CSC 131 - Final Exam Study Guide
(12/7/2015)

Software Engineering - fall semester 2015

Preparation checklist:

- Review the material we covered in class, any notes, class quizzes (2 of them), etc.
- Review the material linked from our class web site (SacCT).
- Review your class project including its deliverables: SRS, SDD, and final reports.
- Study with one or more classmates/project teammates, quiz each other on the list of topics.
- Practice your sample final examination posted in SacCT (a MUST).

Format of exam:

You will be given four questions. One question is true/false and multiple choice. The remained three questions are problem-based questions where you will be addressing using the information you learned from your below checklist.

• Closed book, closed Internet.

• Single session (< 2 hours) exam.

• You may use a 2-sided sheet of notes.

• I am looking for precision, specificity, and evidence of what you have learned in this course.

• Please provide neat handwriting.

Grading:

Your answers will be graded on your demonstration of how well you understand the software engineering topics we have covered in class, and also how well you have been able to understand and incorporate topics from among the list on the following page:

V

Important Topics to review for the final exam (No particular ordering)

Software engineer	**Types of software components for reuse**
Software engineering	**Modeling Diagrams:** **Activity Diagram** **Sequence Diagram** **State Diagram** **Data Flow Diagram**
Software Engineering Proposal and its key points	
Software process model	
Four primary software engineering activities: Specification , Development, Validation, Evolution	**Structure Analysis**
	Object Oriented Analysis
Waterfall process model	**Requirements Process: Elicitation, Analysis, Specification, Validation**
Spiral process model	**Scenario vs Use Case**
Incremental model	**User Story**
Spiral model	**Refactoring**
Agile Manifesto	**Requirement Specification (natural language vs. structured specification)**
Agile software development	
Extreme Programming (activities)	**Requirement validation**
Pair Programming	**Business Requirements - User Requirements – System Requirements**
Requirement Engineering	**Characteristics of excellent requirements**
Software Requirement Specification (SRS)	**Advantages of agile processes. Risks/disadvantages of agile processes**
Functional requirements	**Traceability matrix**
Non-Functional requirements	**Verification and Validation**
Use case diagram	

Architectural .vs. Detailed design **Architectural patterns: Pipes-and-Filters, Client-Server, ModelView-Controller (MVC), Layered, Database Centric, Three tier architecture** **Service Oriented Architecture:** **Web Services** **Software Design and Implementation** **Design Processes** **Functional Decomposition** **Relational Database Design** **Object-oriented Design and UML** **UML Class diagrams** **Software metrics: McCabe's Cyclomatic Complexity, coupling, and Cohesion** **Security and software Engineering** **Describe how the Secure Development Lifecycle (SDL) works.** **Vulnerability Testing scenarios: Buffer Overflow, Script Injection, Numeric Overflow, Programmer Backdoors, SQL Injection, etc.**	**Static verification vs. Dynamic verification** **Failure vs. Fault** **Test cases, test stub, test driver** **Software testing? Why testing? How we test?** **Who do the testing? What is tested? how the test cases are design?** **Differences between white box and black box testing?** **Different level of testing: unit testing, integration testing, system testing, and acceptance testing.** **Unit Testing** **Partition testing (Equivalence Class Partitioning)** **Boundary value analysis** **Path testing (Path Analysis)** **Guideline-based testing** **Software Evolution**