791-C EMPIRICAL METHODS IN SOFTWARE ENGINEERING AND COMPUTER SCIENCE

HW # 2

Assessment of "Quantitative Analysis of Faults and Failures in a Complex Software System"

Ekrem Kocaguneli

2/4/2010

I. Brief discussions on the paper:

1. Why is this empirical study identified as a case study? (10 points)

The characteristics of a case study are: It is more of an observation, there is no control on the field, they scale well, they are easier to plan. For each characteristic that is listed above, I will refer to the paper in the bullets below.

- It is more of an observation: They observe the characteristics of release n and n+1
- There is no control on the field: They do not do controlled experiments, they merely collect the available data from two releases.
- They scale well: The case study they did scales to the projects under review perfectly, since they were aiming to carry the test case on these projects in the first place.
- They are easier to plan: The planning was relatively easy for that case study. All they needed to do was to define the appropriate hypothesis and derive the required data and analyze it for the defined hypothesis. They did not have to plan for the environmental factors etc.

Furthermore, to make sure that the conducted study is a case study, there are 7 steps that we need to follow. These steps and how they were followed in the paper are as follows:

i. **Define hypothesis:** Fenton et. al. have rigorously defined 8 hypothesis that have guided their study. They group these hypothesis under 4 headings:

1) Hypothesis relating to Pareto principle

2) Hypothesis relating to the use of early fault data for the prediction of later fault or failure data

3) Hypothesis about metrics for fault prediction

4) Hypothesis relating to benchmarking figures for quality

ii. **Select pilot projects:** According to Kitchenham et. al. there are two main points of consideration while selecting pilot projects:

1) Representativeness: The projects selected for a case study shall be representative projects of the company or organization. Fenton et. al. have selected is a *legacy project developing switches*, which is a typical project for a telecom company like Ericsson. Furthermore, they have selected two consecutive releases of the same legacy project, which makes sense because their comparisons will be based on the same project, so that the previous release will be representative of the next release and they will have similar characteristics.

2) Describing projects in terms of significant characteristics: Although authors frequently talk about the confidentiality issue associated with their study, they provide extensive project characteristics:

- **Development Centers:** 20 centers in more than 10 countries

- **Module number:** 140 modules for release n and 246 releases for release n+1
- **Size of modules:** Size of modules range from 1000 to 6000 LOC
- **Dependent variable and its phases:** Number of faults in 4 phases (function test, system test, site test, operation test)
- **Classification of faults by company:**
    a. Fault has already been corrected
    b. Fault will be corrected
    c. Fault requires no action
    d. Fault was due to installation problems
- **Independent variables:** Lines of code, cyclomatic complexity etc.

iii. **Identify methods of comparison:** Kitchenham et. al. proposes 3 different methods for comparison of methods: Selecting a sister project, comparing results against a company base and applying it to some components at random. Fenton et. al. have chosen the second one. Namely, comparing results against a company baseline. In their study they take project n as a company baseline and compare results from project n+1 to that company baseline.

iv. **Minimize the effect of confounding factors:** The type of faults that were classified by the company is a confounding factor, since each type of fault may behave differently. The authors have chosen to use the faults that are classified as *"faults that will be corrected"* so that they have eliminated the confusion due to confounding factor of fault type.

v. **Plan the case study:** The authors have planned the case study in a detailed fashion. In the first 2 phases, they describe the case study environment and present their hypothesis and explain why they have planned to define these hypotheses. Furthermore, in the data section, they first identify the data characteristics and then present their plan of selecting and using the data subject to their proposed statistical tests.

vi. **Monitor the case study:** They have consulted the company staff while making decisions such as fault type categorizations. Furthermore, they have checked the internal documentation to make sure that classification of faults is reliable. Also they compare their results for the hypotheses they have to previous results and command on them and bring an explanation if there is a divergence from the previous results. Therefore, they were monitoring their results from the design till results phase.

vii. **Analyze and report results:** For each hypothesis they have presented their results under related heading.

2. Is there any method of comparison used in this study? What you would suggest as an appropriate method for comparison?

- The authors have placed a discussion of analysis techniques they adopted and they have explained why they have adopted Alberg diagrams for comparison instead of widely used t-test. As they claim, Alberg is more useful in their case, since Alberg only assumes that the data is at least ordinal and also Alberg allows them to identify type I and type II errors at the same time, for different discriminative thresholds.
- Authors also use normalization while comparing results to eliminate the bias that could be induced due to numerical differences.

3. Are there any confounding factors considered in this case study?

- One of the confounding factors considered is the classification of fault. The authors have identified 4 different classes (fault has already been corrected, fault will be corrected, fault requires no action, fault was due to installation problems) and have chosen to use faults that were identified as *"will be corrected"*.
- Another confounding factor is the occurrence of faults. In terms of occurrence, the authors have identified the faults as either pre-release or post-release.
- A third confounding factor is the module size and the complexity that is pushed into the representation layers, in the sense that if the module size is too small, then the complexity is pushed into presentation layers and the faults stemming from these layers are equally distributed to small and large components.
- Post-release defect density and the testing effort spent on each component are confounding factors. There is no data available in the study regarding the testing effort. Therefore, the claim that high pre-release defect density means low post-release defect density may simply be due to different testing efforts spent on each component.

4. Come up on your own with at least one example of confounding factors related to any of the hypothesis explored in this paper. Justify your answer. Suggest a way to eliminate the confounding factors.

- **Hypothesis**: 1b, which is "if a small number of modules contain most of the faults discovered during pre-release testing then this is simply because those modules constitute most of the code size".
  Confounding factors: The size measure LOC and the method that is used to map faults to components/modules may form a basis for confounding factors.
  Justification: One of the most important confounding factors to me is the origin of the fault and how a particular fault was associated with modules that are under discussion in this study. The authors discuss the fact that very small modules may push the complexity into representation layers; however, they suffice with saying that such faults stemming from representation layers are equally distributed to modules of different sizes. Therefore, this makes the ***LOC*** as a measure of size and the ***fault mapping*** confounding factors. Because it is not clear how a presentation layer fault is mapped to functions. Furthermore, a function whose size is small in terms of LOC may have a significant size

in terms of function points and it may have significant impacts on the presentation layer components.

Solution: Instead of using LOC, function points could have been utilized and a discussion on the fault mapping process could have been added to paper. In that case, we could have a better understanding of size as well as how influential a component was on the presentation layer, as the function point analysis also includes representation layer information.

II. Comment on the rigorousness of the case study. Identify at least one specific aspect of this study that you would improve. Describe what would be the benefit of the improvement.

i. Possible Improvement: All the graphs regarding the code artifacts as well as regarding the fault distribution have been given only for release n+1. However, it is very trivial to add release n. I think updating the figures and putting release n next to each figure of release n+1 would be a possible improvement.

- **Benefit of the possible improvement:** This way, the reader could have a better idea of how the code artifacts as well as the error ratios have changed from one release to another. Furthermore, such an approach would remove the ambiguities regarding release n. Because, when we only show the artifacts regarding release n+1, then we are also letting the possible bias into our study that would prompt people to ask questions, whether the observed facts are also influenced by artifact changes in two releases.

ii. Strong points:

- The study follows the guidelines for case studies very closely.
- A very detailed hypothesis statement approach has been adopted and each hypothesis has been explained and justified in its related section. Furthermore, enough statistical tests have been performed.
- In conclusion, each hypothesis that was presented throughout the paper have been revisited and summarized together with their results. This is a very clever way to wrap up things and make the final remark.

iii. Weak points:

- Different module sizes in two different releases: 140 in first and 246 in second release. The reason behind the difference is the fact that their data collection process has improved. However, they could have gone back and applied a similar procedure to the first release too. The difference in sizes of train set may introduce bias, although the selection is random.
- In section 3.1.1., while explaining hypothesis 1.a, the authors say that for project n+1, similar results are obtained but not shown here. Why not showing here? It is very easy to just add another graph, next to the graph of project n. I think such a hiding approach induces ambiguity.