Smart-bot for Business Intelligence

Vikrant Dhimate, Divya Nambiar, Rupesh Shantamurty

SDCG

vikrant.mah.dhimate@hpe.com, dnambiar@hpe.com, rupeshs@hpe.com

Abstract

With the current adoption of CICD (Continuous Integration & Continuous Deployment) there is huge requirement to correlatethe software development stages to product management. A lot of times the link between the two never gets established leading to delays in resolving customer issues and duplication of efforts. This paper presents chat-bot to implement business intelligence solutions for better management of project development and better support for services organizations. The solution enables intelligent report generation for support folks by integrating workflow and product management tools like Rally, Jira, QuIX, etc. The solution will help development and support teams by providing easy access, powerful search and intelligent reporting that enables a simple way to retrieve required information along with filtering capabilities using natural language. The tool can be extended further to work with all the product lines of HPE.

Problem statement

Project management softwares like Rally keep records of various workflow and development activities like defects, user stories, features, etc. Though software maintains data, it becomes cumbersome task to retrieve data effectively. Our objective is to proactively let users know whether a change request or enhancement request already exists for any failure in product functionality. For example if there are test failures, engineers perform initial triage and create change or enhancement requests to fix it. If multiple engineers hit the same issue and if they do not know about the open issues in the same area then they end up spending equal amount of time in triaging. This leads to duplicate efforts. The same situation holds true for support organizations when multiple customers are hit by the same set of issues. This results in non-optimized usage of resources, lesser productivity and customer dissatisfaction.

Hence there is a huge need to integrate and share information across various product management and support tools.

Our solution

The current advancement in the chat-bot frameworks and their integration with AI frameworks help us to solve the problem stated above. The AI frameworks are used to mine relevant data from various sources and then correlate them. The chat-bots help to interact with humans and present the mined data in a very simple and systematic way. Our solution lets users find and request services through a conversational and personalized interface. Prototyped chat-bot could effectively answer product lifecycle management related queries with an added advantage that it also provides filtered reports like defect summary by test failures, defect creation etc. So, the engineer gets the relevant information quickly without having to go through a series of web pages.

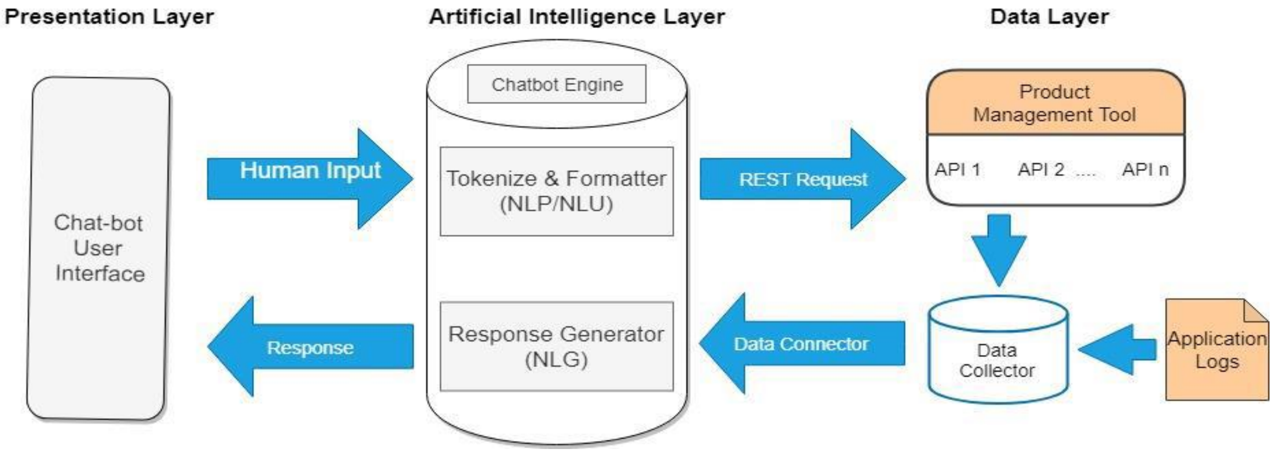


Figure 1: Proposed architecture – Chat-bot for project management

The Steps included are tokenizing the data from the user, generating meaningful words to match with the content derived from product management tools and tagging those words to provide the correct response. Human input in natural language is fed to natural language interpreter i.e. artificial intelligence layer. With the help of NLU (Natural Language Understanding) the exact user request is understood and converted to an appropriate REST API which returns required data. This data is collected in data collector. Data collector is a simple file storage on which chat-bot performs analytics to extract data which will be returned to presentation layer in human readable format as described in Figure 1.

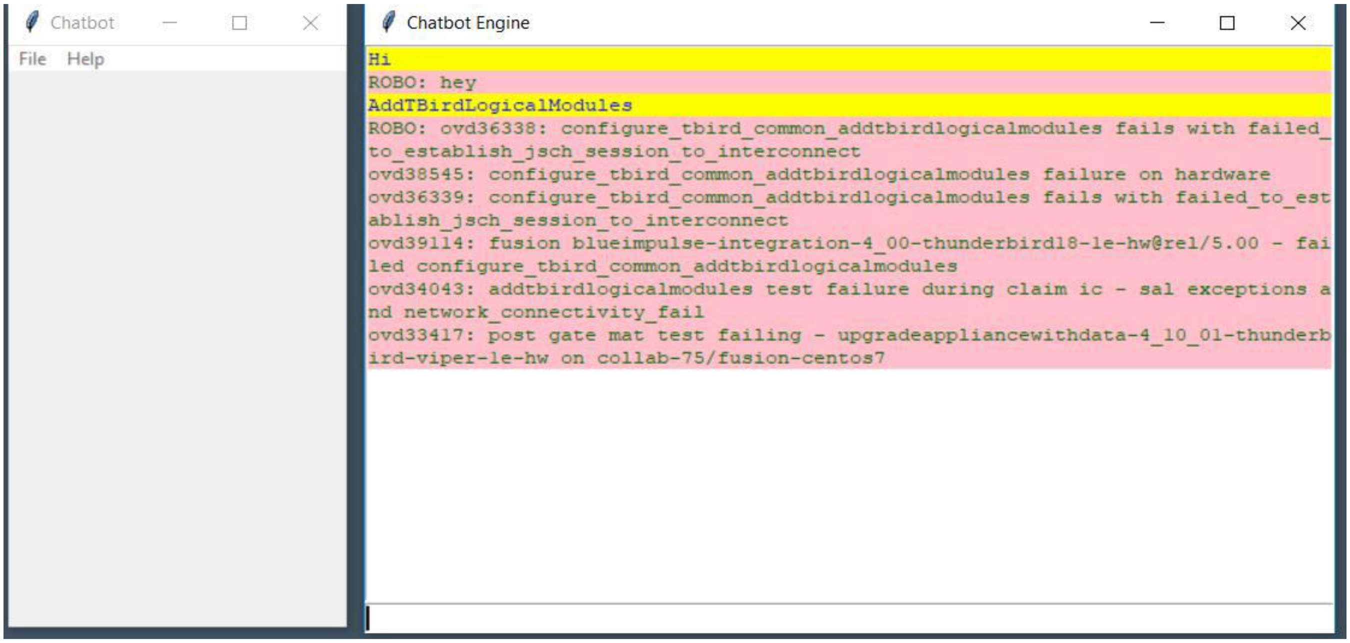
If a developer face any failures like AddTbirdLogicaModules mentioned in Figure 2, he can just ask about existing change or enhancement requests if any and chat-bot responds with defects along with summary, see Figure 2. Also, if we feed engine with logs of all older failures in a specific format, it's able to quickly respond with specific success/failure logs from log data for specific failures, see Figure 3. The NLTK library performs NLP (Natural Language Processing), NLU and NLG (Natural Language Generation) operations quickly. Solution can be used by L1, L2 engineers to find if there are any existing change requests for the failures happened at customer place.

There are situations where chat-bot cannot give right answer or cannot have answer to asked question. For such circumstances, we have planned to develop feedback mechanism for our chat-bot. chat-bot window will have Report Query button along with Submit button. If in case user is not satisfied with the answer provided by system, he can press this Report Query button. The reported query will be logged into log file for further analysis. Developer plays important role for handling and fixing such instances. He will analyze and verify these queries, update chat-bot engine. So that whenever engineer enters the same query next time he will get correct answer. In this way chat-bot will learn and improve its accuracy.

Evidence the solution works

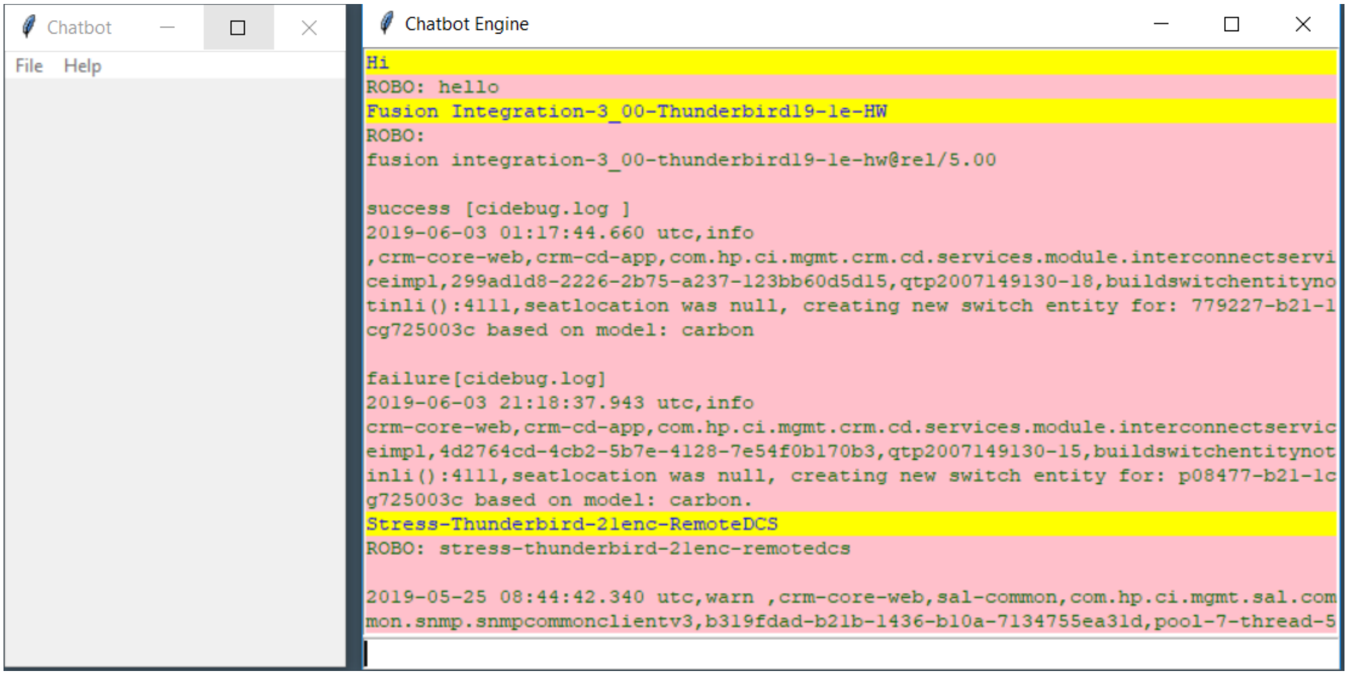
The POC is done in python using NLTK library and Rally APIs. The generic nature of chat-bot makes it easily integrate with other product management tools using REST API interfaces. We have tried different queries to validate the implementation of chat-bot. In this experiment, we have entered queries related to specific failures happened in our day to day development. The analysis of the result is shown below in Figure 2, 3.

The chat-bot response really helps engineers to pinpoint existing analysis on failures, root cause it faster and reduce duplicate efforts. Considering the average of 10-20 failures per day, tool reduces the triaging time by 30%, reduces duplicates work and avoids multiple resources triaging similar failures. The total benefit of this solution can be extrapolated when huge volumes of similar situations are considered.



```
Hi
ROBO: hey
AddTBirdLogicalModules
ROBO: ovd36338: configure_tbird_common_addtbirdlogicalmodules fails with failed_
to_establish_jsch_session_to_interconnect
ovd38545: configure_tbird_common_addtbirdlogicalmodules failure on hardware
ovd36339: configure_tbird_common_addtbirdlogicalmodules fails with failed_to_est
ablish_jsch_session_to_interconnect
ovd39114: fusion blueimpulse-integration-4_00-thunderbird18-le-hw@rel/5.00 - fai
led configure_tbird_common_addtbirdlogicalmodules
ovd34043: addtbirdlogicalmodules test failure during claim ic - sal exceptions a
nd network_connectivity_fail
ovd33417: post gate mat test failing - upgradeappliancewithdata-4_10_01-thunderb
ird-viper-le-hw on collab-75/fusion-centos7
```

Figure 2: Example of correlating existing features or enhancement requests with failure signature



```
Hi
ROBO: hello
Fusion Integration-3_00-Thunderbird19-le-HW
ROBO:
fusion integration-3_00-thunderbird19-le-hw@rel/5.00

success [cidebug.log ]
2019-06-03 01:17:44.660 utc,info
,crm-core-web,crm-cd-app,com.hp.ci.mgmt.crm.cd.services.module.interconnectservi
ceimpl,299ad1d8-2226-2b75-a237-123bb60d5d15,qtp2007149130-18,buildswitchentityno
tinli():4111,seatlocation was null, creating new switch entity for: 779227-b21-l
cg725003c based on model: carbon

failure[cidebug.log]
2019-06-03 21:18:37.943 utc,info
crm-core-web,crm-cd-app,com.hp.ci.mgmt.crm.cd.services.module.interconnectservic
eimpl,4d2764cd-4cb2-5b7e-4128-7e54f0b170b3,qtp2007149130-15,buildswitchentitynot
inli():4111,seatlocation was null, creating new switch entity for: p08477-b21-lc
g725003c based on model: carbon.
Stress-Thunderbird-2lenc-RemoteDCS
ROBO: stress-thunderbird-2lenc-remotedcs

2019-05-25 08:44:42.340 utc,warn ,crm-core-web,sal-common,com.hp.ci.mgmt.sal.com
mon.snmp.snmpcommonclientv3,b319fdad-b21b-1436-b10a-7134755ea31d,pool-7-thread-5
```

Figure 3: Example of failure analysis using chat-bot

Competitive approaches

The current business intelligence solutions available in the existing project management and support management tools are all isolated and do not help to address the problem stated in this paper. The chat-bot frameworks are deployed only to certain domains and not integrated into software development lifecycle as presented in this paper.

Current status

The POC as mentioned in the evidence section above was used to validate the claims stated in this paper. The proposed solution would be a stepping stone in having in place an intelligent query handling chat-bot which could in the next phases not just respond but self-learn to improve the quality of response.

Next steps

We plan to improve chat-bot engine in a way that it should be able to provide more intelligent and collective summary constructed by combining not just the data from product management tools but also from emails, databases and other sources of data. Also, we have plans to make the chat-bot easily configurable to variety of products using configuration file. This will enable integration with other product lines. We have started working on improving performance and time of chat-bot against large datasets. There can be improvement done for Stemming and Lemmatization to get powerful search and accurate results by performing analytics on preexisting dataset.]

References

Paper from International Journal: http://ijarcet.org/wp-content/uploads/IJARCET-VOL-4-ISSUE-10-3783-3785.pdf

NLTK Source : https://www.nltk.org/book

https://pythonprogramming.net/tokenizing-words-sentences-nltk-tutorial/

https://medium.com/analytics-vidhya/building-a-simple-chatbot-in-python-using-nltk-7c8c8215ac6e