Agile TweetViz from team Geek!O

Project Report

Harsha Kadekar
Yash Chopra
Gowtham Nayak
Anoop Jatavallabha
Seema Suresh

March 23, 2016

Contents

1 Introduction **4**

2 Filtering and sorting of tweets based on hashtags, user and tweet relationship **4**

3 Requirements: Problem Statement and Specifications **4**

4 Development Process **5**

5 Architecture and Design **6**
5.1 Tweet Fetch 7
5.2 Tweet Analysis 7
5.3 Tweet Visualization 8

6 Project Plan **8**

7 Management Plan **9**

8 Implementation **10**

9 Validation and Verification **13**

10 Outcomes and lessons learned **15**

11 Future Work, Acknowledgment and Conclusion **17**

List of Figures

1 High Level Modules . . . 7
2 High Level Design of Tweet Fetch Module . . . 8
3 High Level Design of Tweet Analysis . . . 8
4 Product backlogs in Taiga[11] . . . 11
5 Virtual Sprint Board in Taiga[11] . . . 11
6 Task information in Taiga[11] . . . 12
7 Tweet Fetch module execution . . . 12
8 Hashtag relationship . . . 13
9 Tweet - hashtag relation . . . 14
10 User and tweets . . . 14
11 Connected User and tweets . . . 14
12 Popularity of tweets . . . 15
13 Issues list in Taiga[11] . . . 15
14 Issue tracking in Taiga[11] . . . 16

1 Introduction

The popularity of social media has made Twitter^{TM} an important platform for information transfer, news reporting and public opinion. Despite the significant potential for measuring public opinion, finding the perspective for any topic by sorting through massive number of tweets is an arduous task. To counteract this problem, our vision is to give tweet visualization a new meaning by distinguishing and filtering out tweets that are of concern to the user. This can be achieved by creating a top-down tree format so that the bottom most node, gives the detailed view on a topic, be it social/technical. The idea is to look for keywords that matches best from the user given topic and jolt it down further to extract popular Twitter^{TM} users and hashtags that are relevant. Further, these popular Twitter^{TM} profiles and hashtags are combined to obtain a single visualization that gives an overall popularity and analysis of the topics of interest.

The tweets are fetched based on topics given by the user, where a topic could be anything from a word to few sentences. This process is achieved by using two separate libraries in Python: NLTK library (Natural Language Toolkit) and Tweepy library. For analyzing tweets, a Smalltalk based platform Moose/Pharo is used where a metamodel is defined which in turn will feed data for visualization. Finally, visualization is achieved using Roassal library in Pharo.

We have tried to utilize the maximum power of Roassal library by demonstrating assorted visualizations while establishing relationship between different aspects of tweet and its user.

2 Filtering and sorting of tweets based on hashtags, user and tweet relationship

Twitter^{TM} is a platform where individuals can express their views by using sentences having 140 characters or less. These tweets can further be liked or retweeted by other followers, there by increasing the viewership among the users. An important characteristic of Twitter^{TM} is the association of tweets with a certain topic by using "hashtags". A hashtag is created by placing a "#" in front of a word. By doing so, a tweet can be tagged to a certain topic. A tweet can have a single or multiple hashtags thus representing either one or multiple topics simultaneously. For example, to express a security concern in Java, a tweet can be mentioned as "A prog example of trojan using Java. Be aware of threat! #java #security " or to announce a job posting a tweet can be mentioned as "Experienced software engineers needed for front end development. DM me details. #java #UIdesign #jobs". By mentioning more than one hashtag in a tweet, a user is trying to establish the relationship between two different topics. Therefore, in our second example, we can find same tweet by searching #java and #jobs. Hashtag relationships are important because using this on a single tweet level, we can establish a tweets relationship to various topics. But on scalable level we can establish the pattern between user and tweet to vast majority of social/technical topics. The added advantage is, our program will mainly focus on tweets related to software security and will not consider topics such as national security, community security and other forms of security.

3 Requirements: Problem Statement and Specifications

We have gathered our initial requirements from SCORE website[9] and have further refined them by conducting brainstorming sessions with the team and online meetings with our project sponsor.

We have developed different user stories which would help us to achieve our vision of the project.

- Our first part of the vision was to fetch meaningful tweets based on the user input. To achieve this we had "USER-STORY52:As a user of Tweet Viz, I should be able to give the topic or sentence to the Product, using that topic/sentence product has to fetch tweets"

- Second part of the vision was to analyze fetched tweets in a meaningful way. To achieve this we had "USER-STORY53:As a developer I need meta models to store the tweets, user details, hashtags, location etc. in objects. These objects will be used to carry out visualization." or "USER-STORY23:As a user, I would like to relate the fetched tweets with their user handles"

- Third part of the vision was to visualize the way which can clearly show the grouping of tweets and users, hashtag popularity, hashtag relationships. To achieve aspects we had different users stories like - "USER-STORY95:As a user I would like a graph representation of the retweets and favorites for each hashtags", "USER-STORY24: As a user, I would like to visualize tweets with the user who have tweeted them", "USER-STORY79: As a user I need to get a visualization pattern based on the given search category"

Apart from this we had certain process requirements like

- "USER-STORY25: Pharo/MOOSE has to be used for analysis of tweets"
- "USER-STORY26: Roassal should be used for visualization of analysis done on tweets."

We have achieved our vision by first establishing the flow of hashtags and further relating those hashtags to the Twitter^{TM} users and their tweets. Secondly, we have utilized this relationship to categorize the tweets and tweet handlers into different groups. Finally, we have visualized the analyzed tweets and their relationship so that this relationship can be observed and studied in a much compact and simpler manner.

This analysis will help us to determine the popularity of an user in a particular field of interest. For example, user is tweeting on the topics related to #Java, #Security, #Job, #Issue, and #JDK. Among these topics, #Security is the most popular hashtag. This condition can be determined based on the counts of retweets and number of likes on a tweet. Our aim is to clearly understand user area of expertise. Even though Twitter^{TM} user is tweeting about job and jdk, it is clear that user specializes in tweets related to Java Security.

4 Development Process

In this project we used Agile methods[12], specifically the Scrum process with 2 week sprints[13]. The main reason to choose was for its ability to produce deliverable from beginning, without the availability of full set of clear requirements. For example, we were sure on how to proceed for the user stories related to Tweet fetch, but were unclear on the methods that will be used to create meta-model and visualization. Considering this, agile lets us start working on the known components from the day one. Another reason to choose Scrum was for its ability to handle changes effectively. It provided us the capacity to continuously update the backlog, while refining requirements in the background and at the same time working on known components of the project. In the beginning of project our understanding of requirements was only limited to what was mentioned on SCORE website. We knew these requirements will be updated after our meeting with project sponsor Dr. Alexander Bergel. Scrum ensured the risk involved with the updating requirements and using new and unknown tool have minimum negative impact on our project.

From risk management perspective, we were able to identify two major hurdles that we may face during the project duration:

1. Lack of knowledge of Pharo/MOOSE may slow down this time sensitive project.
2. Whether chosen Database Management System would handle or scale to the fetched Twitter^{TM} payload.

Lack of understanding of Pharo/MOOSE was our biggest challenge. We dedicated one full sprint out of five, trying to get better understanding of the new language and environment. We were skeptical about the "What if not" part of the learning process, thus discussing number of other options. First option discussed was; can the functionality depending on the PHARO/MOOSE be developed using some other technology? We identified that these functionalities can be implemented to some extent using the libraries provided by the python and also javascript, but the unique feature of Pharo/MOOSE is "object profiling". The features of MOOSE/PHARO help in the easy creation and integration of meta model and visualization as both are closely related. Given that, use of Pharo/ Moose was also part of process requirements, moving out of Pharo/MOOSE was ruled out. To improve the technical knowledge each team member was given certain topics in the Pharo/MOOSE in parallel to the product development. Thus filling the knowledge void.

Secondly, we were not sure what should be the preferred data storage method as we were planning to store 1 million tweets. Our initial choice of Database was Microsoft SQL Server Express edition. After doing the feasibility study of space and time impact of million tweets database interaction, it was found that Express edition was not able to handle 1 million records due to storage restriction. We decided to go ahead with MySQL. But the question that followed was whether or not we need to use Database at all in the project. As we were fairly new to MOOSE, we were not sure of configuring and accessing database from MOOSE. After further reading and research, we got a better way of adding packages that could directly read from CSV file. This helped us to easily create the meta-model for the project. After exploring and discussing all the options mentioned above, we were clear of right solution, thus making rest of the project development a smooth sail.

As both the platforms were new to us, the main task in our first sprint was to understand them. We went ahead and generated our first set of product related user stories. Some of the user stories were found vague. We discussed this problem with our project sponsor and Dr. Kevin Gary (faculty leader of Software Enterprise course, Arizona State University) along with this we conducted brainstorming sessions to create better user stories. We decided to place more emphasis on Meta-model creation and visualization, rather than user interface.

Agile helped us to continuously add or change the requirements without hindering the product development. At the end of first sprint, requirements related to fetching of tweets were finalized while discussion over meta-model and visualization continued. Requirements for meta-model and visualization were finalized at the end of second and third sprints respectively. We had a discussion with the sponsor at regular intervals and further incorporated changes to the existing requirements. Any unclear requirement acted as a placeholder for a conversation with the product sponsor to better understand the requirement. Software Requirements Specifications(SRS) were updated throughout the project.

Taiga[11] and Google site[10] were used for implementation of software process whereas Git was used for change management.

5 Architecture and Design

From the technical aspect of the project description[9], it was clear that three main parts of the project are

- Fetching the meaningful tweets
- Creating a meta-model
- Visualizing the tweets present in the meta-model

Based on the present information, it was deduced that we need a separate module for the meta-model creation. Our code should have the ability to analyze the tweets and generate the meta-model thus establishing the relationships between tweets. Once we have the meta-model, based on the third technical requirement we have to visualize different components of tweets. Different components of tweets from same meta-model can be used in several forms to create various visualizations depicting different meaning. For addressing all the three points, it is very important to have meaningful set of tweets. Fetching tweets doesn't come under meta model creation or visualization modules. We need a separate module which has the ability to fetch meaningful set of tweets from Twitter^{TM} and can pass it to meta-model module, more like a business logic layer. That's why we have come up with the following three different modules for our product.

1. Tweet Fetch - Get meaningful tweets
2. Tweet Analysis - Create meta-model
3. Tweet Visualization - Visualize the relationship of tweets.

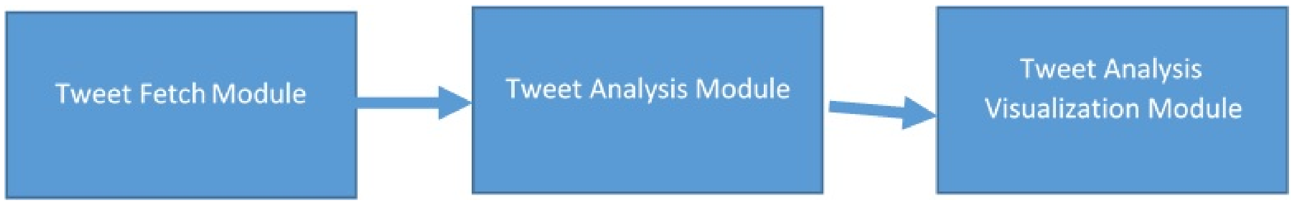


Figure 1: High Level Modules

5.1 Tweet Fetch

Tweet Fetch Module is responsible for fetching tweets based on user input. This module can further be divided into three main parts:

- Processing user input
- Fetching tweets
- Storing tweets

A user input can be anything from a single word to a complete sentence, which is then parsed to generate hashtags. These hashtags are used to search and download the relevant tweets. Once the tweets are downloaded, they are stored in SQL table and CSV file. Our next module, Tweet Analysis, directly takes CSV file as input. DataStorageReader as outlined in figure 2, is responsible for reading configuration file, interacting with TweetViz MySQL DB and interacting with CSV file through filereaderwriter component. Configuration file contains following parameters - SQL connection related parameters; Type of tweets fetched - most recent, popular, mixed; Type of storage - CSV or SQL; CSV file path; Debug level.

For security reasons, SQL parameters are encrypted and stored in configuration file. HashtagGenerator is responsible for generating the hashtags from the given user input using natural language processing. Tweet Fetch module will get the tweets from Twitter$^{\mathrm{TM}}$ based on the given hashtags. TweetBackend is responsible for the integration of all the different components of Tweet Fetch module.

During the design of modules we have incorporated various design patterns.

- Interaction between HashTagGenerator and DataStorageReader is an example of Observer pattern. Tweet Fetch module first calls the HashTagGenerator to generate hashtags and fetch the tweets related to those generated hashtag. Once this event is complete, it calls the DataStorageReader to store those generated tweets. In future, we also intend to use Observer pattern for interacting with the web interface.
- We made DataStorageReader class a singleton. We wanted a single point of communication for the database. All the different sub modules of Tweet Fetch should interact with database via a single object of DataStorageReader. By this we avoided multiple connections to the database.
- DataStorageReader class is a single interface for any communication regarding storage and reading of the data. This storage and reading can be done in two ways Database or CSV. DataStorageReader provides write/read functionality to Database where as for CSV the same task is delegated to CSVFileReader. For this we tried to incorporate Adapter pattern.

5.2 Tweet Analysis

Two main responsibilities of this module are: Read the tweets and Generate the metamodel. CSVFileToTweetVizModel as outlined in figure 3, will use Neo-CSV package of MOOSE to read the CSV file and further generate the metamodel. Metamodel has inter-related objects representing Tweet, Tweet-User, HashTag and SearchCategory.

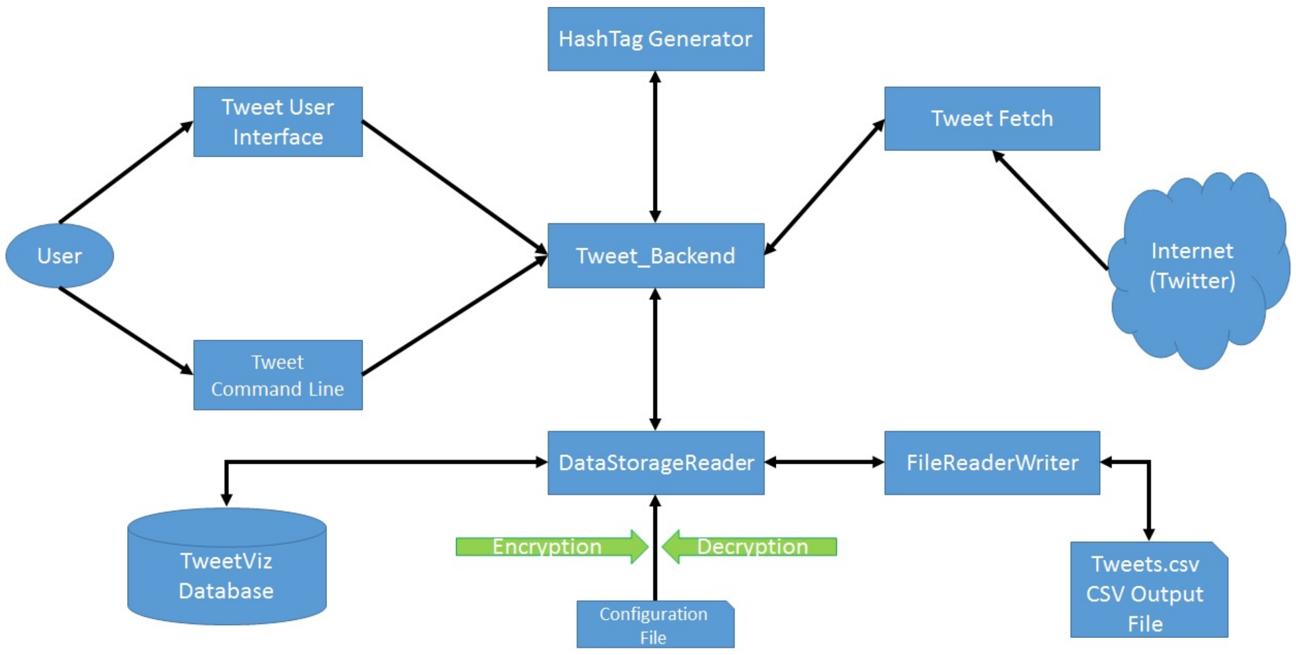


Figure 2: High Level Design of Tweet Fetch Module

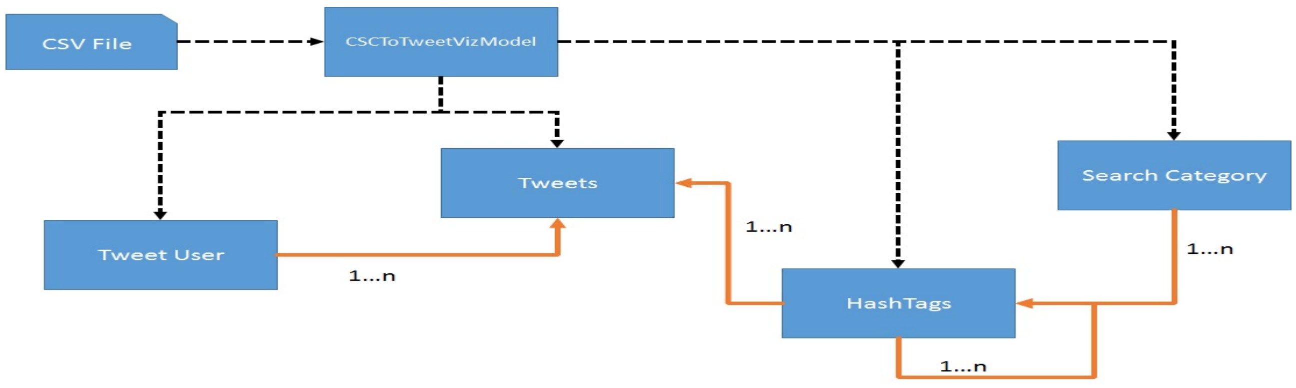


Figure 3: High Level Design of Tweet Analysis

5.3 Tweet Visualization

Multiple visualizations are generated representing the relationships between different objects of the metamodel. To achieve this, we have used RTView and Roassal package in Pharo/MOOSE.

6 Project Plan

Project development was divided into five iterations. First sprint was utilized for understanding of requirements and analysis of tools; next three sprints were dedicated for development of the major technical components of product; and last sprint was allocated for integration, testing and documentation. Sprint meeting was conducted every alternate Saturday (end of every sprint), where sprint agendas were discussed and demonstrated. Sprint details of the project are -

1. Sprint 1 (21 SEP 2015-04 OCT 2015): Understanding tools and requirements
    - Demonstrate different tools mentioned in requirements as learned by team members.
    - Participate in brainstorming session to further elaborate the requirements of the project.
2. Sprint 2 (05 OCT 2015-18 OCT 2015): Obtain user input in the form of a word or a sentence.

- Parse the user sentence to build hashtags.
- Fetch the tweets based on hashtags.
- Store the fetched tweets

After this sprint, the demo was presented in front of an external reviewer, Dr. Kevin Gary and his feedback was used to make technical and procedural changes to our project.

3. Sprint 3 (19 OCT 2015-01 NOV 2015): Analysis of tweets

   - Analyze the data to produce Tweet objects in MOOSE
   - Analyze the data to produce Hashtag objects in Pharo
   - Analyze the data to produce User (Twitter^{TM} handles) objects in Pharo.
   - Analyze the data to produce Search Category (Application user input) objects in Pharo

   The working model will demonstrate meta-model containing several objects. These objects will be further used in Roassal library for the visualization, thus offering user the opportunity to review the topics of visualization.

4. Sprint 4 (02 NOV 2015-15 NOV 2015): Visualization of meta-model

   - Visualize Hashtag tree relationship.
   - Visualize Hashtag and Tweet words cloud relationship.
   - Visualize User, Tweet and hashtag organization chart relationship.
   - Visualize User and Tweet connection circle relationship.
   - Visualize tweet messages of different tweets.

   The demo will present different visualization based on generated meta-model. There are several different types of Smart-arts and charts presented to demonstrate the wide range of visualization power of Roassal library in Pharo.

5. Sprint 5 (16 NOV 2015-23 NOV 2015): Regression

   - Integrate all the modules of the product.
   - Complete documentation related to product and project.
   - Full system run from user input to final visualization.

   The focus of this sprint was on the integration of separate modules and testing the code in parts and in full. The system was required to perform flawlessly.

7 Management Plan

We followed Agile work environment with two weeks of sprint duration and conducted stand-up meetings on every alternate day. For every sprint meeting, the first half was dedicated to sprint review and second half of the meeting was reserved for sprint planning. Sprint review focused on the following points:

- Which tasks were left incomplete and the reasons behind them?
- What worked well for us?
- What didn't worked well for us?
- What actions can we take to improve our process?

In the sprint planning, following things were discussed:

- Objective of the new sprint.
- Any new user stories or any change to the priority of the user stories in product backlog?
- Which user stories needed to be moved from product backlog to sprint backlog?
- Allocation of tasks and roles of different team members (tester/developer/product owner)

Three questions answered by each team member in the daily stand-up meeting were

1. What did I do?
2. What I will be doing till next meeting?
3. Is there any issue which needs immediate attention by the team?

Two tools helped us monitor the whole process - Taiga[11] and Google Site[10]. Taiga was used as our visual management board that helped us to keep record of the product backlogs (figure 4) and sprint backlogs (figure 3).

Once the user story has been moved from the product backlog to the sprint backlog, the state of the tasks of user stories can be monitored during that sprint as: New, In progress, Testing or Completed. Each user stories were provided with unique ID for traceability and a point (5, 10, 15) based system for prioritization. User stories or tasks were also provided with a unit test/ developer integration test (figure 6).

For any task that did not involve coding, a knowledge base document was written to explain the final outcome of the task. For example, our first sprint mostly consisted of the tasks that involved learning new platforms like MOOSE, PHARO, etc. In this case, a document was prepared explaining the concepts learned and shared with the team. Issues were logged in the Taigas issue tracker tool.

Google site[10] was a bird eye view of what was going on in the team. Every sprint and stand-up meeting details were recorded, including date of the meeting and important points discussed. Documents related to the process and product were placed in the wiki section of the site.

There were two scheduled reviews for Quality Assurance. These reviews were performed by Dr. Kevin Gary after sprint two and sprint five. The feedback after first review: 1) User stories are not formed correctly - This should be the place holder for the conversation with the customer 2) Many of the tasks were being closed nearing the end of the sprint - Both of these points were taken into account and further improved on.

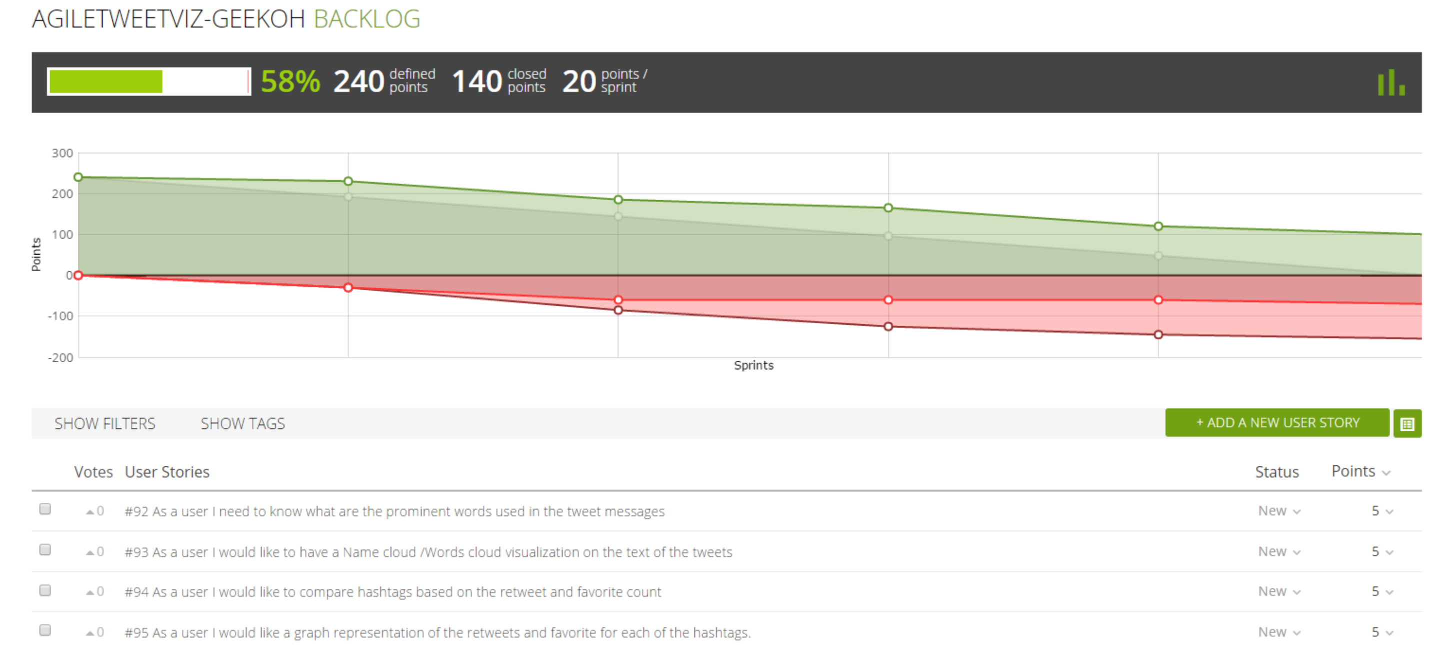
8 Implementation

As mentioned in the architecture, there are three main modules in this product.

- Tweet Fetch
- Tweet Analysis
- Tweet Visualization

Tweet Fetch module (Figure 7) was completely developed using python. Various packages used for implementation are:

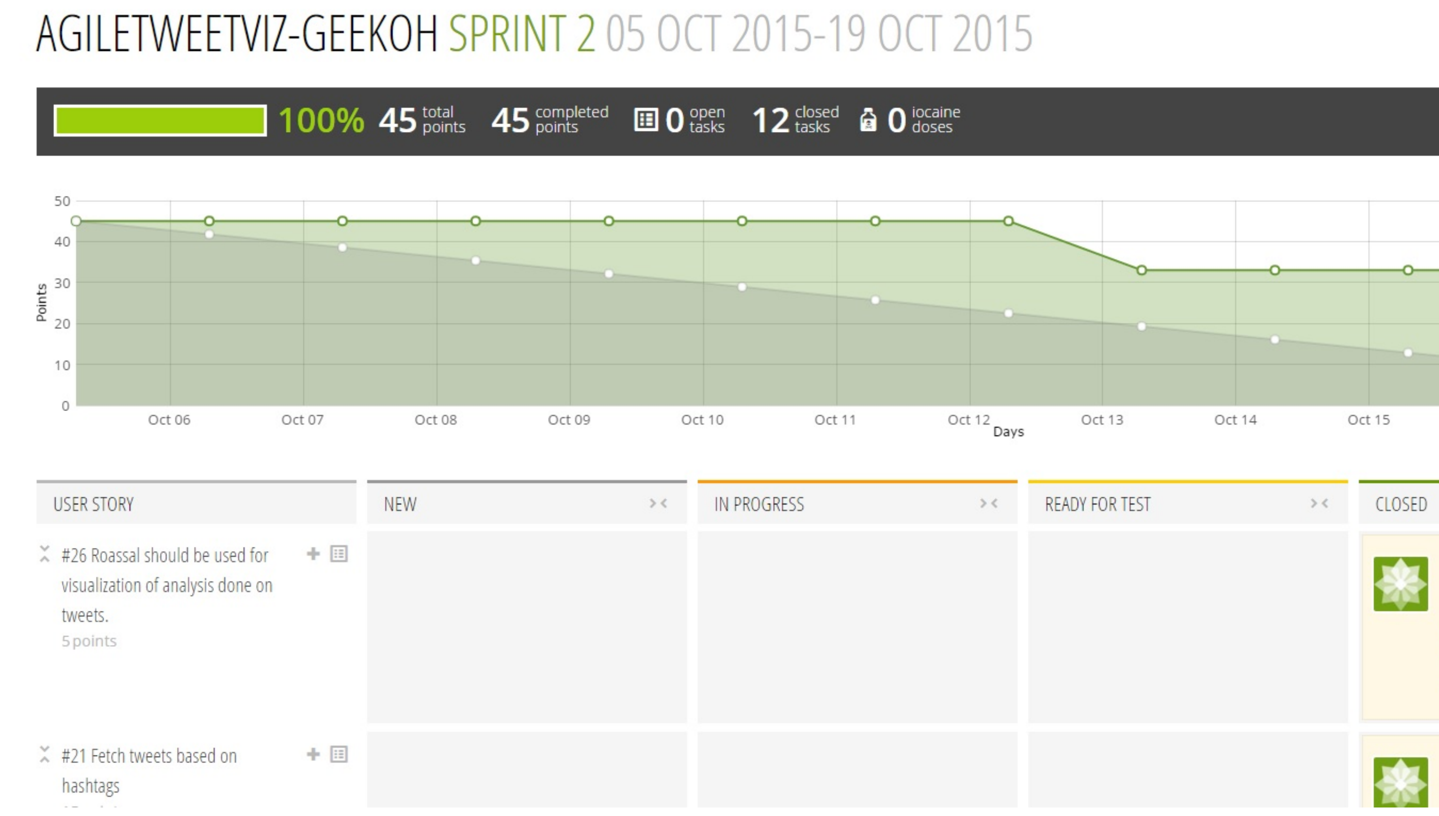
- Tweepy - Fetch tweets from Twitter^{TM}.
- Mysql Connector - Interact with MySQL database.
- NLTK - For natural language processing of user input.
- Py2Exe - To convert the python scripts to a deliverable.


AGILETWEETVIZ-GEEKOH BACKLOG

58% 240 defined points 140 closed points 20 points / sprint

	Votes	User Stories	Status	Points
	0	#92 As a user I need to know what are the prominent words used in the tweet messages	New	5
	0	#93 As a user I would like to have a Name cloud /Words cloud visualization on the text of the tweets	New	5
	0	#94 As a user I would like to compare hashtags based on the retweet and favorite count	New	5
	0	#95 As a user I would like a graph representation of the retweets and favorite for each of the hashtags.	New	5

Figure 4: Product backlogs in Taiga[11]


AGILETWEETVIZ-GEEKOH SPRINT 2 05 OCT 2015-19 OCT 2015

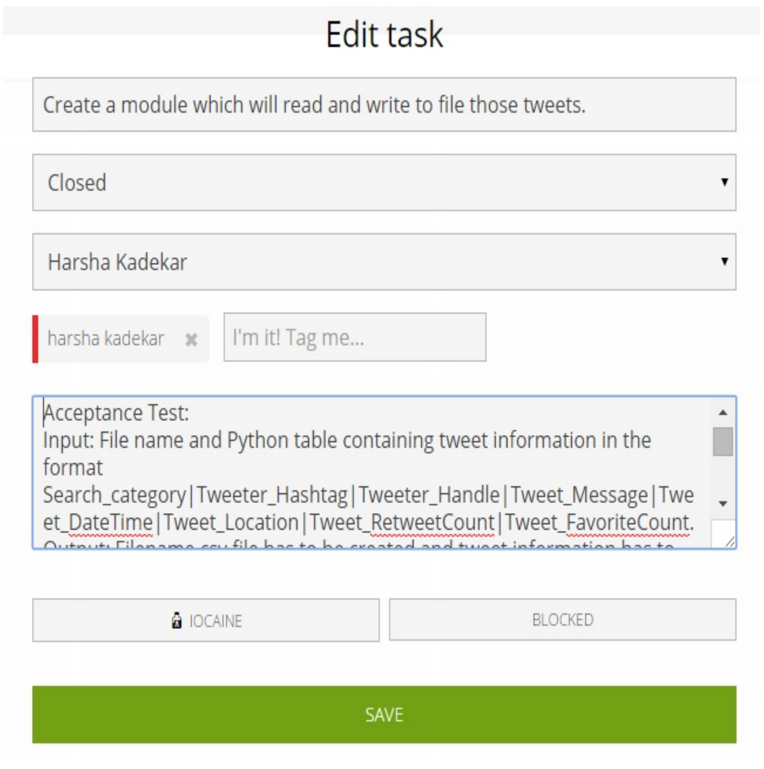
100% 45 total points 45 completed points 0 open tasks 12 closed tasks 0 iocaine doses

#26 Roassal should be used for visualization of analysis done on tweets.
5 points

#21 Fetch tweets based on hashtags

Figure 5: Virtual Sprint Board in Taiga[11]

Best coding practices including naming conventions for variables, functions, files and folders, comments and logging were discussed and followed throughout the project. All the commits to the Git



Edit task

Create a module which will read and write to file those tweets.

Closed

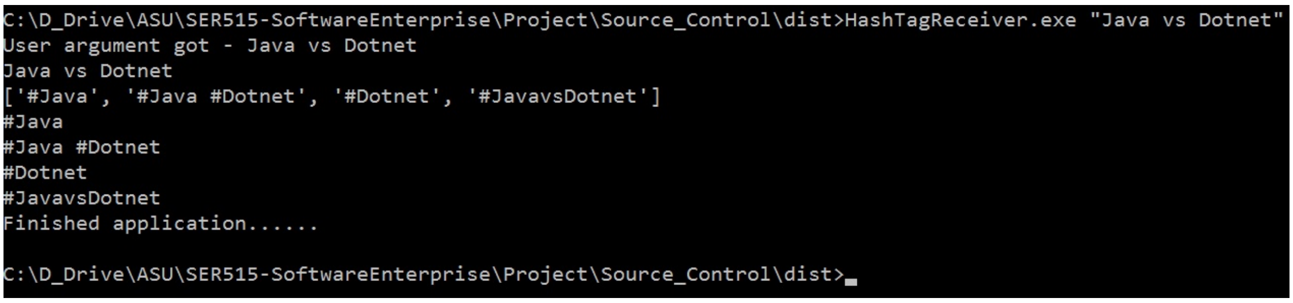
Harsha Kadekar

harsha kadekar

I'm it! Tag me...

Acceptance Test:
Input: File name and Python table containing tweet information in the format
Search_category|Tweeter_Hashtag|Tweeter_Handle|Tweet_Message|Tweet_DateTime|Tweet_Location|Tweet_RetweetCount|Tweet_FavoriteCount.

Figure 6: Task information in Taiga[11]



```
C:\D_Drive\ASU\SER515-SoftwareEnterprise\Project\Source_Control\dist>HashTagReceiver.exe "Java vs Dotnet"
User argument got - Java vs Dotnet
Java vs Dotnet
['#Java', '#Java #Dotnet', '#Dotnet', '#JavavsDotnet']
#Java
#Java #Dotnet
#Dotnet
#JavavsDotnet
Finished application......

C:\D_Drive\ASU\SER515-SoftwareEnterprise\Project\Source_Control\dist>_
```

Figure 7: Tweet Fetch module execution

were accompanied by the taskid and a brief explanation of the changes made to the code.

Tweet Analysis and Tweet Fetch modules were developed in SmallTalk based Pharo platform. Pharo and MOOSE has inbuilt source control. All the packages created in the "package-cache" folder of MOOSE were also checked into Git[14] with proper comments.

Visualizations developed during the course of our implementation are

1. Figure 8: Given a set of tweets based on a search category, we are able to get the hashtag's relationship. This is an important part of the vision. Here we are developing a tree like relationship of hashtags and finally identifying important hashtags relating to the input provided by user. Initial aqua color ellipse in the figure 8, is user input whereas purple ellipses are the hashtags generated from the user input and red color boxes represent the next level of hashtags.

2. Figure 9: Once the hashtag's relationship has been generated, the tweets were added in the visualization. In this visualization we are trying to group the tweets into different hashtags. As we identified the prominent hashtags, by associating the tweets to them we are trying to identify the most relevant tweets. In figure 9, green color ellipses in the outer circumference represents the tweets. Blue color ellipse is the main hashtag whereas red color boxes are sub hashtags. Size of the red color boxes depends on the number of tweets associated with that hashtag.

3. In figure 10, we have generated visualization based on users and their tweets. Here, the idea is

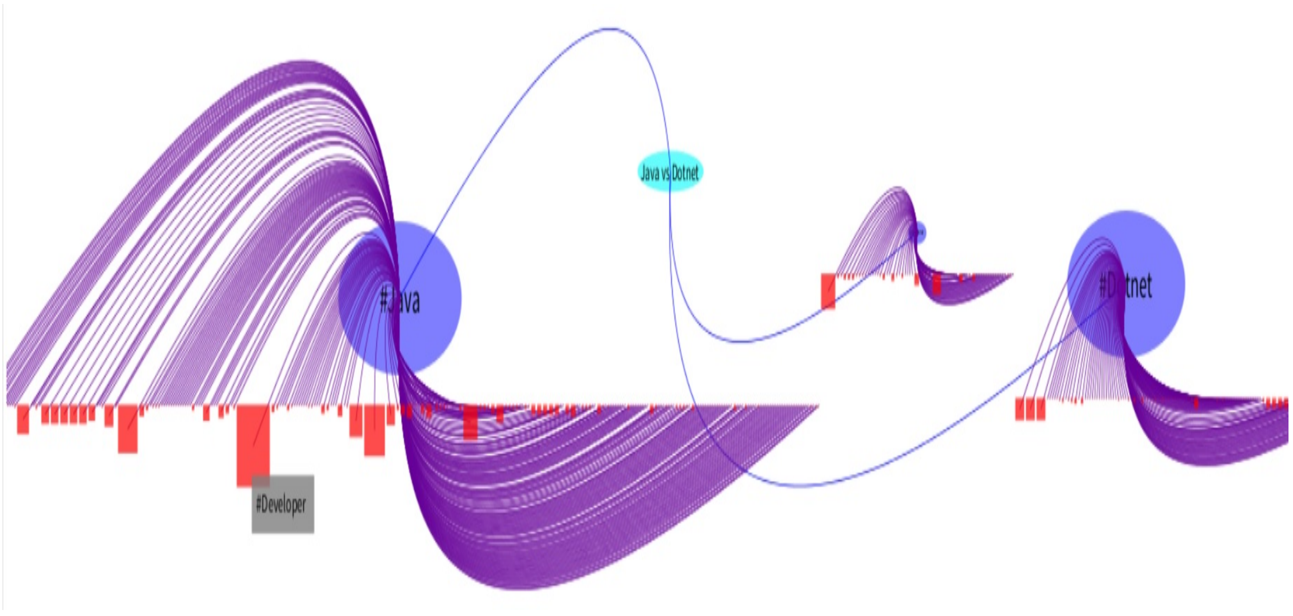


Figure 8: Hashtag relationship

to identify the most popular Twitter^{TM} users who have more tweets related to the given user topic. Here purple ellipses are users and red color boxes are tweets. Size of ellipse is dependent on the number of tweets by that user. Initially, users and tweets are displayed separately. Once the "connect" Button clicked (figure 11), all the tweets tweeted by a user will come together and circle around that user.

4. Figure 12 gives simple understanding of which hashtags are popular based on the retweets and favorites of tweets associated under those hashtags. Simple graphs are used to represent those calculated popularity matrix.

9 Validation and Verification

So for in this project we have emphasized on verification rather than validation. Moving ahead we would like to give equal importance to both, validation and verification. As part of verification we had unit test cases, developer and system integration test cases and finally regression test cases. As part of validation we had acceptance test cases. Validation was performed by manual inspection where visual patterns were matched to the speculated results of small number of tweets.

Initially for validation purpose, we presented our project in class, explaining how and why any particular visualization was developed. We later opened the stage for questions from the audience and let them evaluate our work so far. We also presented our project in "Computing Informatics and Decision System Engineering (CIDSE)" showcase at Arizona State University, where all the students and faculty members from various background had the opportunity to analyze our project. Changes like different color coding and size of the graph were made after the input from both the presentations. The effectiveness of these changes were measured in our second validation process that was conducted using an online survey, where visualization was provided without any explanation. For this survey, participants were chosen randomly, making sure no one was the stakeholder of the project and making sure that participants belonged to different background. We have evaluated the responses from 20 participants as our initial validation process. Survey was conducted for Figure 8, Figure 9 and Figure 11; asking participants 1)The level of understanding of the visualization, 2) Can they identify the most prominent part of the visualization 3) How can we change the visualization to be more intuitive. The results relating to Figure 8 indicated that nearly 50% of the users found it difficult to interpret the visualization. One of the reason is the disassociation of the visualization to the regular Twitter^{TM} use. Visualization of Figure 8 presented the flow of hashtags which did not directly relate to how Twitter^{TM}

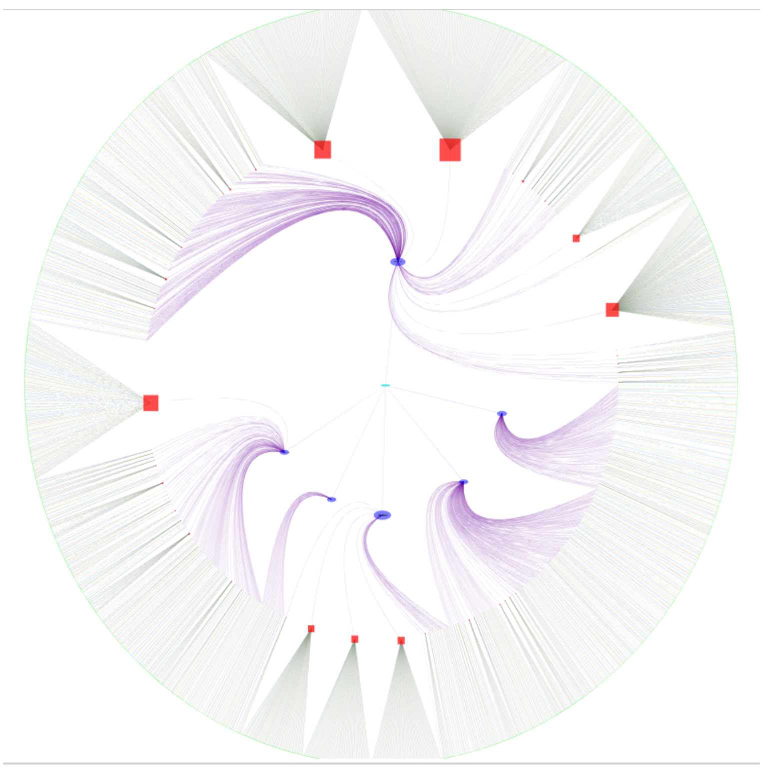


Figure 9: Tweet - hashtag relation

is used by an individual. This visualization was important for the project, as it was building block for our vision and led to more advanced visualizations as we progressed in the project.

The second visualization(Figure 9), which depicts a direct relationship between the hashtag and tweets showed a more positive understanding among the users. This was because of different layers in visualization, more color coding and the lines connecting between layers were not intersecting. Thus making it easy for the users to understand the figure at first glance.

The final visualization(Figure 10 and Figure 11) displayed the most positive result among the three visualizations. More than half of the users reported it to be easy(30%) or very easy(30%) to grasp. Our hypothesis says that the ease of understanding of Figure 11 is due to to the reason that this was the most relatable visualization from a user point of view. It was the relationship between Twitter^{TM} user and the tweets, which directly coincides with the basic understanding of Twitter^{TM}.

Taiga[11] was the tool used to report and track any issues during the time of product development.

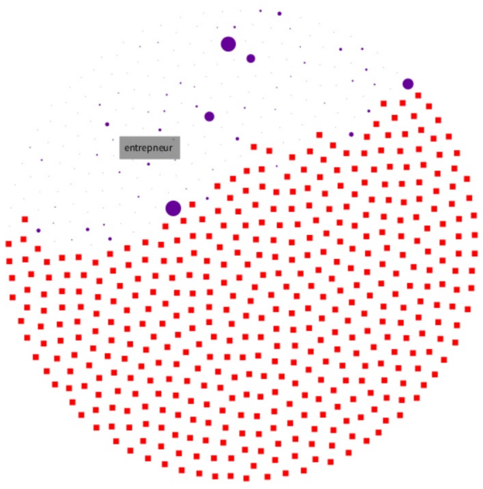


Figure 10: User and tweets

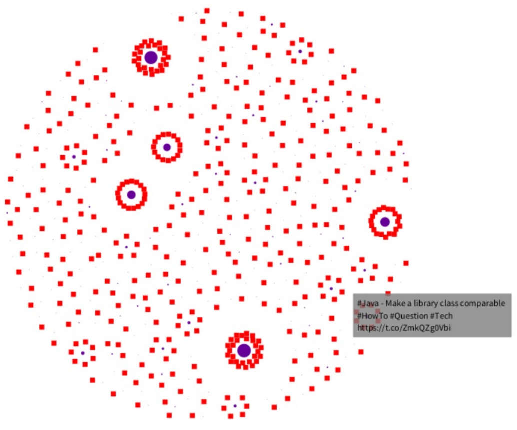


Figure 11: Connected User and tweets

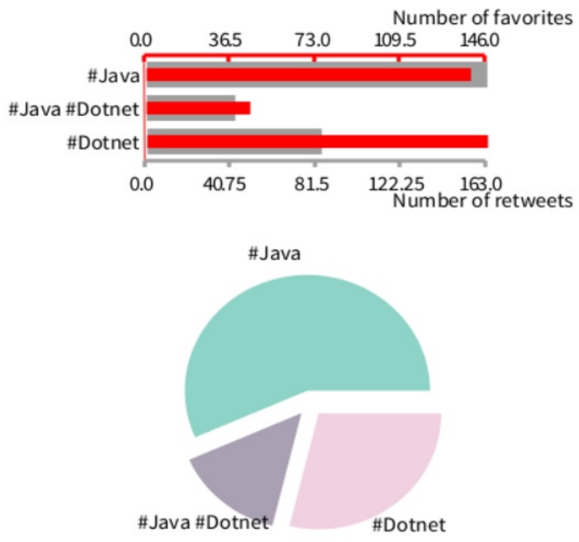
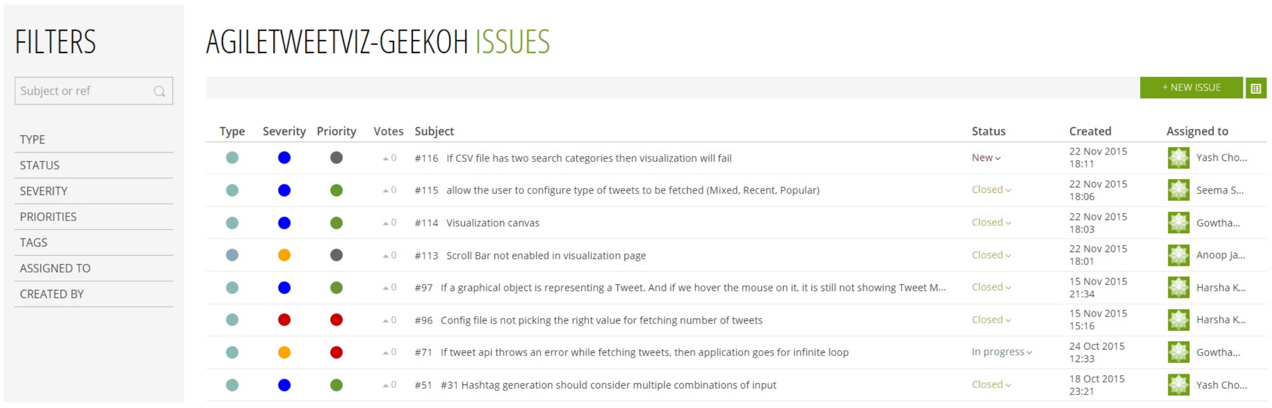


Figure 12: Popularity of tweets


FILTERS

Subject or ref

TYPE
STATUS
SEVERITY
PRIORITIES
TAGS
ASSIGNED TO
CREATED BY

AGILETWEETVIZ-GEEKOH ISSUES

+ NEW ISSUE

Type	Severity	Priority	Votes	Subject	Status	Created	Assigned to
●	●	●	0	#116 If CSV file has two search categories then visualization will fail	New	22 Nov 2015 18:11	Yash Cho...
●	●	●	0	#115 allow the user to configure type of tweets to be fetched (Mixed, Recent, Popular)	Closed	22 Nov 2015 18:06	Seema S...
●	●	●	0	#114 Visualization canvas	Closed	22 Nov 2015 18:03	Gowtha...
●	●	●	0	#113 Scroll Bar not enabled in visualization page	Closed	22 Nov 2015 18:01	Anoop Ja...
●	●	●	0	#97 If a graphical object is representing a Tweet. And if we hover the mouse on it, it is still not showing Tweet M...	Closed	15 Nov 2015 21:34	Harsha K...
●	●	●	0	#96 Config file is not picking the right value for fetching number of tweets	Closed	15 Nov 2015 15:16	Harsha K...
●	●	●	0	#71 If tweet api throws an error while fetching tweets, then application goes for infinite loop	In progress	24 Oct 2015 12:33	Gowtha...
●	●	●	0	#51 #31 Hashtag generation should consider multiple combinations of input	Closed	18 Oct 2015 23:21	Yash Cho...

Figure 13: Issues list in Taiga[11]

The task number has to be mentioned for every issue that was being logged, unless it was discovered during regression or system integration testing. Taiga gives the classification of issues as bug, question or enhancement. Apart from that, priority is provided as low, medium or high. It also classifies the bugs as severity: minor, normal, important, critical or wish-list. Any issue can be in one of the stages: New, In-progress, Ready for test, Closed, Needs Info, Rejected, Postponed.

Three Amigos rule was followed while moving the task from In-Progress to Complete state, where three Amigos included developer, tester and requirement owner. Sprint 5 was dedicated for the integration and regression testing.

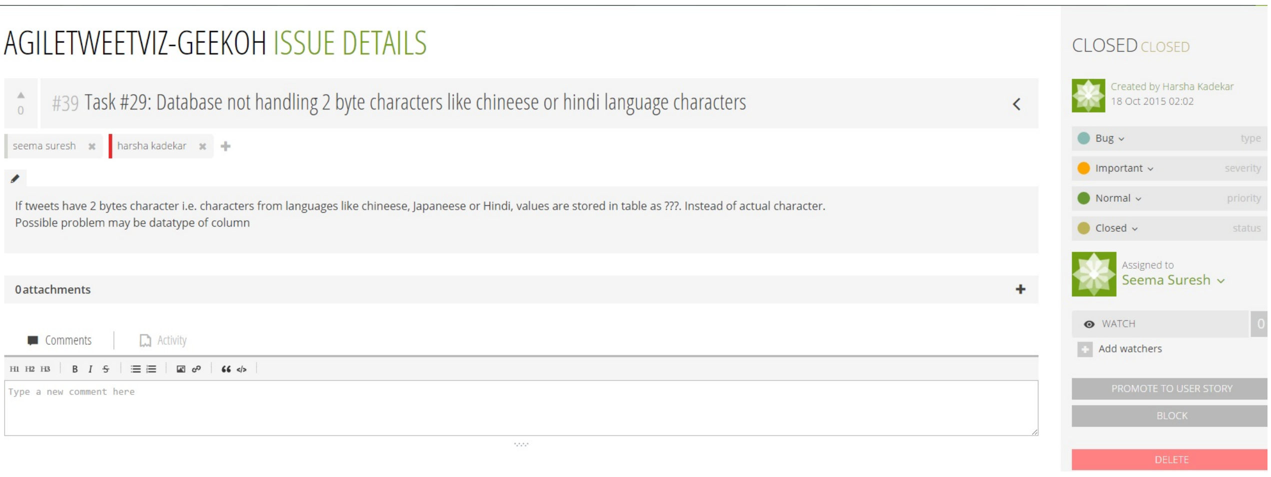
Based on the discussion with product owner, Hapao will be used as test coverage tool in the future.

10 Outcomes and lessons learned

After every sprint we conducted Retrospectives to judge and reflect on our progress so far. Some of the highlighted points of the project are mentioned below divided in the categories as they were discussed: -

What worked well for us?

- Agile process - handling change
- Pharo/MOOSE Roassal


AGILETWEETVIZ-GEEKOH ISSUE DETAILS

#39 Task #29: Database not handling 2 byte characters like chineese or hindi language characters

seema suresh | harsha kadekar

If tweets have 2 bytes character i.e. characters from languages like chineese, Japaneese or Hindi, values are stored in table as ???. Instead of actual character.
Possible problem may be datatype of column

0 attachments

Comments | Activity

Type a new comment here

CLOSED CLOSED

Created by Harsha Kadekar
18 Oct 2015 02:02

Bug — type
Important — severity
Normal — priority
Closed — status

Assigned to
Seema Suresh

Figure 14: Issue tracking in Taiga[11]

- Third party library usage - NTLK and Tweepy library

Our biggest accomplishment in this project was learning of the new tool - Pharo/Roassal and the whole learning process was made easy by Agile process. Agile did not force us to master Pharo beforehand, in fact we learned the tool as and when demanded by sprints and user stories. Agile was also helpful in accommodating our code with evolving requirements, thus providing us with the ability to start contributing to product development from the beginning. For example we started writing code to download tweets using hashtags without knowledge of what kind of visualizations it could lead to.

What did not work well for us?

- Slow learning curve of new tool is hindering the capability to create powerful/ more meaningful visualizations
- Unequal commitment of team members
- Evolving vision of Project

Since Pharo/Rossal is a very new development tool, finding an online community was difficult. It made the learning process harder, thus more time was spent on learning the tool rather than testing its limits to achieve better visualizations. We used the examples mentioned in Roassal library for better understanding and went beyond the examples to create new and vivid visualizations. This process led to several visualizations which were not necessarily required or aligned to our vision, but helped us understand the tool better[10]. Lack of clear understanding of initial requirements lead to some extra coding work that could have been avoided. For example, we focused on User Interface for the initial sprints, but later it was determined that it was not required.

What actions can we take to improve our process going forward?

- Better implementation of Agile process
- Validation and automation of test cases
- Integrate visualization to realize complete vision in a single go

There was always room for us to improve on Agile process. For example, contrary to agile method we were more comfortable in assigning one user story to one team member rather than assigning individual tasks of a user story to individual member and moving stories at constant pace. We were also unable to implement efficient daily stand-up meetings due to the conflicting schedule of team members. We followed a Test Driven Development process where we spent a lot of time in checking our code manually after making any change. Automation of tests could not be implemented due to lack of time. Our concentration was mainly on the verification part rather than validation. This needs to change by effective utilization and community engagement for proper feedback. Finally we have different visualization trying to answer different parts of the vision, in future all these things has to be combined to produce a single visualization where all information can be gathered.

11 Future Work, Acknowledgment and Conclusion

We believe this is just the beginning of the project, there is still a lot of scope for improvement. Some of the important tasks which we are working on are :

1. Map the users through the hashtags used. Categorize the users into different groups based on the hashtag's popularity
2. Develop a single web interface for the whole product. User will provide input in a webpage. The input will be passed to our product in backend to process and generate the visualization. Once the visualization is done, it will be exported to HTML5 and shown to the user in the same webpage where input was given.
3. Currently product deliverables are present in windows format. We are in the process of porting it to Linux/Unix and Mac environments.
4. Once we are able to categorize the tweets and the users based on our hashtag relationship, we would like to compare with the group categorization done via other methods as mentioned in the paper by *Abhishek Sharma, Yuan Tian and David Lo*[1] .

For our project so far we have extracted more than 80,000 tweets in real-time using python and then analyzed them using our metamodel in Pharo. We have further created 6 visualization patterns using Roassal library, which reflects various relationships between tweets, users and hashtags. There are many improvements and visualizations that can be achieved via Pharo and Roassal respectively. For example, we would like to explore the emotional aspects in a tweet.

This project was a great experience as it introduced us to new technology, while giving a practical hands-on experience in agile work environment. We would like to thank our Professor, Dr. Kevin Gary, Arizona State University, for teaching and guiding us in the right path throughout the duration of project and helping us on Agile development process. We would also like to thank our sponsor Dr. Alexandre Bergel ,University of Chile, for giving us opportunity to work in Pharo/Roassal and guiding us from time to time about these technologies as well as helping us to understand the requirements.

References

[1] Abhishek Sharma, Yuan Tian and David Lo. *What's Hot in Software Engineering Twitter Space?.* 2015 IEEE International Conference on Software Maintenance and Evolution (ICSME), 2015

[2] *Agile Visualization Book* `http://agilevisualization.com/#book`

[3] Andrew P Black, Stphane Ducasse, Oscar Nierstrasz, Damien Pollet, Damien Cassou, Marcus Denker, Damien Cassou and Kilon Alios *Pharo by Example* `http://pharobyexample.org/`

[4] Alexandre Bergel, Damien Cassou, Stphane Ducasse and Jannik Laval *Deep into Pharo* `http://www.deepintopharo.com/`

[5] *Pharo in a Nutshell*

[6] *Agile Visualization with Roassal*
`http://pharobooks.gforge.inria.fr/PharoByExampleTwo-Eng/latest/Roassal.pdf`

[7] Tudor Girba `http://www.themoosebook.org/book`

[8] Ian Sommerville *Software Engineering, 10th Edition*
`http://iansommerville.com/software-engineering-book/` *The MOOSE book*

[9] SCORE Agile Tweet Viz website `http://score-contest.org/2016/projects/tweetviz.php`

[10] Project Google Site `https://sites.google.com/a/asu.edu/project-geeko/`

[11] Taiga Virtual Visual Management
`https://tree.taiga.io/project/ser515asu-agiletweetviz-geekoh/`

[12] `http://agilemanifesto.org/`

[13] Ken Schwaber and Mike Beedle `Agile Software Development with Scrum (Series in Agile Software Development)`

[14] Agile TweetViz GitHub source `https://github.com/ser515asu/AgileTweetViz-GeekOh`