Week 3 Journal

Rodger Byrd

I. Topic Map

For my area I'm looking at Anti-Patterns in code, these are also known as code smells. I've also seen them referred to as Atoms of Confusion and nano patterrns. My topic map is included below in figure 1. My thoughts on gaps are some method to demonstrate the found anti-patterns are valid. Also, there seems to be a gap in how to fix anti-patterns and code smells. I've also included my notes on creating the Topic map in Figure 2.

II. Notes on Survey Papers,

For my first detailed read I chose a paper called *A survey on software smells* [1]. The following are my raw notes for this paper.

Kent Beck coined the term code smell, I should look at this paper. Contains some ideas for topic map in abstract. Figure 1 in paper has a pretty detailed topic map. Interesting for comparison. Table 4 code smells. Interesting section on whether anti-patterns and smells synonyms, doesn't seem to be consensus. Some papers yes, some no. Smell indicator of problem, anti pattern definitive problem. Current detection methods cause too many false positives? This wasn't as good as the first two papers, they did a very good job of summarizing the current research. This one talked way too much about how they did their analysis instead of the results.

For my next detailed read I chose a paper called *A systematic literature review: Refactoring for disclosing code smells in object oriented software* [8]. The following are my raw notes for this paper.

Refers to code smells and anti-patterns interchangeably. Related work is basically a background. Included 238 of an initial 1053 articles. Wow, that is a ton of papers. The papers they drew from were on the following topics: Refactoring, anti-patterns, code smells, Object Oriented Design and refactoring, fault tolerance and OOD, SW Metrics and anti-patterns, anti-patterns in cross company projects. Provide a lot of detail on the detection methods Interesting to show charts based on the publisher, Conference proceedings and IEEE were top sources.

For my next detailed read I chose a paper called *Smells in software test code: A survey of knowledge in industry and academia* [5]. The following are my raw notes for this paper.

Interesting variation on the idea of anti pattern: "test smells". Poorly designed tests. Really weird figures in this paper, copy paste screenshots out of spreadsheet and websites like google. Just make a table! They are missing the visualization, they are showing tons of tables instead of the results of the research. Missing the analysis?

The files for this latex document are in the github repository located at `https://github.com/rodger79/CS6000`

Papers that were scanned and trashed are referenced in the bibliography below.

Fig. 1. TopicMap

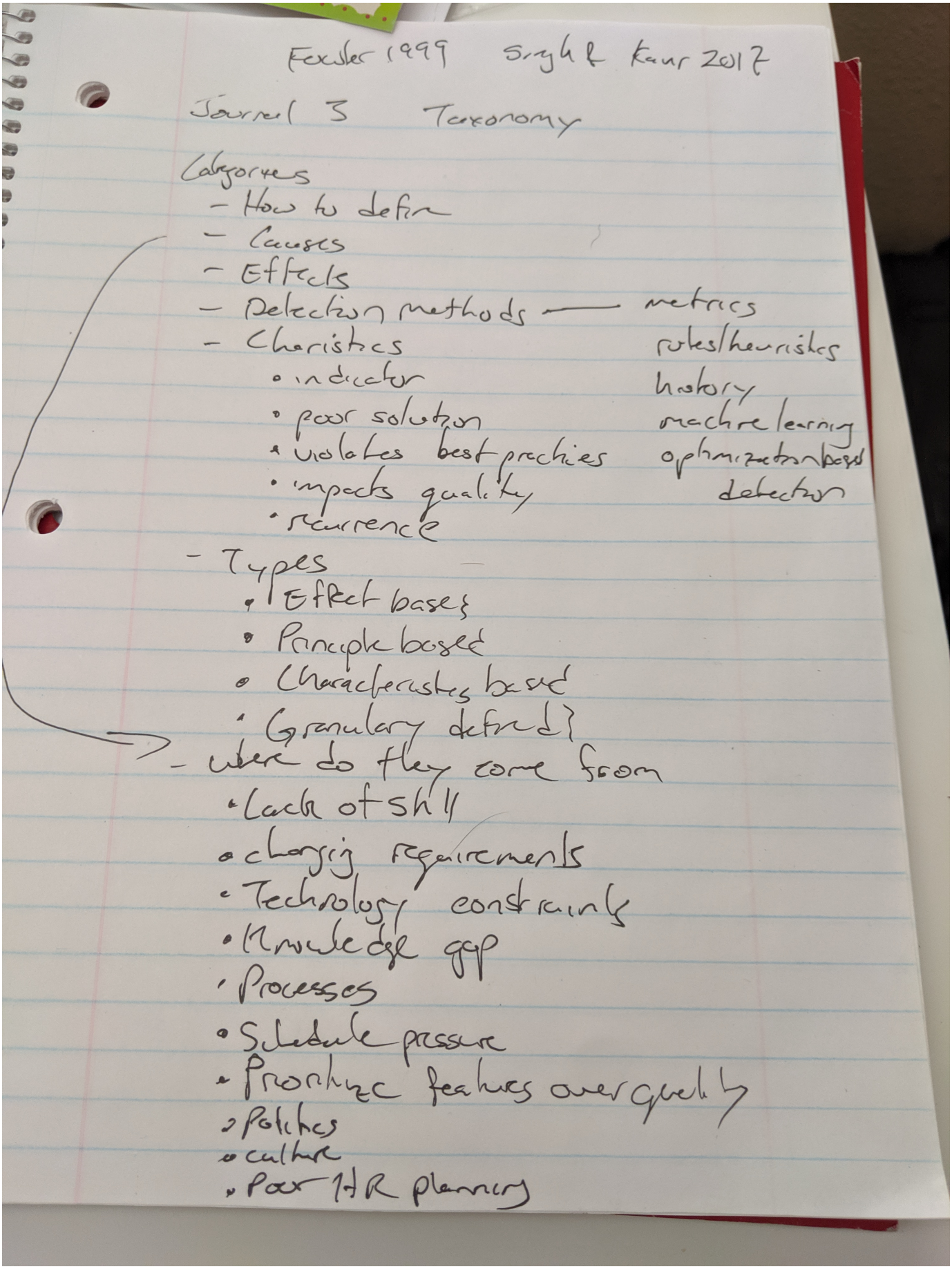

Fig. 2. TopicMap

REFERENCES

[1] T. Sharma and D. Spinellis, "A survey on software smells," *Journal of Systems and Software*, vol. 138, pp. 158 – 173, 2018. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0164121217303114

[2] M. Mantyla, J. Vanhanen, and C. Lassenius, "A taxonomy and an initial empirical study of bad smells in code," in *International Conference on Software Maintenance, 2003. ICSM 2003. Proceedings.*, Sep. 2003, pp. 381–384.

[3] R. Arcoverde, A. Garcia, and E. Figueiredo, "Understanding the Longevity of Code Smells: Preliminary Results of an Explanatory Survey," in *Proceedings of the 4th Workshop on Refactoring Tools*, ser. WRT '11. New York, NY, USA: ACM, 2011, pp. 33–36, event-place: Waikiki, Honolulu, HI, USA. [Online]. Available: http://doi.acm.org/10.1145/1984732.1984740

[4] A. Yamashita and L. Moonen, "Do developers care about code smells? An exploratory survey," Oct., pp. 242–251.

[5] V. Garousi and B. Kk, "Smells in software test code: A survey of knowledge in industry and academia," *Journal of Systems and Software*, vol. 138, pp. 52 – 81, 2018. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0164121217303060

[6] N. Yoshioka, H. Washizaki, and K. Maruyama, "A survey on security patterns," *Progress in Informatics*, no. 5, p. 35, Mar. 2008. [Online]. Available: http://www.nii.ac.jp/pi/n5/5_35.html

[7] D. Gopstein, J. Iannacone, Y. Yan, L. DeLong, Y. Zhuang, M. K.-C. Yeh, and J. Cappos, "Understanding Misunderstandings in Source Code," in *Proceedings of the 2017 11th Joint Meeting on Foundations of Software Engineering*, ser. ESEC/FSE 2017. New York, NY, USA: ACM, 2017, pp. 129–139, event-place: Paderborn, Germany. [Online]. Available: http://doi.acm.org/10.1145/3106237.3106264

[8] S. Singh and S. Kaur, "A systematic literature review: Refactoring for disclosing code smells in object oriented software," *Ain Shams Engineering Journal*, vol. 9, no. 4, pp. 2129 – 2151, 2018. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S2090447917300412