Java Report

Analysis of Chatbot Development Platforms

GROUP 8

Jiabin Chen

Min Li

Xindi Lan

Zipei Xiao

1. Introduction

The report is to analyse three popular chatbot development platforms. By analysing the business development and technology, three mainstream platforms are compared for their pros and cons. Also, based on the research about current chatbot development platforms, the best approach to develop a chatbot in Java is discussed.

2. IBM Watson

2.1 Platform Analysis

2.1.1 Introduction

IBM Watson is designed for various fields and built on the system that is leading in conversational computing technology today. It helps companies provide a personalised experience for their customers and provide the improvement that keeps each conversation better than last time.

The IBM Watson is an AI platform. It is an integration of much cutting-edge technology, such as machine learning, natural language processing, and integrated dialog tools. All of these technologies are applied to create a vivid conversation between business owners and their users. Thus, when the clients ask for an AI chatbot that can self-learn, and deal with complex and difficult conversation, IBM Watson is the preferred choice.

IBM Watson is the mainstream bot-building platform for global businesses, which can be served in different fields with different types of clients. Primarily, it has more advantages on health care, education, finance, transportation, and energy field.

2.1.2 Technology

First, IBM Watson is a system using machine learning technology, which is trained on data instead of only predefined rules. Second, it is built on a neural network, which is about one million words from Wikipedia. Third, IBM Watson adopts advanced natural language processing capabilities, including speech recognition, speech segmentation, morphological segmentation, part-of-speech tagging, parsing, sentence breaking, word segmentation, terminology extraction, and information retrieval ("Chatbot Development With IBM Watson - DZone AI", 2018).

The data processing of IBM Watson is as below. Users start with sending a message to the chatbot on the user interface. The user interface can be a software window, an application on mobile, or a physical robot at home. Then, the input message is sent to the IBM Watson Assistant service. A workspace is connected with it, and dialog flow and training data are included in this workspace. The IBM Watson Assistant service interprets and decodes the user input, and it also collects additional information if it determines it indeed needs. In a further step, the application had an interaction between IBM Watson and its back-end systems, according to the interpreted user's intent and collected additional information. Finally, it ends up with a response and sends back to users.

2.2 Pros & Cons

2.2.1 Pros

1. Integration with other IBM services: It can take advantages of other IBM services to provide high-quality content, such as by Watson Discovery Service for more complex responses.
2. Connection with chatting platform: It cooperates with various social media and communication platform, including Slack, Facebook and Twilio.
3. Guide for further training: It provides suggestions on what to further train, according to the questions which are frequently asked but do not have answers yet.
4. Multiple programming languages: It supports a variety of programming platforms, including Node SDK, Java SDK, Python SDK, iOS SDK, Unity SDK.

2.2.2 Cons

1. Limited languages: Only 13 languages are supported for IBM Watson now, not including some languages that are widely used like Hindi.
2. Coding experience needed: Watson is a coding chatting platform. The chatbot can be combined with other applications under different programming platform. In this way, it requires prerequisite coding experience for chatbot developers.
3. Relatively high service fee: Watson has three service plans, which are Lite (free 10,000 API Calls per Month), Standard (Unlimited API Calls, $0.0025 per API call), Premium (prices are different case by case). The service fee is higher than many other chatbot development platforms.

3. Microsoft Bot Framework

3.1 Platform Analysis

3.1.1 Introduction

The Microsoft Bot Framework is a powerful set of services, tools, and SDKs that allows developers to build and connect intelligent bots. By using the Microsoft Bot Framework, developers can create, integrate, and manage a wide variety of bots for users to interact with (“Microsoft Bot Framework,” n.d.). The tools and services provided by Microsoft Bot Framework can help developers to reduce the amount of code and work more efficiently.

3.1.2 Technology

The Bot Framework has three main components, which are the Bot Builder SDKs, Bot Connector, Bot Directory. The Bot Builder SDKs consist of Bot Application, Bot Controller and Bot Dialog templates, which can manage the conversations with users. The Bot Connector can help to quickly publish services to various channels, such as Skype, Slack, Office 365 mail, etc. and allow users to use services on these channels. The Bot Directory is a Bot's store, where people can find the various bots and also publish their bots (Chan, 2017).

Bot Connector uses REST API and JSON over secure protocol HTTPS, which allows a bot to exchange messages with channels available in Microsoft Bot Framework. When a user sends a message, the Bot Connector sends a POST request to the endpoint. When bot gets a JSON object, the information from it can be used to create a response to the user.

Bot Builder SDKs support C# and Node.js programming languages and can quickly generate an ASP.NET and Node.js backend service. Also, Bot Builder SDKs can manage the conversations with users through the use of services like Dialog and FormFlow. The Bot Application templates provided by Bot Builder SDKs contains a simple project with all of the components for a simple bot, which includes a POST method to accept messages and a dialog builder to generate a response (Sannikova, 2018).

3.2 Pros & Cons

3.2.1 Pros

1. Accessible APIs: Microsoft Bot Framework allows to incorporate LUIS for natural language understanding, Cortana for voice, and the Bing APIs for search (Davydova, 2017).
2. Open Source: Microsoft Bot Framework is open source and available to all on Github.
3. Multiple Languages: Microsoft Bot Framework can automatically translate more than 30 languages, such as Afrikaans, Arabic, Bulgarian, Catalan, Chinese, etc.
4. Low Cost: Most of the components of Microsoft Bot Framework are freely downloadable development tools, so the cost of using this framework is very low.
5. Multiple Computer Languages: Bot Builder SDKs support for multiple computer languages including C#, Python, Node.js, and Android.

3.2.2 Cons

1. Limited Platforms: The platforms for development are only limited to Node.Js and C# platforms.
2. Lack of Flow: Microsoft Bot Framework cannot do like Google’s Dialog flow to help users interpret and make use of intents by designing the flow of conversation.

4. Chatfuel

4.1 Platform Analysis

4.1.1 Introduction

Chatfuel is a tool that helps media and brands create chatbots on instant messaging software. Without any technical foundation, users can create their chatbots based on AI. Chatfuel's AI engine works as users use it, helping them search for more information. Meanwhile, it can also provide personalised recommendations to users. As the official data shows, Chatfuel has provided more than 100 thousand chatbots, serving over 5 million global users ("Chatfuel About us," n.d.).

4.1.2 Technology

Without coding, users can start the chatbots. What they need to do is to click the button to edit their bots. They can add contents by using text cards, gallery cards or plugins.

1. Feature: The interface is a basic building block that consists of several text cards and gallery cards. A text card can only contain text with a length up to 640 characters. And a gallery card is a special UI card that gives users the ability to show up to 9 items (image, URL etc.) in a slider.
2. Plugins: Chatfuel mainly uses many plugins tools to make the chatbots much more powerful. Bots can connect to the existing content or API through them. Here are three main plugins:
   (1) Google Search: The Google site search plugin allows users to display target content from websites. It also supports parameters provided by the User Input plugin. That is great to show customers the exact content they want.
   (2) RSS Import: The RSS plugin allows users to show content from RSS feeds beautifully in the bot. Many blogging platforms like WordPress offer RSS feed support out of the box, and this plugin makes it easy to show new articles in the chatbot.
   (3) JSON API: JSON is an open-standard file format that uses human-readable text to transmit data objects. They can apply user attributes in either the URL or the USER ATTRIBUTES field. With POST request, user attributes will be sent in a standard way. With GET request, they will be added to the URL as GET parameters. And Chatfuel makes a simple GET or POST query, then renders the response straight to the customers ("Plugins JSON API," n.d.).

4.2 Pros & Cons

4.2.1 Pros

1. A simple solution to building chatbots: Users do not need to spend their time struggling with code. They can just click the button to edit their chatbots.
2. Less risk: Chatfuel has less risk regarding getting the value users are looking for (Dana, 2018).
3. Many kinds of plugins: Chatfuel makes use of many kinds of plugins such as Google search, JSON API and RSS import, to make chatbots have more functions.
4. Language support: Chatfuel support 50 different languages in the world including Chinese, English, Japanese Spanish etc. to make it convenient for people all over the world to use.
5. Free to use: Chatfuel is free for all people. Users don't need to pay for special products and services.

4.2.2 Cons

1. Poor support for different conversation structure: The support for branching, variable operations and non-linear conversations are poor or absent.
2. Lack of NLP: Chatfuel does not have NLP and cannot be trained to recognise meanings or typos (Eva, 2016).
3. Poor support for files: Chatfuel does not support import or export files.

5. Best Approach to develop a chatbot in JAVA

5.1 Brief introduction of general steps needed

Nowadays, many chatbot platforms use Java as their programming languages, such as IBM Watson Conversation Service, Pandorabots, Api.ai, and so on. Since chatbot simulates how humans are chatting, it needs to understand the grammar of human languages. That is, every chatbot needs to use NLP (Natural Language Processing) to interpret words input and then output useful information.

The chatbot needs the help of AI or machine learning to "understand" human language. And developers have three different options:
1. Natural Language Software Agent: developers are not expected to have any machine learning knowledge if they choose to use it. Those are pre-written templates for developers to upload data/information to train the chatbot. Famous Natural language software agents are AIML, Jbuddy Bot framework, gaelyk, etc.
2. NLP (Natural Language Processing) or NLU (Natural Language Understanding) Engine: There are two kinds of NLP: Rule-based and statistical NLP. NLP will analyse human language based on its grammar, sentence structure and so on. It is a system that handles end-to-end interactions between machines and humans in the preferred language of the human (HowToDoInJava, 2018). NLU is a subset of NLP, which helps in parsing unstructured inputs. NPL engines can be categorised into five categories: Agents, Entities, Intents, Actions, and Contexts. Some good NLP Platforms for Java are Stanford CoreNLP and Apache OpenNLP.
3. Machine Learning: build your own NLP/NLU by using Machine Learning models, such as a Retrieval Based Model and a Generative Model.

Since most developers do not have enough machine learning knowledge, we assume that we will use for example AIML to create natural language software agents. However, AIML is an XML dialect, so programmers need API to connect AIML to Java. Fortunately, Java has some libraries for this connection.

Therefore, the general steps to build up a chatbot in Java is:
1. Choose a good natural language software agent.
2. Write code in or merely upload data to the natural language software agent to train the machine.
3. Use library in Java as an API to connect to the natural language software agent.
4. Write code in Java to build up a window for users to input their questions and transfer information to the natural language software agent (front-end).

According to the steps mentioned above, the differences in various methods to build up a chatbot in Java is mainly in:
1. Natural language software agent
2. API (Java libraries)

The following part will analyse what the best choice for each step is and why it is the best choice.

5.2 Best solution

5.2.1 Best natural language software agent

We believe that the best natural language software agent is AIML. It is not because that we are required to use AIML for this Java group project. AIML is truly the most convenient way to train a machine. It is based on XML language, which is simple to learn and read. More importantly, you can merely create a new .aiml file in the AIMLl directory of the chatbot you build to add custom patterns, i.e., more intelligence in the interactions of chatbot. In other words, AIML "training" is not based on writing Java codes but instead based on monitoring the chat logs, which consist of question-answer pairs. When chatbot output inappropriate answers, developers can enter correct answer manually. If developers use gaelyk to train you chatbot for example, which is launched in Google, developers need to write complex codes to train it. That is, they cannot update questions and answers by simply upload a file. Moreover, pre-authored AIML sets are available (including the infoTabby AIML set designed for public and academic library use).

5.2.2 Best Java library (API)

We believe that the best Java library to use for connection to AIML is program-ab (https://code.google.com/p/program-ab/) hosted on the google-code repository. As mentioned above, AIML files are just like a type of database that only holds question and answer. And you do not want your chatbot merely pull the correct data from the database. It should also translate the answer into sentences that humans can understand. That's the job of an AIML interpreter, i.e., data retrieval script/code/program, such as Program AB. One advantage of Program AB is that it uses the newer, more feature-rich, and more flexible AIML 2.0 standard, whose learning curve is also a bit steeper. Another advantage of Program AB is that it allows developers to extend AIML with new custom tags easily. Developers need not write pairs of question and answers in AIML anymore. They can write codes in Java to manage the AIML files.

In conclusion, the best way to build a chatbot in Java is using AIML and Program AB together to train chatbot.

References

Chatbot Development With IBM Watson - DZone AI. (2018). Retrieved from https://dzone.com/articles/chatbot-development-with-ibm-watson

Chatfuel About us. (n.d.). Retrieved from https://chatfuel.com/about-us.html

Chan, D. (2017, August 7). How to use Microsoft's Bot Framework, LUIS, QnA Maker to make a simple dialogue robot [Blog post]. Retrieved from http://www.cnblogs.com/DarrenChan/p/7301380.html#_label0

Dana, T. (2018). ManyChat vs Chatfuel an In-Depth Review & Comparison. Retrieved from https://www.thinktuitive.com/comparison-manychat-vs-chatfuel/

Davydova, O. (2017). 25 chatbot platforms: A comparative table. Retrieved from https://chatbotsjournal.com/25-chatbot-platforms-a-comparative-table-aeefc932eaff

Eva, D. (2016). Bot Wars: Wit.ai vs Chatfuel. Retrieved from https://chatbotsmagazine.com/bot-wars-wit-ai-vs-chatfuel-5e978070b9e

HowToDoInJava, “Java chatbot example using aiml library”. Retrieved from https://howtodoinjava.com/ai/java-aiml-chatbot-example/

IBM Watson. (2018). Retrieved from https://www.ibm.com/watson/

Microsoft bot framework and conversation as a platform. (n.d.). Retrieved from https://www.edx.org/course/conversation-as-a-platform-with-the-microsoft-bot-framework

Nath, J. (2018). How to Identify the Right Platform for Your Chatbot. Retrieved from https://chatbotsmagazine.com/identifying-the-platform-for-your-chatbot-312e46ef9e8a

Plugins JSON API. (n.d.). Retrieved from https://help.chatfuel.com/facebook-messenger/plugins/json-plugin/

Sannikova, S. (2018). *Chatbot implementation with Microsoft Bot Framework.* Retrieved from https://www.theseus.fi/bitstream/handle/10024/142561/thesis.pdf

Watson Assistant v1 - IBM Cloud API Docs. (2018). Retrieved from https://console.bluemix.net/apidocs/assistant#introduction