Introduction

During Week 6, the primary focus was on software estimation and measurement, exploring essential concepts such as measurement theory, module cohesion, coupling, cyclomatic complexity, psychological complexity, and effort estimation. Through the assigned activities, I developed a clearer understanding of how these metrics contribute to evaluating software quality and project planning. The discussion post emphasized the challenges of correlating internal metrics, like cyclomatic complexity, with external attributes, including maintainability (Fenton & Pfleeger, 1997). In the programming assignment I was able to use these concepts and came to understand the cyclomatic complexity of the Quicksort algorithm using McCabe's metric, better equipping me to judge the complexity and maintainability of code-sections (McCabe, 1976).

Difficulties Faced

One of the biggest problems faced was defining the relationship of internal software attributes (e.g., cyclomatic complexity) with external quality factors (e.g., maintainability). The debate made clear that, even though metric values such as cyclomatic complexity can provide leads, they do not feature a straightforward, and universally applicable mapping to actual quality attributes, owing to reasons such as the different development context in which a system evolves, and human-related variables (Kitchenham et al., 2002). In the programming assignment, I initially struggled with accurately constructing the control flow graph and counting the decision points, but repeated practice and reference to McCabe's formula helped clarify these steps. Additionally, interpreting the implications of the calculated complexity on code quality and testing requirements required additional reflection and research.

Activities Performed

This week’s activities included participating in a discussion post and completing a programming assignment. In the discussion post, I critically analyzed the difficulties of linking internal measures like cyclomatic complexity to software maintainability. I engaged with scholarly sources and provided examples that highlighted the complexity of establishing these relationships. For the programming assignment, I analyzed the Quicksort algorithm by constructing its control flow graph and calculating its cyclomatic complexity. I applied both the formula and decision point counting method, concluding that the algorithm’s cyclomatic complexity is 4, which implies a moderate level of complexity requiring careful testing (Kan, 2002).

Conclusion

This week significantly enhanced my understanding of how software estimation and measurement techniques influence quality assurance and project management. Applying cyclomatic complexity analysis has equipped me with a practical tool to assess software maintainability proactively. I realized that while metrics are invaluable, they must be interpreted carefully, considering project-specific contexts and human elements. This reinforces the need for a holistic approach to software quality assessment that blends metrics with expert judgment.

References

Fenton, N. E., & Pfleeger, S. L. (1997). *Software Metrics: A Rigorous and Practical Approach.* PWS Publishing Co. https://www.amazon.com/Software-Metrics-Rigorous-Practical-Approach/dp/0534954251

Kitchenham, B., Pfleeger, S. L., Pickard, L., Jones, P. W., Hoaglin, D. C., Emam, K. E., & Rosenberg, J. (2002). Preliminary guidelines for empirical research in software engineering. *IEEE Transactions on Software Engineering, 28*(8), 721–734. https://libcatalog.usc.edu/discovery/fulldisplay?docid=cdi_webofscience_primary_000177281900001CitationCount&context=PC&vid=01USC_INST:01USC&lang=en&search_scope=MyInst_and_CI&adaptor=Primo%20Central&tab=Everything&query=sub,exact,%20Software%20Research,AND&mode=advanced&offset=0

McCabe, T. J. (1976). A complexity measure. *IEEE Transactions on Software Engineering, SE-2*(4), 308–320. https://ieeexplore.ieee.org/document/1702388

Kan, S. H. (2002). *Metrics and Models in Software Quality Engineering.* Addison-Wesley. https://www.amazon.com/Metrics-Models-Software-Quality-Engineering/dp/0201729156

Wordcount: 407