tiene una buena interfaz conversacional?  
  Una buena interfaz conversacionales logra que los usuarios requieran menos entrenamiento para poder usarla.

Una experiencia conversacional no es adecuada para todos los casos de uso, funcionan bien si vamos a realizar una cita, pero son muy mala idea si decidimos buscar grandes cantidades de información.

Para ver si la implementación de una interfaz conversacional es adecuada necesitas verificar que tu caso de uso cumpla con la mayoría de los siguientes puntos:

* La conversación debe ser intuitiva, tus usuarios ya suelen tener conversaciones con otras personas acerca de la tarea o tema.
* La interacción de la conversación es corta.
* La conversación reduce el tiempo y esfuerzo que una UI en pantalla.
* Pueden llevar a cabo la tarea mientras hacen otras actividades ligeras.
* La conversación permite al usuario hablar libremente, no toca temas privados

TU MARCA

* Es importante levantar requerimientos de usuario, pero también es muy importante definir la marca , personalidad del bot.

1. Nombre y audiencia.
2. Valores de la marca.
3. Tono de voz de la marca.

**User Persona.**  
Quienes son? , cuales son sus necesidades? , como completan tareas el día de hoy? , que palabras usan para comunicarse ?, que circunstancias detonan las tareas en mis usuario?

**Se analizan los datos**  
Se tendrán que asumir algunas cosas para poder definir un perfil de usuario en base a los datos.

Perfil de usuario  
Si fuera una persona ¿Quien seria ?

sus motivaciones  
Que es lo que quiere lograr? objetivos y metas?

Ansiedades  
¿Cuales son sus limitaciones y miedos?

Su contexto  
Donde esta normalmente en su día a día? donde es mas común que consuma la experiencia conversacional?

Factores temporales  
En que momentos interactua con el asistente?

Suposiciones  
Que otros elementos de su día a día son específicos a su perfil

SYSTEM PERSONA

Con el *system persona* vamos a definir a nuestro asistente virtual, las características importantes son:

1. Su nombre y una breve descripción sobre él.
2. Su historia.
3. Su rol y estilo.
4. Su personalidad.
5. La forma y estilo en qué se comunicaría.

DIALOGOS DE PRUEBA

**¿Por que escribir diálogos de prueba?**  
Rápidamente nos dejan ver como se siente y se escucha la interacción que estamos diseñando.  
Nos centramos en la experiencia del usuario sin distracciones.

**Evitar sobre diseñar**  
**Empieza por conversación oral**  
Se recomienda diseñar para un dispositivo sin pantalla. Lograr un buen flujo es mas fácil si toda la información existe en los dialogos. A medida de que soportes mas tipos de dispositivos , puedes usar otros elementos gráficos como prompts y chips.

¿Qué es la UX Conversacional?  
Se podría decir que la UX Conversacional es la experiencia de usuario en las conversaciones que podrían ser por voz o texto entre un humano y un Bot con el uso de lenguaje natural para emular una conversación humana. Y que como toda conversación tiene sus flujos y regresiones que la dinamizan, o en el peor de los casos, que la limitan y friccionan. El unicornio de la “UX Conversacional” es conseguir que una conversación logre sentirse humana y natural pese a que una de las partes no lo sea.  
Se podría sugerir que la “Usabilidad Conversacional” se encarga de medir la calidad de la experiencia conversacional del usuario en una conversación entre éste y el Bot.

MEJORANDO NUESTRA CONVERSACION

Probar y mejor tu conversacion, por lo cual se puede utilizar una tecnica que es\*\* la tecnica del mago de OZ\*\*

* Los experimentos de Mago de Oz son el MVP de prototipos para voz.
* Sirve para simular el look and feel de la experiencia.
* En lugar de tener software tienes a una persona (el mago) simulando el comportamiento del System Persona.

Hay 2 tipos :

* Rapido y Sucio : Usa lo que tengas . Solo necesitas tus dialogos de prueba. Busca a alguien que no conozca tu proyecto y pideles que participen del rol.
* Estandar : Simula a tu persona usando el simulador de TTS de Actions on Gooogle . Descarga el audio y tenlo listo para reproducirlo a demanda. TIP : si el sonido no se reproduce adecuadamente puedes usar SSML

**Tips para experimientos:**

* Di tus dialogos en voz alta
* Graba tus sesiones
* Pide Retro

Para mejorar el dialogo hay que estar al pendiente de :

1. Detectar que la conversación sea natural,
2. Detectar si los usuarios están confundidos, saben que decir y que contestar al asistente
3. ¿tu usuario se expresa de forma inesperada?
4. ¿Mostro Señales de frustración e impaciencia?
5. Observa quien esta hablando mas.

ACTION EN GOOGLE Y DIALOGFLOW

* Actions en google es el framework de desarrollo para extender al Google assistant.
* DialogFlow es una herramienta que permite que nuestras aplicaciones entiendan el lenguaje natural.
  + Clasificación de intenciones.
  + Extracción de entidades.
  + Agentes: Frases de entrenamiento, parámetros y respuestas.

CREAR NUESTRO PRIMER INTENTO

Un intent es algo que el usuario está tratando de lograr, toma como entrada frases de entrenamiento y parámetros. El intent debe regresar una frase como respuesta.

Dialogflow ya cuenta con algunas entidades programadas para que las usemos dentro de nuestros intent.

Dentro de Dialogflow podemos generar nuestras propias entidades si es que no encontramos alguna que se acople a nuestras necesidades, solamente debemos definir las palabras clave y sus sinónimos.

Fulfillment nos permite realizar respuestas mediante código sobre nuestros intent.

// See https://github.com/dialogflow/dialogflow-fulfillment-nodejs

// for Dialogflow fulfillment library docs, samples, and to report

'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 ciudades = ['Bogotá', 'Ciudad de México']

const tematicas = ['Inteligencia Artificial', 'React', 'Firebase']

const siguienteLive = {

dia: '18 de febrero',

hora: '8 pm',

tema: 'Como convertirte en un desarrollador estrella'

}

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 obtenerCiudad(agent){

if(ciudades.includes(agent.parameters.ciudad)){

agent.add(`Genial! Te puedo ayudar a encontrar placticas

y talleres en tu ciudad o eventos en linea.

¿Por cual te gustaría empezar?`)

} else {

agent.add(`Oh! Aún no hay meetups en tu ciudad , pero el

siguiente Tiempo Live es el dia ${siguienteLive.dia} a las

${siguienteLive.hora} y el tema es ${siguienteLive.tema}`)

}

}

function seleccionDeTematica(agent){

agent.add(`Super! A mi tambien me encantan los retos.

Estos son los temas que se van a cubrir proximamente en tu

ciudad: ${tematicas.join(',')}. ¿Cual te interesa más?`)

}

function detalleTiempoLive(agent){

agent.add(`Oh! Aún no hay meetups en tu ciudad , pero el

siguiente Tiempo Live es el dia ${siguienteLive.dia} a las

${siguienteLive.hora} y el tema es ${siguienteLive.tema}`)

}

let intentMap = new Map();

intentMap.set('Obtener Ciudad', obtenerCiudad);

intentMap.set('Live', detalleTiempoLive);

intentMap.set('Talleres', seleccionDeTematica);

agent.handleRequest(intentMap);

});

INTENTS Y CONTEXTOS

Los contextos son la forma en que DialogFlow enlaza parámetros y variables de un intent a otro. Nos sirve para indicarle a DialogFlow la secuencia de nuestros intent.

Los intent de seguimiento o follow-up intents son la forma en que DialogFlow nos crea de manera automática intents sencillos que solamente buscan la contestación o confirmación de otro intent.

FULFILLMENT CON FIREBASE FUNCTIONS

1. Crea el directorio local del proyecto en tu pc
2. Baja el archivo CLI a la raíz de la carpeta del proyecto creado. Yo lo descargue de este enlace: <https://firebase.google.com/docs/cli/?hl=es-419#install-cli-windows>
3. Ejecutal el archivo descargado “firebase-tools-instant-win.exe” en modo administrador.
4. En la consola ejecuta el comando “firebase login”
5. Ejecuta el comando “firebase init”
6. Ejecuta el comando “firebase init funcionts”
7. Para ver los archivos de una carpeta colocas en la línea de comandos DIR (nombre del folder), así: “Dir Functions”
8. En mi caso descargue el CLI, debo descargar el nmp para seguir los comandos de ejemplo del docente. Para esto: ejecuto la siguiente línea de comando “npm install -g firebase-tools” (recuerda estar en la ruta de la carpeta functions), también puedes correr el comando del profe “npm install”
9. Ahora sí puedes ejecutar el comando del docente para instalar actions on google y dialogflow “npm install actions-on-google dialogflow-fulfillment --save”
10. modificar index, yo use atom como el docente.
11. Finalmente ejecutar el deploy: firebase deploy --only “functions:dialogflowFirebaseFulfillment”

INTENT DE CIERRE

Es importante indicarle a DialogFlow cuál será nuestro intent de cierre para que finalice la conversación siempre que caiga en este intent.

Siempre debes finalizar la conversación, nunca dejes un intent esperando una respuesta del usuario de manera innecesaria, recuerda que la privacidad del usuario es muy importante.

Dentro de las opciones que nos brinda el simulador de Actions on Google, encontramos la posibilidad de activar las pruebas en nuestro dispositivo móvil.

USANDO SLOT FILLING

Slot filling es una práctica que nos permite recibir más de una variable por intent. Podemos hacer uso de esta práctica para mejorar la experiencia conversacional y que nuestros usuarios no tengan que mandar una variable por cada intent.

Antes de desplegar nuestro Action debemos:

* Responder las preguntas de las capacidades de superficie, están van a determinar en qué dispositivos puede ser invocada tu Action.
* Verificar si la pronunciación del nombre de nuestro asistente es correcta, podemos modificarlo añadiéndole un texto a parte del nombre.
* Llenar los datos que nos pide el apartado de *Directory information*.

Es posible – y recomendable – desplegar versiones alpha y beta para realizar pruebas de tu Action.