DAY 29-DAILY ASSIGNMENTS-16-12-2024

ANSU MARIUM SHIBU

REPORT:

1.Software Development Life Cycle Early Phases and Quality Metrics: A Systematic Literature Review" by Shokhista Ergasheva and Artem Kruglov:

- **Objective**: To assess and evaluate software quality in the early phases of the software development life cycle (SDLC) through a systematic literature review (SLR).
- **Focus Areas**:
    - Emphasis on Requirements Management and Design phases.
    - Exploration of existing models and approaches for quality assessment.
- **Methodology**:
    - Defined research questions related to software development phases and metrics.
    - Conducted a systematic review of relevant literature using explicit inclusion and exclusion criteria.
    - Collected over 200 publications from various databases (Scopus, Web of Science, ResearchGate).
- **Findings**:
    - Distribution of publications by phase:
        - Requirements phase: 17.2%
        - Requirements and Design phases: 24.1%
        - Design phase: 31.0%
        - Design and Code phases: 3.4%
        - Code and Testing phases: 3.4%
        - All phases: 17.2%
    - Some studies define early phases differently, including User Needs Analysis and Preliminary Design.
- **Quality Metrics**:
    - Identified Structure Quality and Content Quality as key aspects of documentation assessment.
    - Suggested indicators for Structure Quality include:
        - Document structure (chapters, sections, length)
        - Readability and completeness
        - Usability of API and Wiki documentation

    - Content Quality indicators include:
        - Adaptability, installability, and interoperability of documentation.
- **Conclusion**:
    - The review highlights the need for improved quality metrics in the early phases of software development.
    - Emphasizes the importance of structured documentation and quality assessment techniques.
- **References**: The paper cites various studies and methodologies, including works by Aversano et al. and Kitchenham.

2.Analysis of SDLC Models for Embedded Systems" in bullet points:

- **Introduction to SDLC**:
    - Defines Software Development Life Cycle (SDLC) as a framework for software development activities.
    - Discusses various SDLC models: Waterfall, V-model, Iterative, Incremental, Spiral, and Agile methods.
- **Agile Methods Overview**:
    - Agile methods are evolutionary, focusing on collaboration among self-organizing, cross-functional teams.
    - Emphasizes the Agile Manifesto principles: valuing individuals and interactions, customer collaboration, working software, and responding to change.
- **Application of Agile Methods**:
    - Agile methods promote rapid cycles and executable deliverables, which can be challenging in embedded systems due to specific functionality requirements.
    - Highlights the need for granular feature sets in embedded systems to facilitate short delivery periods.
    - Discusses the balance between coding and documentation, noting that embedded systems often require extensive documentation.
- **Continuous Refactoring**:
    - Continuous refactoring is beneficial for optimizing embedded systems, provided the initial architecture is sound.
    - Addresses the long-lived nature of embedded systems and the potential absence of original developers during maintenance.
- **High Value Delivery**:
    - Focuses on delivering features that add significant value to the business while minimizing low-value features.

    - Agile methodologies enhance customer interaction and feedback, leading to better optimization and code refactoring.

- **Conclusions**:
    - Agile methodologies are adaptable and beneficial for embedded system development, but they also present challenges such as potential threats to business continuity due to insufficient design and documentation.
    - Suggests the need for improvements in Agile methods to mitigate these risks.
- **References**:
    - Cites various studies and publications that support the analysis and findings presented in the paper.