Conversational User Experience at Workplace

Abhishek Konduri
Computer Software Engineering
San Jose State University
San Jose, USA
abhishek.konduri@sjsu.edu

Mayur Barge
Computer Software Engineering
San Jose State University
San Jose, USA
mayur.barge@sjsu.edu

Shubham Sandeep Sand
Computer Software Engineering
San Jose State University
San Jose, USA
shubhamsandeep.sand@sjsu.edu

Varun Jain
Computer Software Engineering
San Jose State University
San Jose, USA
varun.jain@sjsu.edu

***Abstract* — There are various tools used at software companies viz. version control system for managing source code, CI/CD tools for integration and deployment, agile management system and log aggregation tool. Sometimes it becomes arduous task to access data from various services. Developer finds it difficult to get information from various resources. This could be because of multiple human dependencies or multiple scattered system. A silo is built among the developers because of accessing various information from different tools. We can avoid this situation by building an integrated system with different service on a single and easily accessible platform. All the segregated information can be made available on a unified and ubiquitous system like your virtual assistant over mobile devices and computers.**

Keywords—Jenkins, JIRA, GitHub, Splunk, Google Dialogflow, Firebase, Conversational User Experience, Google Assistant, agile management, log aggregation, version control, Slack.

I. Introduction

In growing era of software development, projects tend to go on being complex over time with new software tools being used for every aspect of the software development process starting from using tools for software development lifecycle, version control, modeling continuous integration and continuous deployment and finally log analysis and its aggregation. One major challenge is to integrate these partite tools on one platform and thus make it available to cross-functional teams across various domains in a software company. There is a prevalent problem that certain software tools are difficult and complex to use to teams who don't have expertise in these tools and they are dependent on other teams to convey and perform necessary transactions on these tools.

To help a user interact more easily with these tools, we developed an interactive application that provides an abstraction to various underlying software development tools using a conversational interface developed using Google's "Dialogflow". The conversational user interface understands natural language sentences and expressions and adjusts keywords and criteria based on the content and the context of the sentences. Furthermore, such a conversational system can be integrated into existing company infrastructure for communication such as "Slack" or an automated phone service.

Dialogflow understand the human natural spoken language and process it to construct a conversational system. Dialogflow is powered by Google Machine learning framework which allows to extract user's intent during conversations and extract necessary parameters required by the underlying framework to query various software tools using conversational interface. It also comes up with various additional features including automatic spelling corrections and user sentiment analysis for each of the user query.

The Fulfillment service of the Dialogflow allows integration of JIRA, GitHub, Jenkins, and Splunk tools by connecting to the exposed application programming interfaces provided by each of these software tools.

The findings and contributions to this project include: 1) Building an N-module conversational agent that supports interaction across text and voice modalities with a common Node JS backend, 2) an exploration of the value of our DialogFlow agent for workplace efficiency and eliminating cross domain silo's, and 3) findings from a controlled deployment, showing that despite technical intricacies of the conversational interface, it has the potential to be used casually and allow a more engaging and interactive conversation with software tools.

II. Design and implementation

While building the application, the major concern was to process and understand natural language sentences and expression and to adjusts keywords and criteria based on the content and the context of the sentence. Dialogflow understand the human natural spoken language and process it to construct a conversational system. Dialogflow is powered by Google ML technology which allows to extract user's intent during conversations and extract necessary parameters. Agents are known to be Natural Language Understanding modules. Agents translate the regular text or spoken sentences into a request and initiate an Intent which can trigger an Action. An Intent is a way you to interact with the system. For example, all training phrases related to weather will be written under a single intent, let us say "Weather Status", which can be "Tell me about the weather" or "Will it rain today" or "How hot will be the day" is written under a relevant Intent. When the Dialogflow process the incoming speech or text instruction from a user, it matches with training phrases of all the intents. When a match for an intent is found, the functionality written inside executes.

Now, the agent will analyze the natural language instruction and will collect the Parameters from it. Parameters are dynamic values in training phrases, which can change the static execution of an intent to a dynamic one. For example, “What is the weather for *today*”, “What will be the weather for *28th* of this month”. Here, the task of an intent is to get weather information, but parameters are changing the day for which it wants the information. Parameters can be of two type - System Defined or User Defined knowns as Entity. System-defined parameters such as system date, time, number. An entity is used when a user wants to declare its own variable/parameters instead of system predefined ones.

A response of an intent can be written in an intent itself or we can program the response we want in fulfillment section where the actual Node.js code is written which hits the REST API’s.

The fulfillment is hosted on Firebase by Google Cloud Platform, allowing easy access to other projects running on Google Cloud Platform. In fulfillment, the interaction with different services like JIRA, GitHub, Splunk, and Jenkins comes into the part. Here the Node.js hits the API’s of these services and the retrieved information is sent back to the agent which transfer it to the incoming intent and send back to the user.

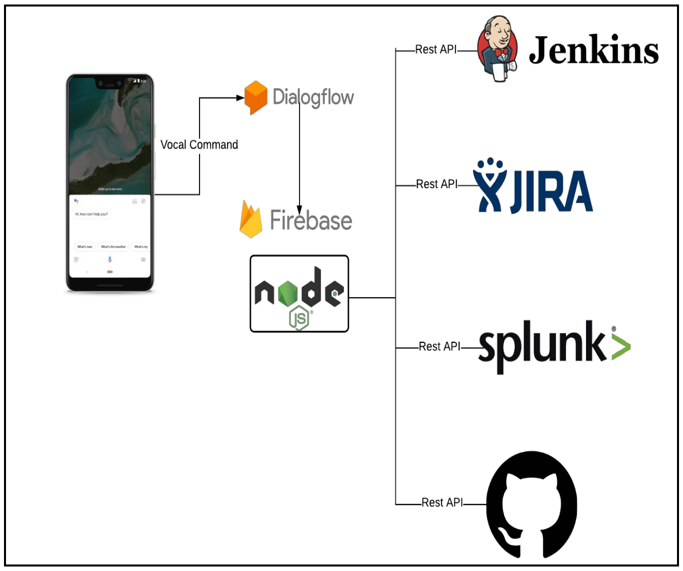
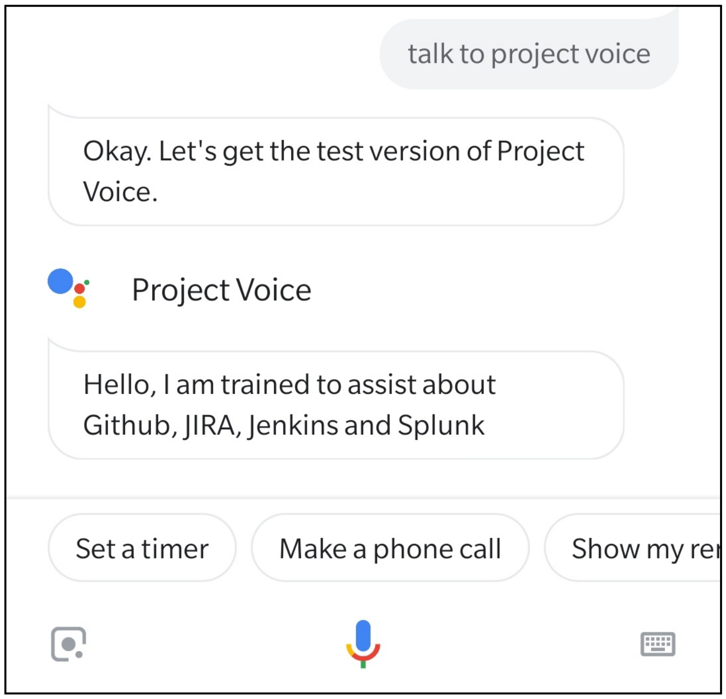


Figure 1. System Architecture

The Google Dialogflow comes with a very flexible list of 3rd party tools to which we can integrate with our agent. The most commonly used is Google assistant. Any user who is part of a project can log in google assistant with his/her credentials and access the project by saying the initial command to initiate the agent (“Talk to Project Voice” in our case) which makes it a secure system access.



talk to project voice

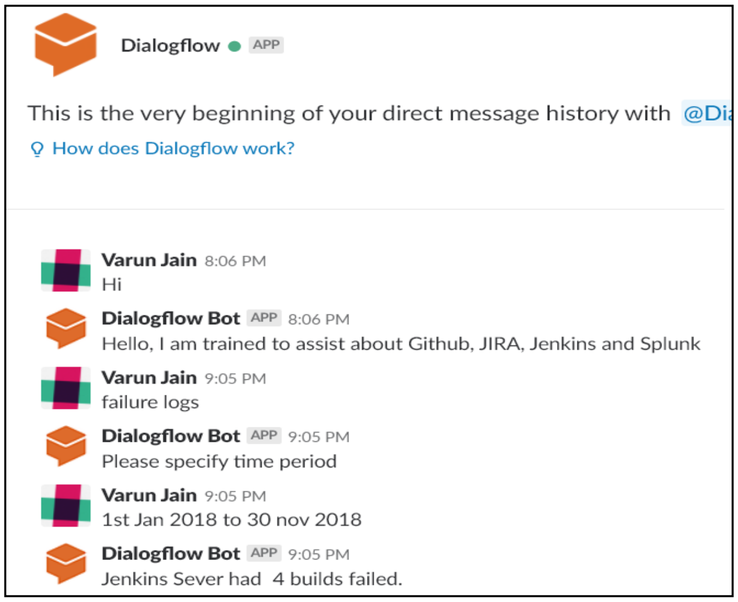
Okay. Let's get the test version of Project Voice.

Project Voice

Hello, I am trained to assist about Github, JIRA, Jenkins and Splunk

Figure 2 : Project Voice on Google Assistant

Google Assistant is widely used and easily accessible. However, to extend the usability and ubiquitousness of our project, we integrated it with Slack, popularly used in the software industry as team collaboration tools that bring team's communication and files in one place.



Dialogflow APP

This is the very beginning of your direct message history with @Di

How does Dialogflow work?

Varun Jain 8:06 PM
Hi

Dialogflow Bot APP 8:06 PM
Hello, I am trained to assist about Github, JIRA, Jenkins and Splunk

Varun Jain 9:05 PM
failure logs

Dialogflow Bot APP 9:05 PM
Please specify time period

Varun Jain 9:05 PM
1st Jan 2018 to 30 nov 2018

Dialogflow Bot APP 9:05 PM
Jenkins Sever had 4 builds failed.

Figure 3. Integration with Slack

III. USE CASES

The field of application for our interaction with the conversational interface can be defined by the following use cases:

1) JIRA: Used for Software Development Life Cycle Management and congruity to agile development. Our conversational interface is used to integrate with JIRA to support following functionalities:

- To assign user stories to an epic
- Move user stories from the current sprint to backlog or different sprint
- Change the status of user stories between "To Do", "In Progress" and "Done".
- Show number of stories assigned to a user.

2) GitHub: GitHub is a version control and distributed source code management system is usually the primary mechanism to be used in most of the software companies. The conversational interface integrates the GitHub application programming interface to support frequent tasks usually done by programmers including:

- Find the number of issues or pull requests in a repository and details like title and assignee
- In an organization, a project manager can access any number of latest commits and user responsible, for any repository
- To find the maximum commits done for any repository in an organization
- Identify the highest contributor in a repository

3) Jenkins: It is an open source automation server used for continuous integration and continuous deployment. Following functionalities are supported by our conversational Dialogflow interface:

- Trigger a build from GitHub Source code.
- Identify failed builds and re-trigger them.
- Showing no of slave machines configured.

4) Splunk: A tool used for log analysis and aggregation of machine-generated big data collected from various sources. The conversational interface integrated Splunk to include following log analysis queries:

- Getting the status of Splunk app and number of applications running on it.
- Agents configured to listen to various servers generating log data.
- Number of Jenkins builds failed between the user provided time period.

IV. CONCLUSION AND FUTURE WORK

In the above paper, we have eliminated these problems and saved valuable time for many developers and made work easy and portable, we have come up with an innovative idea of building conversational user experience on mobile for these tools. We have developed a centralized system to interact with all these tools and retrieved basic information and command simple tasks which will enhance developer experience. A developer won't have to anymore go anywhere to get updates. All the updates and information is now just one voice away. "Talk to Project Voice".

Now that we are successfully able to integrate Dialogflow to above mentioned tools, it can be improved further to assist for more complex scenarios. Another aspect the project can be improved upon is deployment of application in an enterprise environment with embedded security. The system can be extended to integrate with more platforms apart from Google Assistant and Slack like Cortana.

ACKNOWLEDGMENT

We are grateful to Prof. Rakesh Ranjan and are thankful for guiding us pushing us in exploring new ideas. We would like to take this opportunity to express our profound gratitude to him not only for his academic guidance but also for his personal interest in our project and constant support throughout the project. The success of the present work is mainly due to his valuable guidance, suggestions, astute judgment, constructive criticism and an eye for perfection. He always answered a myriad of my doubts with prodigious patience, never letting us feel that we are novices by always lending an ear to our views, appreciating and improving them and by giving us a free hand in our project.

Finally, we are grateful to our Institution and colleagues whose constant encouragement served to renew our spirit, refocus our attention and energy and helped us in carrying out this work

REFERENCES

[1] HTTPS://DIALOGFLOW.COM/DOCS/GETTING-STARTED

[2] https://developer.atlassian.com/cloud/jira/software/rest/

[3] https://developer.github.com/v3/

[4]https://wiki.jenkins.io/display/JENKINS/Remote+access+API

[5] http://dev.splunk.com/javascript