# Questions to Developers

Ao Dong

October 29, 2020

[A pilot will be conducted with these questions in order to test them. Test subjects may include PhD students. —PM]

## 1 Information about the developers and users

[Jegatheesan, 2016]

1. Interviewees' current position/title? degrees?
2. Interviewees' contribution to/relationship with the software?
3. Length of time the interviewee has been involved with this software?
4. How large is the development group?
5. Do you have a defined process for accepting new contributions into your team?
6. What is the typical background of a developer?
7. What is your estimated number of users? How did you come up with that estimate?
8. What is the typical background of a user?

## 2 Information about the software

[3 aspects: dev process, software quality, user feedback —AD]

[consider adding 'What, if any, actions were taken to address x?' to most/all questions, similar to question 10 —PM] [questions 1,3,4,5,6 build to visibility/transparency but a question would need to be modified to include something along the line of 'has this information been added to the documentation?' or as above asking about actions —PM]

1. Currently, what are the most significant obstacles in your development process? [visibility/transparency]

2. How might you change your development process to remove or reduce these obstacles?

3. How does documentation fit into your development process? Would improved documentation help with the obstacles you typically face? [traceability, visibility/transparency]

4. In the past, is there any major obstacle to your development process that has been solved? How did you solve it? [visibility/transparency]

5. What is your software development model? For example, waterfall, agile, etc. [visibility/transparency]

6. What is your project management process? Do you think improve this process can tackle the current problem? Were any project management tools used? [visibility/transparency]

7. ~~Can you name several software qualities that are the most important to this project?~~

8. Was it hard to ensure the correctness of the software? If there were any obstacles, what methods have been considered or practiced to improve the situation? If practiced, did it work? [correctness]

9. When designing the software, did you consider the ease of future changes? For example, will it be hard to change the structure of the system, modules or code blocks? What measures have been taken to ensure the ease of future changes and maintains? [maintainability, modifiability]

10. Provide instances where users have misunderstood the software. What, if any, actions were taken to address usability issues? [usability, understandability] [should this be split into two questions? —PM]

11. Do you think the current documentation can clearly convey all necessary knowledge to the users? If yes, how did you successfully achieve it? If no, what improvements are needed? [unambiguity]

12. Do you have any concern that your computational results won't be reproducible in the future? Have you taken any steps to ensure reproducibility? [reproducibility]

# References

Thulasi Jegatheesan. Case studies in document driven design of scientific computing software. Master's thesis, McMaster University, Hamilton, Ontario, Canada, July 2016.