How to communicate to get your patches accepted in Linux kernel community

Dear Sir or Madam:
I am a researcher from [redacted]. My colleague and I are investigating 'How to communicate to get your patches accepted in Linux kernel community'.
According to our previous research on Linux kernel developers, almost all the respondents (97.68%) think communication is important during patch review via email. Even though it has been realized that giving a proper description of the patch is essential to patch acceptance, it is not easy to do it well. More than half (52.38%) of the respondents fell it is difficult to write a good description. Worse still, a lot of valuable efforts of both developers and reviewers are wasted just because the patch is not described well. We found 59.52% of the developers had their patches rejected and 73.08% of the reviewers rejected patches ever.
In order to help the developers to communication effectively during patch submission, we discovered 18 (best) practices through mining the mailing-list interactions. The purpose of this questionnaire is to understand the differences in the importance and operating difficulty of these best practices. The importance means the extent to which this practice may help the reviewers (or the community) to understand and therefore accept the patch. The operating difficulty is the amount of effort required to follow this practice or the possibility of forgetting to do so.

1-5 means: 1--Not important (difficult); 2--Slightly important (difficult); 3--Medium important (difficult); 4--More important (difficult); 5--Very important (difficult)

***1. What kind of contributor are you?**

- Only write code
- Review code
- Write code and review code

A. Every developer should follow the following practices, which are related to 'Regulation of the community'.

***2. Start communication at right time. (Reserve sufficient review time for the patch (except fixing bug) before merge window closed.)**

Importance	1	2	3	4	5
Difficulty	1	2	3	4	5

***3. Separate each logical change into a separate patch.**

Importance	1	2	3	4	5
Difficulty	1	2	3	4	5

***4. When describing what you did, use imperative mood (e.g. don't use 'This patch ...').**

Importance	1	2	3	4	5
Difficulty	1	2	3	4	5

)

*5. **Pay attention to the rules in the community, such as the formatting requirement.**

Importance	1	2	3	4	5
Difficulty	1	2	3	4	5

B. All types of patches should follow the following practices, which are related to 'The type of patch' and 'Implementation of patch'.

*6. **Try your best to describe your motivation, including why you did it and what you did.**

Importance	1	2	3	4	5
Difficulty	1	2	3	4	5

*7. **If the implementation is complex and it requires reviewer to spend a lot of time to understand it, briefly describe the implementation details.**

Importance	1	2	3	4	5
Difficulty	1	2	3	4	5

*8. **If there is an improvement after applying the patch, clearly describe the improvement.**

Importance	1	2	3	4	5
Difficulty	1	2	3	4	5

*9. **If there is a cost after applying the patch, provide the trade-off.**

Importance	1	2	3	4	5
Difficulty	1	2	3	4	5

*10. **If the patch refers other information, e.g., commits, email content, the view of the expert, or links of related information, include it.**

Importance	1	2	3	4	5
Difficulty	1	2	3	4	5

*11. **If the patch is a prototype or on-going, include the limitation of the current version.**

Importance	1	2	3	4	5
Difficulty	1	2	3	4	5

*12. **If the patch has been tested or reviewed by others, use tags such as 'Reviewed-by', 'Tested-by'.**

Importance	1	2	3	4	5
Difficulty	1	2	3	4	5

***13. If there is more than one way to implement it, showing the superiority of your solution is much better.**

Importance	1	2	3	4	5
Difficulty	1	2	3	4	5

C. The practices for certain types of patches, which are relate to 'The type of patch', 'Implementation of patch', 'Human factors'.

***14. If the feature is relative complex or it is very important but has not been recognized by the community, provide sufficient reasons why this new feature is required.**

Importance	1	2	3	4	5
Difficulty	1	2	3	4	5

***15. If the feature has been discussed previously, or the feature is a widely recognized feature, you may omit providing the reason. What do you think of this practice?**

It's significant, because it can save developers' effort.

It's not significant. _____

It's meaningless. _____

Please give the reason.

***16. If the new feature includes several functions, list all of the functions or give a brief introduction.**

Importance	1	2	3	4	5
Difficulty	1	2	3	4	5

***17. If the patch is bug fix, describe the scenario of bug or how to reproduce it, or the issue report.**

Importance	1	2	3	4	5
Difficulty	1	2	3	4	5

***18. Especially if the bug has serious consequence, e.g. memory leak, system crash, including the seriousness of bug can bring more attention from reviewer.**

Importance	1	2	3	4	5
Difficulty	1	2	3	4	5

***19. For the patches such as: easy case, fix bug, simplify code, add driver, because the reason for submitting these types of patch is very clear, just express what you did which has already reflected the reasons. What do you think of this practice?**

It's significant, because it can save developers' effort.

It's not significant. _____

It's meaningless. _____

Please give the reason.

20. Are there any other practices that you consider important but we missed? If there are , please provide.