INTRODUCTION TO SOFTWARE ENGINEERING

Instructor: Nguyen Van Vu

REVIEW OF THE CLASS

waterfall model: 20-30%
làm theo tuần tự, giảm dư thừa với đk có yc phần mềm sẵn

gặp khó khăn vì cần hoàn thành yc phần mềm

1. Software process models
    a. What is software process? What, who, when, why?
    b. What is software process model?
    c. Models
        i. Waterfall model: traditional approach

agile: phát huy chức năng khó khăn của waterfall model

viết tài liệu nhiều

            1. Sequential
            2. Making sure one stage completed before doing the next
            3. Suitable for projects having stable requirements: it is eliminating the waste of projects.
            4. Stages: requirements definition, design & analysis, implementation, testing, deployment, maintenance & operation.
        ii. RUP: traditional approach
            1. Iterative: doing things repeatedly, incrementally
            2. Phases: Inception, Elaboration, Construction, and Transition
            3. Iterations = cycle
                a. One iteration is like one cycle of waterfall

nhiều

            4. Detailed process to do things
            5. Detailed roles of team members
            6. Many documents to be written
            7. Use-case driven development
        iii. Agile methods
            1. Four value propositions of Agile methods
            2. Scrum

30-40% trong giới cnpm

có nhiều pp agile khác nhau

có agile manyfactal

                a. Sprint, time-boxed iteration
                b. Backlog (product backlog, sprint backlog, impediment backlog)
                c. Roles
                d. Activities
                    i. Daily scrum
                    ii. Sprint review
                    iii. Sprint planning
                    iv. Release planning
            3. XP
                a. Philosophy: take down good practices into extreme
                b. Key practices
                    i. Pair programming
                    ii. Small release
                    iii. Coding standard
                    iv. ….
2. Software project management
    a. Goals
    b. Roles
        i. PM

báo cáo tiến độ, đảm bảo rủi ro

        ii. TA

iii. BA
iv. Tester
v. Maintainer

c. Activities
    i. Planning
        1. Estimation
        2. Scheduling
        3. Task assignment
    ii. Team building
    iii. Human management
    iv. Controlling and monitoring
        1. Report
        2. Problem resolving
    v. Customer collaboration
    vi. Risk management

3. Software requirements & requirements engineering
    a. Software requirements
        i. User requirements (concepts of operation) & system requirements
            1. User requirements
            2. System requirements
                a. Use-case
                b. User story
        ii. Functional requirements & non-functional requirements
        iii. Domain requirements (functional & non-functional)
    b. Requirements engineering
        i. Requirement gathering
            1. Techniques
                a. Interview
                b. Survey
                c. Observe
                d. Record
        ii. Requirement analysis and documentation
            1. Use case model
        iii. Requirement validation
            1. Inspection
            2. TDD – Test-driven development (test generation)
            3. Prototyping (PoC)
        iv. Requirement management
        v. Question: why do we need to do requirements validation? Why requirement inspection/review is effective?

phân biệt, mô tả yc chức năng, phi chức năng, loại yc người dùng

vision doc: những yc cơ bản

4. Software Analysis and Design
    a. Software architecture
        i. Question: why do we need to detail/form an architecture for the software? (Log4j)
        ii. Scalability -> horizontal scale vs. vertical scale
        iii. Architectural critical requirements
    b. Architectural design (architectural design)
        i. Based on requirements to form the architecture
        ii. Question: how does architecture affect the performance of the software?

iii. Loosely coupled vs. tightly coupled?
iv. Fine-grained vs. coarse-grained components?

c. Detailed design (low-level design)
i. High-level design vs. detailed design?
ii. Class diagram
iii. Sequence diagram maybe

5. Software Verification and Validation
a. Verification vs. Validation
i. Why do we need both?
b. Techniques
i. Static inspection
1. Review (artifacts, work products)
2. Analysis (code analysis)
ii. Dynamic
1. Testing
2. Analysis
iii. Test-driven development
iv. Model-driven development
v. …

6. Software testing
a. Levels of testing
i. Unit testing
ii. System testing
1. Integration
2. Release testing
iii. Acceptance testing

vì sao phải thực hiện nhiều loại testing như thế?

b. Types of testing
i. Functional testing
ii. Non-functional
1. Performance
2. Load
3. Usability
4. Security

c. Techniques
i. Regression testing
ii. Ad-hoc testing
iii. Smoke testing
iv. ….

bỏ ii,iii

d. Concepts
i. Test case
ii. Test steps
iii. Test data
iv. Test results
v. Defects
vi. Test coverage
1. Code coverage
2. Function coverage
3. UI coverage
4. Path coverage

e. Techniques

        i. Requirement-based → test cases
        ii. Equivalence partition → test cases
            1. Ex. Generating test cases for testing the registration function with username having at least 8 characters and a special character.
        iii. Path testing
            1. Given a snippet of code, write test cases to cover all paths
        iv. Ad-hoc testing
        v. Smoke testing bỏ iv,v
7. Test automation (not included in the final exam)
    a. Goals
    b. Approaches of test automation
    c. Levels
        i. Unit testing
        ii. Integration testing
        iii. System testing
    d. Tools
        i. Selenium
        ii. Appium
        iii. Katalon
8. User interface design (not included in the final exam)
    a. Principles of UI design

9. Software reuse (not included in the final exam)
    a. Techniques

FINAL EXAM

- Two-sided "cheatsheet": students are allowed to have one two-sided "cheatsheet" to summarize whatever they like.
- Condition
    - No discussion, sharing allowed
    - No phrases, sentences taken from the Internet
- Time: 90 minutes