Software Engineering, ESOF 322, Fall 2019
Review for exam 2, Nov. 25

You may bring notes on one side of a sheet of paper into the exam. These notes must have been written by you. They cannot be images from the text or Internet or notes gotten from another student.

"Programming in the Small" Chapter 1 – Writing a Program

- Be able to define what is meant by software engineering
- Know the difference between functional requirements, non-functional requirements, project requirements, design constraints, and quality attributes
- Know the difference between verification and validation
- Know the difference between white box and black box testing

"Building a System" Chapter 2

- Know that developing software is a coordination effort between process, product and people
- Know that programs can be complex in terms of many functions, features, interfaces to external systems, simultaneous users, number of data types and data structures, transfer of control, sharing data
- Know what is meant by loose coupling and high cohesion
- Know what is meant by a software development process
- Know the importance of estimation
- Be able to discuss the article "Velocity in Software Engineering" by Tom Killalea

"Engineering of Software" Chapter 3

- Know what was meant by the "Software crisis" and when and where the term "software engineering" was coined. (First used at 1968 North Atlantic Treaty Organization (NATO) conference held in Germany.)
- Be able to describe what it means to "engineering" software
- Know what is needed for startups and next-generation technology companies

"Software Process Models" Chapter *4*

- Be able to describe the waterfall, incremental, spiral, and RUP process models, and to discuss pros and cons of each
- Know what CMM stands for and its goals

"New and Emerging Process Methodologies" Chapter 5

- Know the four key principles of Agile development
- Be able to describe the XP, Crystal, and Scrum Agile methods
- Given a development environment and project goals, be able to select an appropriate development process model

"Requirements Engineering" Chapter 6

- Know what requirements and requirements engineering are
- Know what the requirement engineering activities: elicitation, analysis, specification, validation and management involve and given a description, be able to place it within these categories.
- Know the difference between functional, non-functional and quality attributes
- Know how UML, DFD and ERDs can be useful in communication requirements
- Know what is meant by requirements traceability

"Design: Architecture and Methodology", Chapter 7

- Know what is meant by a software architecture and that every system has one, whether or not it is documented
- Know that systems have a logical view, process view, subsystem decomposition views and the physical architecture view
- Know that UML diagrams can be split amongst structure, behavior and interaction diagrams and be able to list and describe 2 diagrams in each group
- Know that architectural knowledge has been codified in architectural styles or patterns, architectural tactics and reference architectures, be able to give examples of each, and to categorize architectural knowledge in one of these
- Know what is meant by pipes-and-filters, event-driven, client-server, model-view-controller, layered, database centric, model centric and 3-tiered architectures are and be able to give examples
- Know what REST stands for and its 6 principles
- Know the 4 phases of database design: data modeling, logical database design, physical database design, deployment and maintenance
- Know the advantages and disadvantages of "Not onlySQL" database, and some example databases
- Know what is meant by "sharding"

"Design Characteristics and Metrics", Chapter 8

- Be able to define cohesion and coupling
- Know what is meant by coincidental, logical, temporal, procedural and functional cohesion, be able to give examples of each, and be able to categorize given examples
- Know what is meant by content, common, control, stamp and data coupling, be able to give examples of each, and be able to categorize given examples
- Be able to describe the Law of Demeter at a high level
- Know at least 4 rules of user-interface design and be able to give examples

"Implementation," Chapter 9

- Know that quality attributes such as readability, maintainability, and performance are sometimes in conflict
- Know that quality attributes such as usability, reliability, availability, maintainability, portability and performance and must be considered throughout the life cycle
- Appreciate the need for consistency and adhering to coding standards
- Know to use long names for global identifiers and short names for local identifiers
- Know what is meant by comments that repeat, explain, summarize, describe and give external references to the code, be able to give examples of each, and be able to categorize given examples
- Be able to give guidelines for locating errors
- Know what is meant by refactoring code and give guidelines for code that ought to be refactored
- Know the difference between infrastructure as a service, platform as a service and cloud application services, be able to give examples, to categorize a given service, and to discuss pros and cons of each

Testing and Quality Assurance, Section 10

- Know the best way to obtain quality in software
- Know 4 techniques for detecting errors
- Know the difference between verification and validation, be able to give examples of each, and be able to categorize examples
- Know the difference between an error, defect and failure
- Know the three main levels of testing, be able to give examples of each, and be able to categorize examples
- Know what is meant by acceptance, conformance, configuration, performance, stress and user-interface testing, be able to give examples of each, and be able to categorize examples
- Know the difference between black and white box testing, be able to give examples of each, and be able to categorize examples
- Know what is meant by equivalence-class partitioning and boundary value analysis, and be able to do these for a sample system
- Know what is meant by statement, path and decision coverage
- Know the 5 steps of test driven development and be able to do it
- Know the process of error seeding and what it is accomplishing
- Know what formal methods are and why they are used

Configuration Management, Integration, and Builds, Chapter 11

- Be able to discuss what configuration management is, what might be tracked and the complication of intra-artifact relations
- Know the 3 tiers of configuration management support

Software Support and Maintenance, Chapter 12

- Know the different skills needed by customer service/support people and the technical problem/fix analysts
- Appreciate the need for a well-established change control and be able to describe the process of change control

Software Project Management, Chapter 13

- Know the four phase of software project management (POMA), their relation and activities done in each, be able to give example activities, and be able to categorize activities
- Know the risk management is an integral part of software project management, the three major components and fertile areas to look for risk
- Know what is meant by SMART project goals and be able to write project goals that are SMART
- Know what COCMO stands for, the purpose of this cost model, and be able to describe the process from a high level
- Know that there are several measurements of size such as KLOC and function points
- Know what is meant by function points, their advantages, that there is an ISO standard and user group for function points, along with considerable historic data
- Know the components of function points, how the unadjusted and adjusted function point count is determined, and be able to estimate function points at a gross level
- Know the relation between function points and story points

Ethics

- Know that professions have a duty to public welfare and the environment
- Know the characteristics of a profession and be able to list the 4 principles of the Software Engineering Code of Ethics and Professional Practice.
- Know the different between ethics and morals
- Know what is meant by subjective relativism, cultural relativism, divine command theory, Kantianism, utilitarianism and social contract theory
- Given a case study be able to use the theories of Kantianism, utilitarianism and social contract theory to analyze the impact of computing and engineering solutions on individuals, organizations and society
- Know and be able to discuss Rawls Theory of Justice