Paper Reviews.

Code Review Metrics

- Paper Citation
  Code Review Metrics - OWASP. (2016). Web.archive.org. Retrieved 28 May 2016, from http://web.archive.org/web/20151009202719/https://www.owasp.org/index.php/Code_Review_Metric
- Summary
  Give a brief view of what code review is and divided the code review metrics into two distinct classes, relative class and absolute class. Then give a definition of each of the class. Then the advantages of metrics are states. Then some of the secure development metrics just like Defect Density and Lines of code are listed and give a general introduction. At last this article analysis the metrics and give some standard to evaluate the quality of code review is useful or not.
- Strengths
  The advantage of using code metrics is good and can be used in the essay. It doesn't talk just theoretically but have use example for explanation. It states some specified metrics including the LOC we used in this semester and analysis why this is useful.
- Weaknesses
  This article not compare the differences between different metrics and doesn't talk about anything about the code review after the compiling. LOC is one of the metrics I used in this years, it have some information but not enough for it, need more. This is a website and be modified by several times and have no references.
- Other comments
  Not sure if this can be used because of the unreliable, but according to the other articles I read and my own experiences, this article seems to be fine.

Is lines of code a good measure of effort in effort-aware models?

- Paper Citation

  Emad Shihab, Yasutaka Kamei, Bram Adams, Ahmed E. Hassan, Is lines of code a good measure of effort in effort-aware models?, Information and Software Technology, Volume 55, Issue 11, November 2013, Pages 1981-1993, ISSN 0950-5849, http://dx.doi.org/10.1016/j.infsof.2013.06.002.
  (http://www.sciencedirect.com/science/article/pii/S0950584913001316)

- Summary

  This paper, the Lines of Code (LOC) is introduced in details and analysis this kind of code review metrics. After doing some research between LOC and other code and complexity of metrics, they found that the LOC have lower correlation comparing to others in all four experiments they have done. And they found that the code metrics have better performance than LOC. They also introduce the code metric and churn metric. The experiment is based on JIRA issue database and mainly try to find out the issues and bugs happened in the files and try to predict it. Even though the prediction is not including in this assignment but the code review metrics analysis has been used as the same way.
- Strengths

  They have find out a lot of code metric that can be used such as LOC, Cyclomatic Complexity Density, Coupling between Objects and Response of a Class. They use all these metric they found to do the experiments and draw a solid conclusion and give a chart to demonstrate the results.
- Weaknesses

  This article is not mainly about code review, but talk about the analysis of LOC compared to other code metrics. This article also talk about the prediction of issues, even though the use of the LOC and other metrics is same in the code review but maybe there will be something different I cannot recognized.
- Other comments

  Can use the LOC part as the way to use code review and the analysis can be used as a proof that the code review works well.

Achieving Code Quality using Code Review System

- Paper Citation
  Bhosale, S., Bhosale, S., Bhalerao, V., & Bhagwat, K. (2012). Achieving Code Quality using Code Review System. Emerging Trends In Computer Science And Information Technology, 25-26.
- Summary
  Introduce what is code review and introduced the code views aims. The author stats that the common purpose of code review are 1. To build a no-defect and well-document software. 2. To make the software compiles good. 3. To communicate with other developers to improve the skills. This article also introduce some code review techniques such as Iterative Code Review (ICR). Several methods that can make the code review agile is also introduced. Over the shoulder, Email pass-around, Pair Programming are also included. The proposed system also introduced such as bugnizer, code review system and helpdesk system.
- Strengths
  The definition and advantage of code review is well introduced. The purpose is also summarized into three main purpose. These can be used in the introduction and background.
- Weaknesses
  Only talk about the code review in the whole process but not in specific PSP phase. Have not talking about any differences between no-compiled and compiled.

The Impact of Design and Code Reviews on Software Quality: An Empirical Study Based on PSP Data

- Paper Citation
  C. F. Kemerer and M. C. Paulk, "The Impact of Design and Code Reviews on Software Quality: An Empirical Study Based on PSP Data," in IEEE Transactions on Software Engineering, vol. 35, no. 4, pp. 534-550, July-Aug. 2009.
  doi: 10.1109/TSE.2009.27
  URL:
  http://ieeexplore.ieee.org.proxy.library.adelaide.edu.au/stamp/stamp.jsp?tp=&arnumber=4815279&isnumber=5186358
- Summary
  This paper mainly describes the quality of the software using code review. It talk about code review in the PSP and introduce the different phase of PSP and states the life cycle model of the software quality and point out the reviews is important. The modes is general can be conclude as four steps: Production activity, initial work product, review, related work. Then the PSP data sets data is provided to do an experiment. They use a quality model, and operational variables to do the experiments and draw a conclusion. They found out that some statistic variables have no significant influence on the quality such as years of experience, programming language, number of assignment completed by each developers. They also found out that the PSP is useful when try to improve the quality of a software developed in industry.
- Strengths
  Give back ground of PSP, and analysis different phase the code review as a different roll. It also analysis the code review between PSP and none-PSP developers. What is more, the author have found the result when this comes to industry.
- Weaknesses
  The developers only have 243 contribute to this experiment. What about the developers new to PSP and not reported? The whole experiment is based on C++, what about other language? Even though the author find this not influence much but it is from other studies and not specified the details.
- Other comments
  Really useful materials.

Improving PSP education by pairing: An empirical study

- Paper Citation

  G. Rong, H. Zhang, M. Xie and D. Shao, "Improving PSP education by pairing: An empirical study," *Software Engineering (ICSE), 2012 34th International Conference on*, Zurich, 2012, pp. 1245-1254.

  doi: 10.1109/ICSE.2012.6227018

  URL: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6227018&isnumber=6227015

- Summary

  This article give a research on comparing how solo and pair student doing PSP course. And the result show that the paired student have better performance than solo student in most of the time studying PSP. The performance include following the course process, the quality of the product, the corrective to describe the issues in the log. But the time to finish the examination is almost the same between paired and solo students.

  The author also states that the PSP can give student the developing skills and Paired Programming (PP) should be introduced to PSP course in case of the higher improvement on skills, quality.

- Strengths

  This article focus on a special part about PSP: Paired & Solo Programming and give a comparison. The data collected in two years and have a large volume. The result is show in charts, diagrams and graphs and this make the result more convinced.

- Weaknesses

  The data collected is just from one university in a specific province of China. This make the data is not stand for all but just Chinese and in a small area of China.

Improving software developer's competence: Is the Personal Software Process Working?

- Paper Citation
  Abrahamsson, P., Kautz, K., Sieppi, H., & Lappalainen, J. (2002) Improving software developer's competence: Is the Personal Software Process Working? Presented at Workshop on Empirical

  Cite as: **arXiv:1311.0228 [cs.SE]**
- Summary
  The article give some useful opinion: 1) The PSP is an well know technique to improve software process in individual and will secure the quality. 2) PSP aim to make the software engineering project more predictable and in good quality.
  The article give a research about 58 students in 3 different university studying PSP. And result shows that only compile is significant improved. Author conclude this result as PSP can't significant increase the software engineer's size and time estimation skills but can improve the quality of the product.
- Strengths
  This article use the real research and data to conclude a result that can make people convinced. Using graph and chart to support its opinion make the result can be clearly seen.
- Weaknesses
  The research data is too small, 58 students is not an appropriate data source even though the students are from three different universities.
- Other comments (optional)
  This article's result can be partly believed, even though the data source is in a small volume but still got some evident to proof that the PSP have benefit to the Software Engineer but not so significant.