Department of Computer Science and Engineering

College of Engineering, University of Nevada, Reno

CS 791 Graduate Topics in Computer Science [Software Engineering]

Set of questions for the midterm test of November 25th, 2019

Version 2 – November 10, 2019 [complete]

A. Study Material

- Chapter 2, Software Processes [Sommerville 2015] – see Lecture 04
- Chapter 3, Agile Software Development [Sommerville 2015] – Lectures 06 and 07
- Chapter 4, Requirements Engineering [Sommerville 2015] – Lecture 07
- Chapter 5, System Modeling [Sommerville 2015] – Lectures 08 and 09
- Paper R1 by Sarah Beecham et al, 2014 [presented by instructor]– Lecture 09
- Paper R2 by Mary Shaw, 2002 [presented by instructor] – Lecture 16
- Paper R3 by Theisen et al, 2017 [presented by instructor] – Lecture 17
- Paper R4 by Otero and Peter, 2015 [presented by instructor] – Lecture 23
- Paper SR1 by Ivanova et al, 2019 [presented by Vineeth] – Lecture 12
- Paper SR2 by Koschel et al, 2017 [presented by Alex] – Lecture 12
- Paper SR3 by Karac and Turhan, 2018 [presented by James] – Lecture 12
- Paper SR4 by Kersten, 2018 [presented by Aritra] – Lecture 13

B. Questions

From Chapter 2, Software Processes [Sommerville 2015]

1. Explain what a *software process* is and briefly describe the typical phases of a software process (specification, design, etc.).
2. Explain the differences between *plan-driven* and *agile* software processes.
3. Describe the *waterfall* software engineering process model. Also, indicate its advantages, disadvantages, and applicability.
4. Describe the *incremental development* software engineering process model. Also, indicate its advantages, disadvantages, and applicability.

5. Describe the *integration and configuration* (*reuse-oriented)* software engineering process model. Also, indicate its advantages, disadvantages, and applicability.

6. Describe the *incremental delivery* software engineering process model. Also, indicate its advantages, disadvantages, and applicability.

7. Explain what is meant by *software specification* and describe the activities involved in the requirements engineering process.

8. Explain what is meant by *software design* and describe the activities involved in the software design.

9. Describe the *testing phases* in a plan-driven software process.

10. Explain what is meant by *software evolution* and describe the activities involved in the software/system evolution.

11. Explain what is meant by *software prototyping* and indicate its benefits. Also, explain why *throw-away prototypes* should be discarded.

From Chapter 3, Agile Software Development [Sommerville 2015]

12. Describe the *philosophy* behind agile methods as reflected in the *agile manifesto*.

13. Describe the 5 main *principles of agile methods.*

14. Give at least 5 examples of *questions* one should ask when deciding on the *balance between a plan-based and an agile approach*.

15. Describe 5 *principles or practices specific to extreme programming* (out of 10 indicated in the book).

16. What are several *problems/issues* pertaining to applying *agile methods*? (Indicate at least 5.)

17. What is meant by *refactoring*? Give at least 3 concrete examples of refactoring.

18. Describe the *Scrum* approach. Indicate what you think are its main advantages and disadvantages.

From Chapter 4, Requirements Engineering [Sommerville 2015]

19. Explain what is meant by a *user requirement* and by a *system requirement*. Give a concrete example for each.

20. Explain what is meant by a *functional requirement* and by a *non-functional requirement*. Consider your CS 791 SE project and give 3 examples of each. Briefly indicate what your project is about.

21. Give at least 6 examples of *different metrics* that could be used for specifying *non-functional requirements* (e.g., Gbytes for memory size).

22. Describe and compare 3 *notations* for writing a software/system requirements specification document.

23. Explain what is meant by a *software requirements document* (SRS) and outline its structure.

From Chapter 5, System Modeling [Sommerville 2015]

24. Briefly indicate and describe 4 types of *UML diagrams.*

25. Explain what a *context model* is and give an example of such model.

26. Explain what a *use case* is and give an example of *a use case diagram* with at least 6 non-trivial use cases.

27. Explain what a *sequence diagram* is and give an example of such diagram that involves at least an actor and two objects.

28. Briefly describe the three main types of *relationships between classes*. Give an example of a *class diagram* with at least 7 classes in which all three types of relationships are represented.

29. Briefly explain what a *state chart (state diagram)* is, indicate its main notation elements, and give an example of a state chart with at least 6 states.

30. Explain what *model-driven engineering* is and discuss its benefits and limitations.

From paper R1 by Sarah Beecham *et al* [2014] – presented by instructor, Lecture 09

31. Indicate at least 4 *challenges* in Global Software Engineering (GSE) as highlighted by Beecham et al [2014]. What do you think are the *reasons* practitioners do not look at related GSE research papers for finding solutions/guidance to address these challenges?

32. Describe the *Practice-Research paradox* (or divide) and discuss the reasons for its existence (see paper by Beecham et al [2014]).

33. What do Beecham *et al* [2014] suggests to do in order *to make research accessible* and thus address the practice-research divide?

From paper R2 by Mary Shaw [2002] – presented by instructor, Lecture 16

34. Describe some of the *critiques* of experimental Software Engineering (SE) mentioned in the [Shaw 2002] paper.

35. Describe the 3 *main types of research contributions* found by Newman [1994] as prevalent in 5 engineering fields – and mentioned in the [Shaw 2002] paper.

36. Table 1 of the [Shaw, 2002] paper presents several types of *research questions in software engineering.* Indicate 3 such types of research questions and give 2 examples of questions for each type.

From paper R3 by Theisen *et al* [2017] – presented by instructor, Lecture 17

37. Describe the *main goals* of the [Theisen et al, 2017] paper and discuss their *main findings*.

From paper R4 by Otero and Peter [2015] – presented by instructor, Lecture 23

38. Explain what is *big data* and what is *big data software* as defined by Otero and Peter [2015].

39. Describe the *single hop* and the *multi-hop intelligence production processes* presented by Otero and Peter [2015].

40. Describe the *requirements problem* identified by Otero and Peter [2015] in relation with big data analytics software engineering.

41. Describe the *construction problem* identified by Otero and Peter [2015] in relation with big data analytics software engineering.

From paper SR1 by Ivanova *et al* [2019] – presented by Vineeth – Lecture 12

42. Consider the paper by Ivanova et al [2019]. Describe the term *gamification*, explain why it is useful, and describe 2 examples of gamification for education.

43. Consider the paper by Ivanova et al [2019]. Briefly describe the *Scrum* and *Kanban* methodologies and present 5 differences between them.

44. Describe how the *Alphabet Brainstorming* game was used in the [Ivanova et al, 2019] paper and summarize the results of its use.

From paper SR2 by Koschel *et al* [2017] – presented by Alex – Lecture 12

45. Consider the paper by Koschel et al [2017]. Describe what is meant by *monolithic architecture* and discuss its advantages and disadvantages.

46. Consider the paper by Koschel et al [2017]. Describe what is meant by *microservice architecture*, summarize its advantages, and briefly present at least 3 of its major characteristics [for this question, see also related PPTX slides by Alex Yovev].

From paper SR3 by Karac and Turhan [2018] – presented by James – Lecture 12

47. Consider the paper by Karac and Turhan [2018]. Explain what is meant by *Test Driven Development (TDD)* and describe what TTD promised when it initially came out.

48. Consider the paper by Karac and Turhan [2018] on *Test Driven Development* (TDD), which cites [Causevic et al, 2011]. Indicate 5 factors that limit TDD's use in the industry.

49. Describe what Karac and Turhan [2018] recommends in terms of "how do you decide whether and when to use Test Driven Development (TDD)?"

From paper SR4 by Kersten [2018] – presented by Aritra – Lecture 13

50. Consider the paper by Kersten [2018]. Describe the reasons (catalysts) for the existing massive *DevOps tool diversification.*

51. Consider the paper by Kersten [2018]. Explain what the author means by *fundamental diversity* and by *accidental diversity* in relation with software tools.

52. Describe 3 varieties of *fundamental diversity* (pertaining to *software tools*) as presented in [Kersten, 2018].