Initial Post

Initial Post

by Oi Lam Siu - Tuesday, 26 March 2024, 1:06 PM

Reusability means using existing components to create new ones, such as classes, methods, libraries, modules, or documents. To achieve reusability in software code, it should be planned and organized, leading to significant benefits (Padhy et al., 2017).

Padhy et al. (2018) provide a list of factors that they believe impact the reusability of object-oriented software. The following is my prioritization of the list, along with an explanation:

1. Requirement analysis (RA) is a critical process of gathering project requirements. It enables reusability by grouping cohesive requirements, identifying patterns, and creating adaptable software systems. Effective RA understands user needs, aids in early reusability identification, and is crucial for project initiation.

2. Architecture-driven approach (ADP) provides a structured framework for creating reusable software components. It enhances reusability by defining the overall project structure, facilitating seamless integration, modularity, and scalability. Well-designed architecture enables component reuse across projects, promoting adaptability and ease of maintenance.

3. Design patterns (DP) reuse existing designs for similar requirements, providing refined solutions. It ensures consistency, dependability, and expedites development, making it effective for creating new projects.

4. Documentation in the project (DIP) is a valuable asset created during the software development life cycle (SDLC). New documents are designed, often sharing features with old ones, to minimize time and cost.

5. Knowledge requirements (KR) generate valuable assets during the SDLC. Reusing knowledge, including experience, ideas, and reasoning, enhances reusability and informed decision-making across projects.

6. Used in the data project (UD) involves frequent reuse of data which representing recorded experiences from previous projects. Reusing data promotes data-driven decision-making and facilitates informed decision-making based on past experiences.

7. Modules in the program (MIP) refer to dividing a project into separate, executable files that constitute parts of the application. Modularity and task separation enabled by modules facilitate component reuse within a project.

8. Models in the project (MP) can represent the tasks that consist of meaningful code and are also able to represent solutions and insights. This enables developers to utilize existing models and adapt them for new use cases, reducing redundancy and improving overall efficiency.

9. Test cases/test design (TCTD) develop reusable test case suites for software modules, but only applicable across different versions within the same family.

10. Algorithms used in the program (AP) is the reuse of the algorithms in software design only if the same type of problem occurs.

11. Service contracts (SC) facilitate two-way communication between developers and users intending to reuse products. These contracts can be modified based on varying software development requirements.

Reference:

Padhy, N., Satapathy, S., & Singh, R. P. (2017) Utility of an Object Oriented Reusability Metrics and Estimation Complexity. Indian Journal of Science and Technology 10(3): 1-9. https://doi.org/10.17485/ijst/2017/v10i3/107289

Padhy, N., Satapathy, S., & Singh, R.P. (2018) 'State-of-the-Art Object-Oriented Metrics and Its Reusability: A Decade Review', in: Satapathy S., Bhateja V., Das S. (eds) Smart Computing and Informatics. Smart Innovation, Systems and Technologies. 77. Spr

Maximum rating: -