Software Engineering in Practice. My First Open Source Contribution

Ioannis Petropoulos - t8160107@aueb.gr

20/05/19

Abstract

The most important objective of the Software Engineering in Practice (SEiP) course was to familiarize with the real conditions of open source software development and to encourage the technologies used for this matter. In the meanwhile, that was a unique opportunity to co-operate with engineers that maintain the whole project through a Github repository and be a part of the development team.

Introduction

Open Source software is a trend that is picking up momentum [1] and that is something my university is taking into serious consideration. Personally, SEiP has been a springboard for my academic career. Under the supervision of professor Diomidis Spinellis and the guidence of PhD candidate Antonis Gkortzis, I managed to gather knowledge and gain experience that will make up a solid foundation for my professional evolution in the field of Software Engineering.

1   Project Understanding

This should have been the most critical phase of the whole process. Github offers a wide variety of projects to choose from, so you have to choose wisely and be very prudent. But once you do a thorough research, you should find something that fits you perfectly. And that's what happened with me.

Jarvis by Sukeesh is an open source project written in Python, inspired by Tony's Stark AI system [2] and available for Linux, MacOS and Windows. In a few words, Jarvis is a simple personal assistant which works on the terminal and he can do many things such as tell you the weather, find restaurants near you and help you out with many types of calculations.

To be able to understand the project structure, I dedicated a few days testing the features and reading the code and the contribution rules (checked out the PEP 8 guidelines for Python) in order to get familiar with it. In the beggining, chaos prevaled but as I was spending more time with it, it started becoming more clear. After 3 - 4 days, I was able to understand the code structure, so I thought that it's time to get my hands dirty.

2   Project Contribution

To the fun part now, it is time to write code. What I though would be a great idea for Jarvis was to create a plugin that Measures your Body Mass Index and outputs some valuable results.

2.1   Change Log

What I had in mind was an interactive BMI plugin that is easy to use and gives value to the results. Thus, I developed the plugin in Python in about 171 lines of code. Moreover, I refactored the health plugins packaging, due to the fact that they were not easily accessable by the user.

Figure 1: Jarvis contributors list by GitHub

2.2 Code Quality

This section will describe how I assured quality to my code.

2.2.1 Documentation

Documentation plays an important factor for software development. Every method I used was well documented so in this way, every future contributor would not struggle to understand what my code does. Below you can see an example of a documented method.

```python
def find_body_state(self, jarvis, calc):
    """

    According the bmi number, find_body_state finds out the state of the body
    and prints it to the user using colorama library for some coloring
    """
```

Figure 2: Flower one.

```python
def get_system(self, jarvis, syst):
    """

    get_system asks for the user to choose which system he wants to use
    1 for metric and 2 for imperial
    """
```

Figure 3: Flower two.

2.2.2 Linting Errors

My final commit to the project was devoted to code quality. Using a Python Library called pylint I did some code analysis on:

- Coding standard (variable names, PEP8 style guide, check imported modules usability)

- Error detection

- Refactoring duplicated code

Check for negative values on metrix system - bmi.py	Verified	124ad76
Add unittests for bmi plugin	Verified	dbb5f01
Separate health plugin	Verified	2fbe64e
Fix lint errors	Verified	c719e2f

Figure 4: Devoted commit on Linting Errors

2.3 Continuous Integration

Jarvis is using Travis CI for Continuous Integration. This part is very important, since the code that you contribute to a project has no meaning and is not going to be accepted if it can not be integrated with the rest of the code. Unfortunately, in the refactoring phase, the plugin of another contributor had some linting errors, so Travis was not able to pass the building phase. However, after contacting the administrator of the project, he undertook the issue and fixed it in one line of code. These things always happen, the important thing is to solve them immediately in communication with the developing team.

2.4 Testing

Coding in python, which is a scripting language, you have to write some unit tests in order to test the methods functionality. That's what I did and trust me when I say that the development team was so happy about that. Using the unittest library I wrote 5 different test cases for 2 of my methods.

3 Internal & External Communication

Luckily for me, the development team of Jarvis is very friendly and helpful. Each time I had a difficulty with the code structure, they responded very fast showing me the way through the solution.

3.1 Development Team integration

The first time I talked to this guys was through an issue I opened, discussing about a new feature I would like to see to Jarvis in the future.
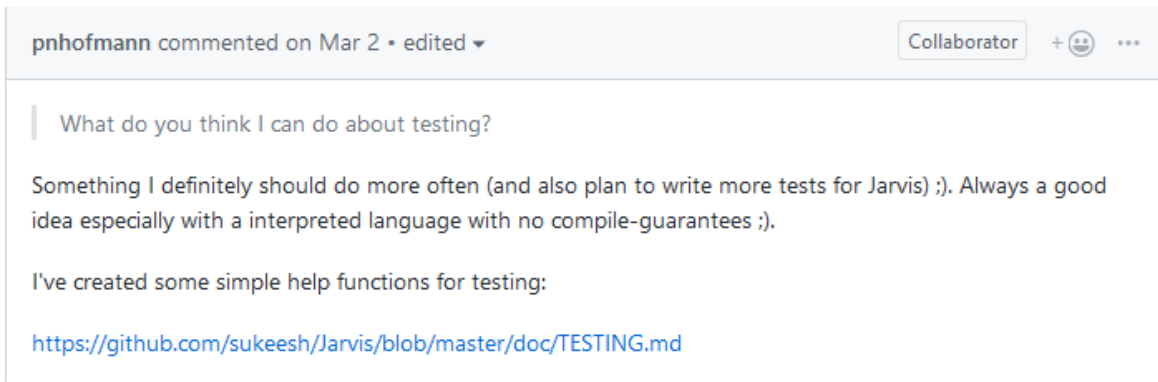
Figure 5: Discussing with DEV team about testing practices

3.2 Presentations

You can find both of my progress presentations on the GitHub repository:

ionpetro/Software-Engineering-in-Practice-Final-Report [3]

3.3 GitHub handling

Withing this course, apart from the coding, we gained valuable knowledge on Version Control Systems such as git. As you can see below, by the time I understood the value of such a tool, I haven't stop using it. Apart from that, I worked on windows-port branch (since I developed the plugin on Windows) and avoided to commit on master branch, which is basically used for production.
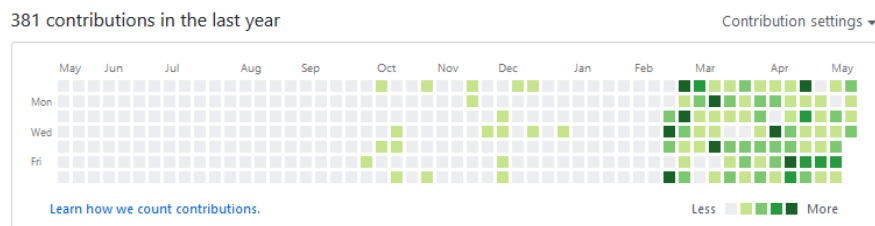

Figure 6: Github commits for 2019

3.4 Code Review

Finally, the administrator of the repository after reviewing my code finally merged the Pull Request to the original repository.
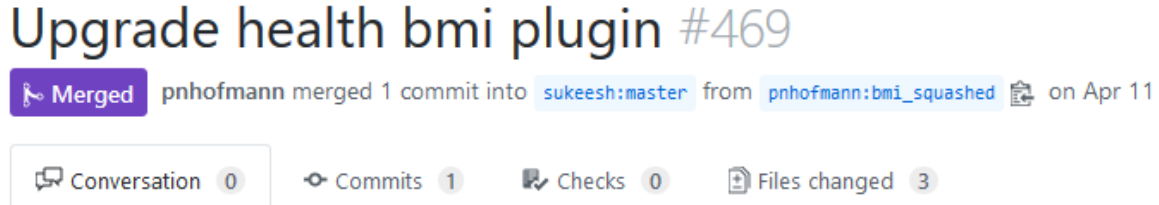

Figure 7: Merging PR to original repo

4 Conclusion

Contributing in an open Source Software was a valuable experience for me. Through this process I improved my coding skills, I met people who had similar interests and built public artifacts that helped me grow a reputation. Open source is a warm community is which people are coming back for years. **With this report, SEiP is coming to an end. What will stay though, is my willingness to contribute more to the open source community!**

References

[1] *Forbes - The trend to open source software and what it means for businesses and consumers.*
https://www.forbes.com/sites/richardfinger/2014/02/04/the-trend
-to-open-source-software-and-what-it-means-for-businesses-and-consumers.

[2] *J.A.R.V.I.S — Iron Man Wiki*
https://ironman.fandom.com/wiki/J.A.R.V.I.S.

[3] *Original Github Repository*
https://github.com/ionpetro/Software-Engineering-in-Practice-Final-Report