Week 2. Assignment

UCCS CS 6000 Computer Science Research

Jose Luis Castanon Remy

I. Research direction

During my last working experience, I always had the idea that our security systems were in place but not in the correct way. Similar to a wind power plant, the plant is placed in a windy area but if the back of the fan is not correctly placed facing the wind, the blades are not able to generate any movement and as a consequence, no electricity is being drawn from the wind.

Somehow our systems were there, we were able to log information from different sub-systems, but we were not focusing our efforts on the areas that needed more time (at a first glance there was a lot on our plates). For instance, we were not analyzing the collected data from logs and other systems. I always had the feeling that all the security systems in place were not able to give us the real ”whole picture” of our security status. We had different tools giving us different metrics in different places. During my time there, my duties did not let me think about the critical problem behind all that ”confusion”. In addition to this, we were not able to map all the assets that needed protection. Most of those were newly discovered after we had our tools and myself analyze the architecture and the systems built within the company.

I feel that companies should have a way to centralize metrics as well as compound a list of assets to be used in different security processes. One of the questions that come up is, What metrics need to be centralized?

Related to this initial idea, comes up the following question, What metrics can show us the real ”whole picture” of our security status?

In order to get a proper understanding of the initial direction, I should not be just mapping software but assessing software qualities that can give us a valuable metric pool to measure within the software.

To address the issue, my advisor gave me a hint to research software (SW) maintenance and maintainability. SW maintenance/ maintainability can lead to specific aspects within SW that can lead to a better understanding of security issues, help detect areas with flaws, etc.

I have compiled the following list of papers: [1], [2], [3], [4], [5] and [6]. Most of the notes are within the printed version of the 4 papers that I have chosen to read. I will be adding the keynotes to the bibliography.

II. Reading

From the list of papers that I have selected to read, 3 of them are related to software maintainability and one of those is related to software maintenance. Before getting to the analysis, I should be reading more about software maintenance. I feel that I need to research a little bit more in that direction.

Within the selected papers, published years in descending order are 2021 [2], 2020 [4], [1] and 2016 [3]. I have read a survey paper about software maintenance from 2009, [6], which I found to be very helpful in order to get myself older trends and perspectives.

The most cited paper, currently 55 citations, based on Google Scholar information, is related to software maintainability prediction [1]. I think it is a highly cited paper because it contains very helpful components within its organization. It is the only paper that I have read that contains a good process review diagram. There is a data extraction analysis that is directly related to the research questions. I consider this is very helpful. It is introducing in a simple way the main ideas that the researchers have encountered during the literature review process. In general, I got the feeling that everything within this paper is helpful. There is nothing that I feel is useless in this paper. I have not had this ”feeling” with the rest of the papers that I have read.

Related to the concepts that were introduced in class, This paper is able to answer all the Ws. The paper is correctly structured and there are relevant details that help me narrow some of my initial research questions.

The rest of the papers with lower citations, are either too focused on machine learning methods and prediction algorithms, which I do not feel are going to be helpful right now, or these papers contain useless/ non-structured information. Most of those fail to answer the ”for whom” questions specifically to me.

III. Topic map

I have built the following topic maps. The first one is the map of the topics coming out from the papers that I have decided to read. These topics are repeated and color-ordered.

The second map is an organized map of the first one. It is organized by topic, software maintenance or maintainability, and the topics that came out from the previous topic map.

I have not included in the topic map the gaps that exist. Honestly, I need to read more. I can state that no one is writing about the importance of maintainability within vulnerable software. There is also no one speaking about the influence of maintenance as a metric to be centralized within software-based companies. I think these gaps can be addressed in the future. I will be adding those gaps in the future, as long as I am able to properly define and state those. I feel that adding something that might not be what I am looking for can distract me from my path.

References

[1] H. Alsolai and M. Roper, “A systematic literature review of machine learning techniques for software maintainability prediction,” *Information*

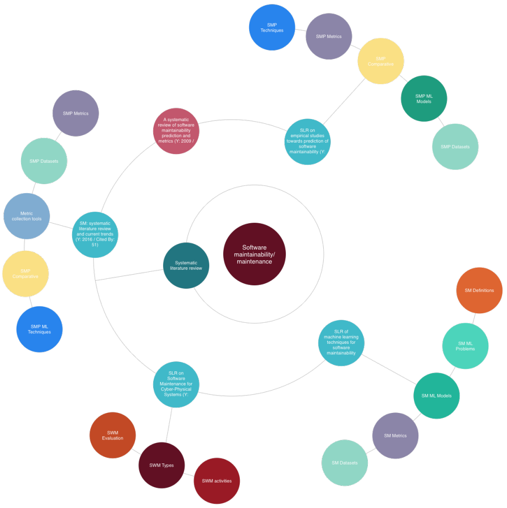
SMP Techniques
SMP Metrics
SMP Comparative
SMP ML Models
SMP Datasets
A systematic review of software maintainability prediction and metrics (Y: 2009 /
SLR on empirical studies towards prediction of software maintainability (Y:
SMP Metrics
SMP Datasets
Metric collection tools
SM: systematic literature review and current trends (Y: 2016 / Cited By: 51)
SMP Comparative
SMP ML Techniques
Systematic literature review
Software maintainability/ maintenance
SM Definitions
SM ML Problems
SLR of machine learning techniques for software maintainability
SLR on Software Maintenance for Cyber-Physical Systems (Y:
SM ML Models
SM Metrics
SM Datasets
SWM Evaluation
SWM Types
SWM activities

Fig. 1. Topics per paper

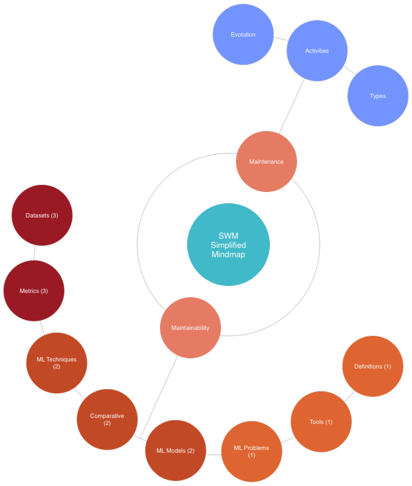

Fig. 2. Topics

and Software Technology, vol. 119, p. 106214, 2020, classification internal and external attributes, search process note the libraries and publishers, process review fig. 2, table 3 measurements and metrics RQ1 and RQ2, table 4 interesting comparison, maintainability cannot be measured directly, table 5, fig.7, read more about evaluation measures in fig. 13.

[2] N. Khezami, M. Kessentini, and T. D. N. Ferreira, "A systematic literature review on software maintenance for cyber-physical systems," *IEEE Access*, 2021, interesting reasearch questions, definition of cyber-physical system is too broad, paper reference per category and percentage.

[3] R. Malhotra and A. Chug, "Software maintainability: Systematic literature review and current trends," *International Journal of Software Engineering and Knowledge Engineering*, vol. 26, no. 08, pp. 1221–1253, 2016, different range of years included within the literature review, desgin metrics most favored capture characteristics bero deploying software, factors, improvement and prediciton, impact factor on publishers, list of comnferences, maintenance advantages, levels to measure maintainability, process, architecture or code level.

[4] R. Malhotra and K. Lata, "A systematic literature review on empirical studies towards prediction of software maintainability," *Soft Computing*, vol. 24, no. 21, pp. 16 655–16 677, 2020, interesting comaprative RQ8, table 2 empty, range of number of studies within the same range as other papers, too focus on machine leraning techniques, type of metrics, traditional, composite and OO metrics.

[5] F. Tian, T. Wang, P. Liang, C. Wang, A. A. Khan, and M. A. Babar, "The impact of traceability on software maintenance and evolution: A mapping study," *Journal of Software: Evolution and Process*, vol. 33, no. 10, p. e2374, 2021.

[6] M. Riaz, E. Mendes, and E. Tempero, "A systematic review of software maintainability prediction and metrics," in *2009 3rd international symposium on empirical software engineering and measurement*. IEEE, 2009, pp. 367–377.