Chatbots & OpenEdge

Marian Edu, Acorn IT
About me

Working with Progress since 1998
4GL & OO, AppServer
Architecture & Integration
Java, .NET
node.js, javascript, typescript
angular, nativescript

acorn.ro
A chatbot is a computer program or an artificial intelligence which conducts a conversation via auditory or textual methods. Such programs are often designed to convincingly simulate how a human would behave as a conversational partner, thereby passing the Turing test.

Wikipedia
Chatbot & AI

Natural Processing Language.

Semantic vs. Linguistic Model

Keyword matching.

Don’t really need to pass the Turing test.
Usage

Virtual Assistants.

Messaging Apps.

Organisation Apps/Websites.
Use Cases

Customer Support.

e-Commerce.

Marketing, news, finance, etc...
Skip Theory

Assistants

• Alexa Skills Kit
• Actions on Google

Messaging Integration.

OpenEdge Integration.
Assistants

Understanding User Input
NPL, AI, Intents, Workflow

- IBM Watson Assistant
- Amazon Alexa
- Google Assistant

Chatbots & OpenEdge
Assistants

- REST Services (PAAS – subscription)
- JSON payload
- Intent/Workflow definition (designer)
- API’s
Smart Speakers

Speech Recognition

• Amazon – Echo, Echo dot, Echo Plus, Echo Spot, Echo Auto…
• Google – Google Home, Mini, Max
• Facebook – Portal (Alexa)
Amazon Alexa

Alexa Console – Skills and Intents
Fulfilment – REST endpoint, Lambda
OpenEdge – REST services
## Alexa Console


### Sample Utterances (3)

<table>
<thead>
<tr>
<th>Utterance</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>What might a user say to invoke this intent?</td>
<td></td>
</tr>
<tr>
<td>What's the address of <code>{custName}</code></td>
<td></td>
</tr>
<tr>
<td>Where to ship goods for <code>{custName}</code></td>
<td></td>
</tr>
<tr>
<td>Where to find <code>{custName}</code></td>
<td></td>
</tr>
</tbody>
</table>

### Intent Slots (1)

<table>
<thead>
<tr>
<th>Slot Type</th>
<th>Actions</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>{custName}</code></td>
<td>Edit, Dialog, Delete</td>
</tr>
</tbody>
</table>

---

Chatbots & OpenEdge

[acorn.ro](http://acorn.ro)
Endpoint

Service Endpoint Type

Select how you will host your skill's service endpoint.

- AWS Lambda ARN (Recommended)
- HTTPS

Default Region

Enter URI...

Select SSL certificate type

North America

Enter URI...

Select SSL certificate type
Fulfilment

JSON In/JSON Out

AWS Lambda Function

REST web service
AWS Lambda

“AWS Lambda lets you run code without thinking about servers.”

```javascript
exports.handler = (event, context, callback) => {
};
```

Event: request, session
Context: succeed/fail
Intent name, Input values (slot)
Alexa Requests

LaunchRequest

IntentRequest

SessionEndedRequest

```json
{
  "version": "1.0",
  "session": {
    ...(session properties not shown)
  },
  "request": {
    "type": "LaunchRequest",
    "requestId": "string",
    "timestamp": "string"
  }
}
```
Handler APIs

- **OnSessionStarted** (event.session.new)
- **OnLaunch** (request.type = "LaunchRequest")
- **OnIntent** (request.type = "IntentRequest")
- **OnSessionEnded** (request.type = "SessionEndedRequest")
Response

- Synchronous
- `context.succeed / context.fail`
- Asynchronous
- `callback (error, response)`
Response

```json
{
  version: "1.0",
  sessionAttributes: {...}
  response: {...},
  reprompt: {...},
  shouldEndSession: true
}

outputSpeech: {
  type: "PlainText",
  text: output
},

card: {
  type: "Simple",
  title: title,
  content:
}
```
Google Assistant

DialogFlow Console – Agents and Intents
Fulfilment – REST endpoint, Cloud Function
OpenEdge – REST services
## Intents

### Error

#### Contexts

- Welcome
  - Add input context

- 1 Welcome 1 Error
  - Add output context

#### Events

#### Training phrases

- Add user expression
- it's 1234
- error 1234
Fulfillment

Webhook

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.

URL* 
https://esd.acorn.ro/google

BASIC AUTH
Enter username
Enter password

HEADERS
Enter key
Enter value

DOMAINS
Enable webhook for all domains
Fulfilment

JSON In/JSON Out

Cloud Function (Firebase)

REST web service

SDK: node.js, python
Fulfillment

Inline Editor (Powered by Cloud Functions for Firebase)

Build and manage fulfillment directly in Dialogflow via Cloud Functions for Firebase.

```javascript
// See https://github.com/dialogflow/dialogflow-fulfillment-nodejs
// for Dialogflow fulfillment library docs, samples, and to report issues
'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(async (request, context) => {
  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 welcome(agent) {
    agent.add('Welcome to my agent!');
  }

  function fallback(agent) {
    agent.add('I didn't understand');
    agent.add('I'm sorry, can you try again?');
  }

  try {
    await functions.httpsCallable('dialogflowAgent')(agent);
  } catch (error) {
    console.log('Exception: ' + error);
    fallback(agent);
  }

  return agent.fulfill();
});
```
“Google Cloud Functions is a lightweight, event-based, asynchronous compute solution that allows you to create small, single-purpose functions which respond to cloud events, without the need to manage a server or a runtime environment.”

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

exports.dialogflowFirebaseFulfillment = functions.https.onRequest((request, response) => {
  const agent = new WebhookClient({ request, response });

  function welcome(agent) {
    agent.add(`Welcome to my agent!`);
  }

  let intentMap = new Map();
  intentMap.set('Default Welcome Intent', welcome);
  agent.handleRequest(intentMap);
});
{
  "lang": "en",
  "result": {
    "resolvedQuery": "error 2089",
    "speech": "",
    "action": "get_progress_error",
    "parameters": {
      "errorno": 2089
    },
    "contexts": [
      {
        "name": "error",
        "parameters": {
          "errorno": 2089,
        },
        "lifespan": 1
      }
    ]
  }
}
DialogflowApp = require('actions-on-google').DialogflowApp;

var dialogApp = new DialogflowApp(
    
    request,
    
    response

});

// build an action map, which maps intent names to functions
let actionMap = new Map();
actionMap.set(PRODUCT_ACTION, getProductInfo);
actionMap.set(PROERROR_ACTION, getProgressError);

dialogApp.handleRequest(actionMap);
var botResponse = dialogApp.buildRichResponse()
    .addSimpleResponse('Hello')
    .addBasicCard(
        dialogApp.buildBasicCard('Card')
            .setTitle()
    );

dialogApp.ask(botResponse);
"speech": "Hello world",
"contextOut": [
{
"name": "_actions_on_google_",
"parameters": {...}
}
],
"data": {
"google": {
"expect_user_response": true,
"rich_response": {
"items": [...]
}
## Integration

<table>
<thead>
<tr>
<th></th>
<th>Web Demo</th>
<th>Facebook Messenger</th>
<th>Dialogflow Phone Gateway</th>
<th>Slack</th>
</tr>
</thead>
<tbody>
<tr>
<td>Viber</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Twitter</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Twilio IP</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Twilio (Text messaging)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Skype</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Tropo (Text messaging)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Telegram</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>kik</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LINE</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Cisco Spark</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Amazon Alexa</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Microsoft Cortana</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
OpenEdge Integration

REST

What kind of flavour?

Library (API/SDK)

akera.io – we’re just happy with node.js
Facebook Integration

Facebook Messenger

Need one Facebook Page/App

Page Access Token

Webhooks Configuration