Course Review & Exam Preparation

COMP4130 Semester 1 2017

Topic Summary

Five Modules

- Introduction to Managing Software Quality and Process (Week 1)
- Software Process and Agile Practices (Week 2)
- Right Product – Client Needs and Software Requirements (Week 3 – Week 5)
- Done Right – Reviews, Testing, and Metrics for Software Improvement (Week 6 – Week 10)
- Managed Right – Project Planning and Monitoring (Week 10 – Week 11)

See Module Summary on:
https://wattlecourses.anu.edu.au/mod/book/view.php?id=1065096

Foundations of Managing Software Quality and Process

- Lecture 2, Module 1 – Topic 1.2
  - FURPS + for defining quality
  - The "iron" triangle of software development
  - Quality in the lifecycle
    - Why process is important
    - Why requirement is important
    - Why monitoring is important
  - Three goals of building high-quality software
    - Right product
    - Done right
    - Managed right

Software Process and Agile Practices

- Lecture 3, Module 2 – Topic 2.1, Topic 2.2
  - What is a process?
    - Phases – Activities – Tasks; Task Dependencies; Tasks – Roles – Work Products – Resources; Processes – Practices – Methodologies
  - Software engineering activities
    - Phases - project management, specification, design & implementation, verification & validation
  - Process models & prototyping strategies
    - Linear, spiral, unified process
    - Illustrative, exploratory, throwaway, incremental, evolutionary
- Lecture 4, Module 2 – Topic 2.3
  - Which process model works or does not work well with agile practices
  - Extreme programming
    - 12 code-centric practices for development
  - Scrum
    - Roles in a Scrum team – product owner, development team, Scrum master
    - Scrum events – Sprint planning, Daily scrum, Sprint review, Sprint retrospective

Right Product – Client Needs and Software Requirements

- Lecture 5, Module 3 – Topic 3.1
  - Requirement activities
    - Eliciting, expressing, analysing, verifying and validating requirements
    - Prioritizing and managing requirements
  - Types of requirements
    - Business requirements, business rules
    - User requirements, functional requirements, non-functional requirements
    - External interface, physical setting, development constraints
- Lecture 6, Module 3 – Topic 3.2
  - Eliciting requirements
    - Define "user" – primary, secondary, tertiary
    - Involving clients
      - You are neither a waiter nor a trial lawyer
      - Good open-ended questions to ask
    - Techniques for requirement elicitation and expressing
      - Use case diagram + use case description
      - Wireframing + storyboard

the type of requirement is important to remember

Right Product – Client Needs and Software Requirements Cont'd

- Lecture 7, Module 3 – Topic 3.3
  - Expressing requirements
    - User story – As a <*role*>, I want to <*function*>, so that <*benefits*>
    - INVEST – independent, negotiable, valuable, estimatable, small, testable
  - Verifying requirements
    - Acceptance criteria and tests
  - Prioritizing and managing user stories
    - Product backlog, story map
- Lecture 8, Module 3 – Topic 3.4
  - Analysing requirements
    - More user story criteria – correct, clear, consistent, feasible, traceable
    - Ambiguous requirements – 11 categories of words
- Lecture 9, Module 3 – Topic 3.5
  - Validating requirements
    - Sprint review meeting
    - User studies

Done Right - Reviews, Testing, and Metrics for Software Improvement

- Lecture 10&11, Module 4 – Topic 4.1
  - Family of review techniques
    - walkthroughs, technical reviews, inspections
  - Requirement technical review & repair
  - Code review techniques
    - Pair programming
    - Refactoring
      - "If it stinks, change it"
      - "Improve existing code in small steps"

Done Right - Reviews, Testing, and Metrics for Software Improvement

- Lecture 12 – Module 4 – Topic 4.2 – Lesson 4.2.1 & 4.2.2
  - What is testing
    - the process of comparing "what is" with "what ought to be"
  - Testing challenge
    - the impossibility of testing everything
  - Test case design
    - input, output (expected versus actual), order of execution
  - Types of testing
    - Black box testing and white box testing
  - Levels of testing
    - Unit, integration, system, and acceptance testing
  - Test driven development

difference between 2 types of testing

Done Right - Reviews, Testing, and Metrics for Software Improvement

- Lecture 13&14, Module 4 – Topic 4.2 – Lesson 4.2.3
  - Boundary value testing
    - Boundary values of independent, bounded physical quantities
  - Equivalence class testing
    - Define an equivalence relation to partition input domain
  - Four types of boundary value testing and equivalence class testing
    - Normal vs. robust
    - Weak versus strong
  - Decision table testing
    - Logical relationships among input variables (conditions), cause-effect relationships between conditions and actions

determine the input output logic of the testing techniques ; whether there is relaitonship between the output and inputs

- Lecture 15&16, Module 4 – Topic 4.2 – Lesson 4.2.4
  - Derive a control flow graph for a program
  - Testing coverage
    - Statement, branch, condition, multiple condition, loop, path
  - Basis path testing
    - Cyclamatic complexity (two equations to compute)
    - McCabe's baseline method
      - Systematically flip the outcomes of decision points

Done Right - Reviews, Testing, and Metrics for Software Improvement

- Lecture 17, Module 4 – Topic 4.3
  - Issues around monitoring the project
    - Finding the time to compile metrics
    - Lack of knowledge and industry standards
  - Use the metrics effectively
    - Goal, Question, Metric framework – use the metrics with a clear goal
    - Desirable properties of metrics – metrics give the results you can trust
  - Defect analysis
    - Ratio of found versus fixed defects
    - Defect density

Managed Right – Project Planning and Monitoring

- Lecture 18, Module 5 – Topic 5.1
  - Story points
    - Avoid mistaking estimates as commitments
  - Velocity
    - Story points completed for the user stories done / the duration of a Sprint
  - Release plan
    - Scheduling user stories into the next few Sprints based on their priority
  - Iteration plan
    - Scheduling developer tasks within a Sprint

the difference between types of burndown chart, what is burnup, what is burn out

- Lecture 19, Module 5 – Topic 5.2
  - Daily scrums
    - Answer three questions about the project's daily execution
  - Release burndown chart
    - Basic release burndown – how much work remaining to complete
    - Top Work Done or Adjustable Floor – the team's velocity, the effect of changing requirement
  - Iteration burndown chart
    - Burn up, burn across

Exam Preparation

Time & Venue

- Time: 2PM, Tuesday 06/06/2017
- Venue: Haydon-Allen G40, Building 22
- Reading time: 15 minutes
- Examination duration: 120 minutes

Hurdle Assessment

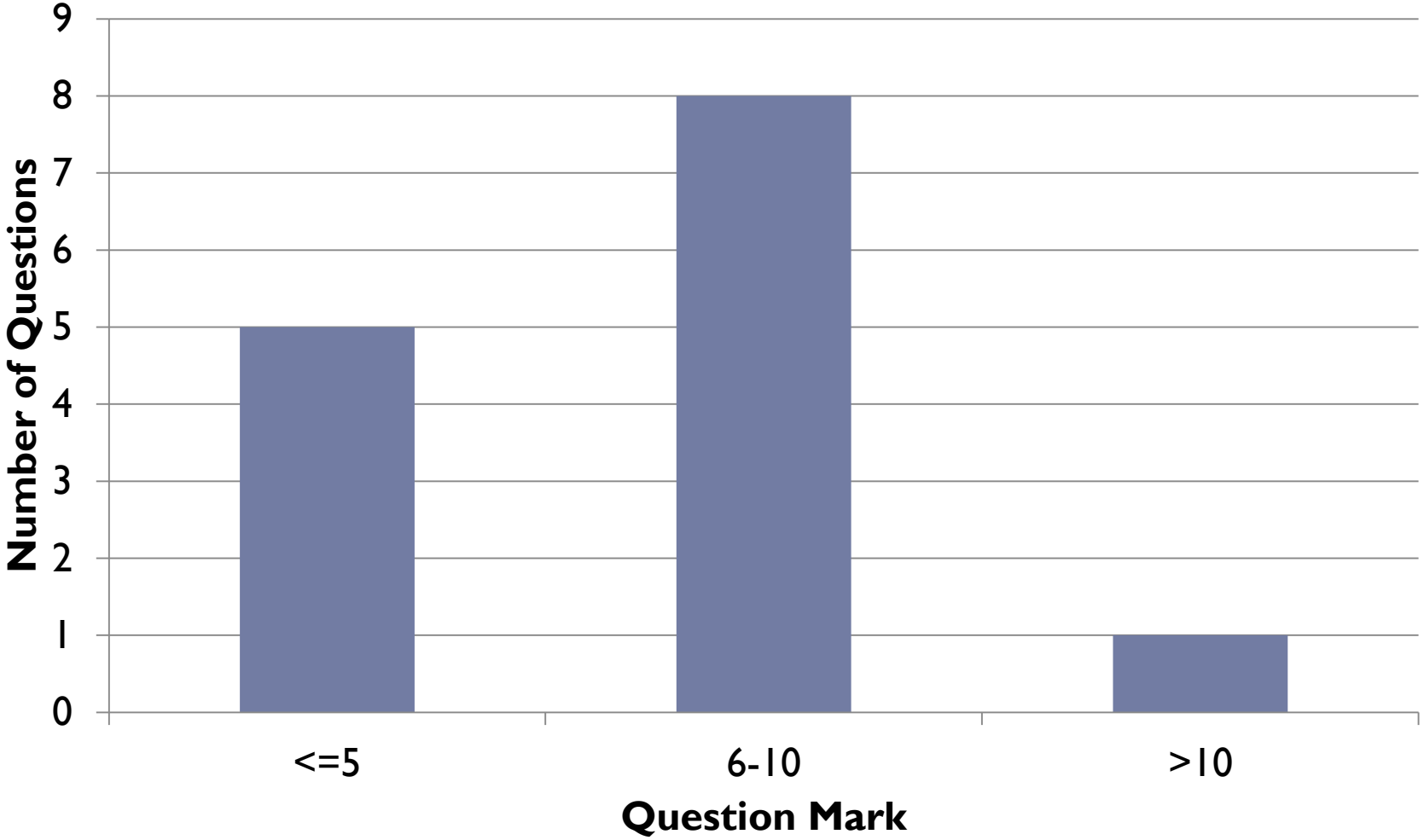
- To pass the course, you must have >= 50% in the final exam, no matter how well you did in assignments and workshops

- I expect to see you ALL there

Format

- Close book, but one A4 page with notes on both sides is allowed
- The paper contains 5 sections and 14 questions
- Answer ALL questions
- Questions carry 2 – 15 marks
- NO MCQ or fill-in blank questions

Question Mark Distribution



Questions

- Nature – know-how, not just know-what
  - You need to know when to use which techniques and how to apply certain relevant techniques to solve the problem
- Style – Similar to workshop exercises
  - You will be given certain scenarios and questions are derived from these scenarios
- Difficulty – Similar to workshop exercises
  - But you have to answer the questions on your own, without the help of your group member

Exam Preparation and Taking Strategies

- Do not blindly review lectures. Think about what we practice more throughout the course to allocate your effort.

- Answer all questions first. Do not spend too much time on one question to get "perfect" answers.