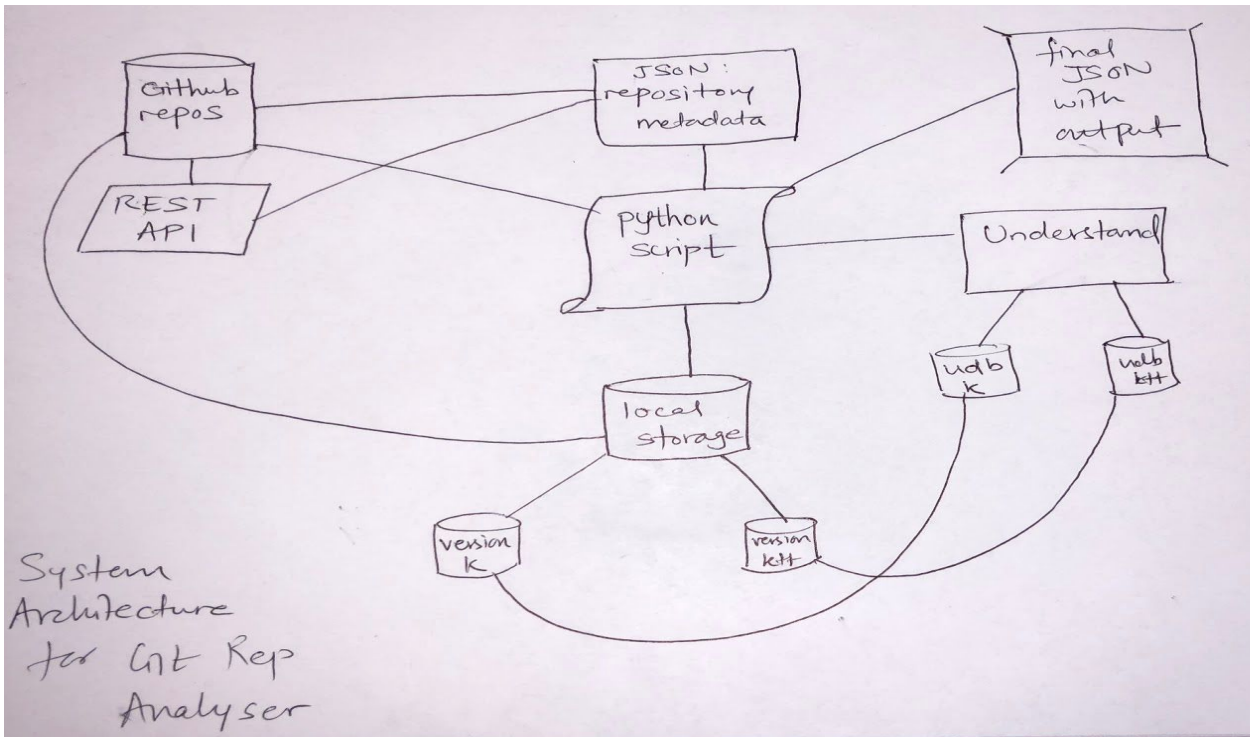
CS 540 Homework 2 - GitHub Repository Analyzer

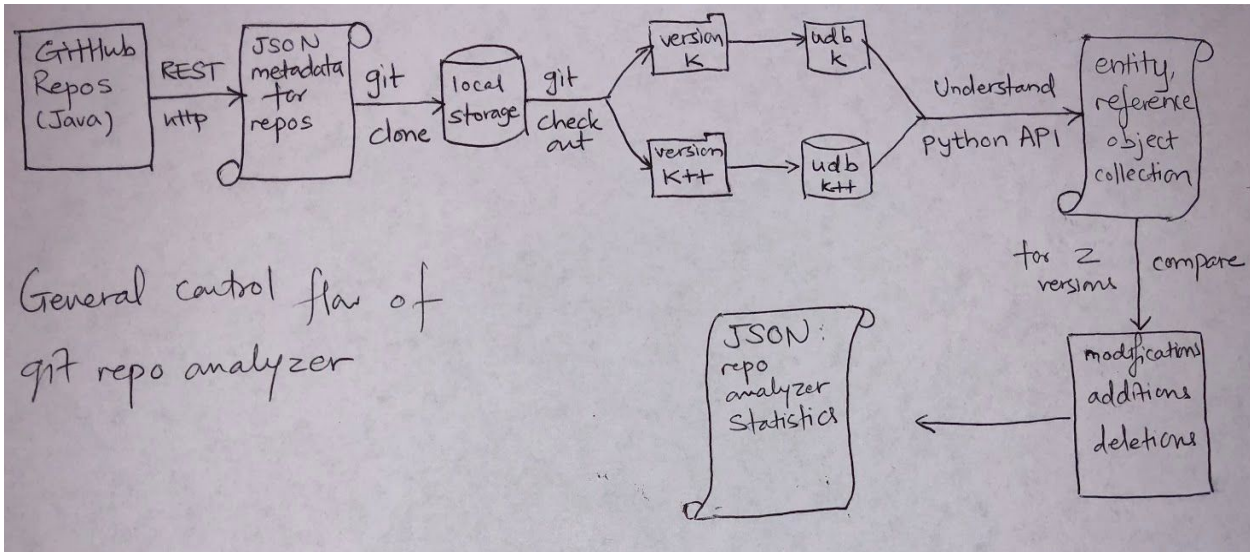
Pratik Kshirsagar(pkshir2@uic.edu), Sai Phaltankar(sphalt2@uic.edu)

Overview:
Our task was to create a GitHub repository analyzer for Java Projects. We had to use the Understand tool to analyze specifics of the source code such as Entities and References.

The following diagrams can explain our system architecture and also our general flow of control.

The second image describes: Program flow.
The first image describes: System architecture.





Software used:
Python v3.6.5, Understand v9

Steps:

Retrieve GitHub repository into local storage.
We achieved this via GitHub's REST API. We used the 'requests' package in Python to deal with HTTP requests. This was very similar to what we did in HW1.

Obtain all patch files for a repository.
We obtained all the patch files for Pull Requests which were merged with the master branch. The reason for this being, all the commit that correspond to these patch files can be easily accessed via "git checkout [SHA]". Commits from unmerged PRs would lead to unexpected behaviour of git checkout with some fatal errors hampering our program's behaviour.

Iteratively check out 2 successive versions of source code.
This was made possible by using the git checkout command applied to various commit SHAs. We parsed the patch files and obtained all of the pertinent commit SHAs for the source code. Using a for loop, we could achieve this sequential comparison of versions.

Create Understand databases for each version of the source code.
We had to obtain the entities and references from every version of the source code in order to compare and find any differences between them.

Creating an understand database allowed us to freely use it's rich Python API for dealing with entities and references.

Compare entities and references for each successive version of the source code.
This was a necessary step that allowed us to find the actual differences in the source code. Finding difference in entities, references let us know what kind of modifications took place in between 2 successive patches.

Create JSON file with reports of the repository analysis.
Comparing the contents of the source through Understand, we were able to pinpoint significant changes brought onto the code. Our focus was on finding the Changes in values of entities such as variables and parameters and their frequency as well as Addition/Deletion of entities in the source code.

Our JSON file would contain:

- The kind of entity that is updated over the patches.
- The change that it undergoes.
- Frequency of updates that belong to the same nature.

For our current system, we deal with only one specific Java repository, which can be found at
"https://github.com/structurizr/java"

This is so because we wanted to keep the system simple so as to understand it's functionality fully and tune it to the task, rather than overcomplicate it without achieving the fundamental objective. We have observed that this particular problem statement involves a lot of resource intensive computations so that using bulk data all at once would definitely have hurt the performance of the system.

Limitations: We are only working around a single Java repository due to performance constraints. Another limitation in that we have to download the data locally in order to perform analysis. Additionally, we couldn't use Entities and References to their full potential and extract more sophisticated patterns with Understand as the documentation provided by Understand was not so comfortable and hence our experience with it's Python API was not as rich as we'd liked it to be. The lack of learning resources for Understand available online definitely have affected the complexity and sophistication of our system.

Future Scope:
We would like to implement Machine Learning algorithms into our system for finding patterns more thoroughly. Also we'd like to work on the limitations as mentioned above.

IMPORTANT NOTES REGARDING IMPLEMENTATION:

- We use the inbuilt subprocess and os packages from Python to execute command line instructions through python script.
- We use the requests package to deal with http requests.
- We have used json and generateJson packages to create json file through python.
- We are assuming you have Understand already installed and set up for use.
- Please feel free to read comments in the code to get a better idea of our algorithm.