CS 422 Software Engineering Principles **Study Questions (Ch10-17)**

Reminder: *No key will be provided prior to the exam.*

1. (In general you should know) What do we mean by software engineering?

Lets use the current definition as promoted by the IEEE Computer Society: It is (1) the application of a systematic, disciplined, quantifiable approach to the development, operation and maintenance of software, i.e. the application of engineering to software. (2) the study of approaches as in (1).

2. What do we mean by "fundamental principle"?

A fundamental principle is less specific and more enduring than methodologies and techniques. It should be phrased to withstand the test of time. It should not contradict a more general engineering principle and should have some correspondence with "best practice." It should be precise enough to be capable of support and contradiction and should not conceal a tradeoff. It should also relate to one or more computer science or engineering concepts.

3. How were the candidate principles identified?

A first workshop was held at the Forum on Software Engineering Standards Issues of 1996 (SES'96) to establish what a fundamental principle is and which criteria it should conform to. The Software Engineering Management Research Laboratory of the University of Quebec in Montreal was then mandated by the IEEE Software Engineering Standards Committee, to do some research on this topic. They directed the following studies. A Delphi study was conducted among 14 renowned personalities of the Software Engineering community, to identify the fundamental principles of software engineering. A second workshop was held at the International Symposium on Software Engineering Standards of 1997 (ISESS'97) to eliminate or reformulate some of the principles and the criteria. Finally, a second Delphi study was conducted in 1998 among 31 IEEE Software Engineering officials in order to improve the principles. From these studies, a list of fifteen candidate fundamental principles of software engineering has been compiled. For more information, please see http://www.lrgl.uqam.ca/fpse.

---

4. True/False In Model-based specification methods the state of the system is not hidden unlike algebraic specification methods (Chap10/11).

5. Name the two most widely used model-based specification languages (Chap11).

6. Z schema include three parts (Chap11):

a) A _____ **name** identifying the schema.

b) A _____ **signature** introducing entities and their types.

c) A _____ **predicate** part defining invariants over these entities.

7. Z schema can be included in other schema's and may act as _____ **type** definitions (Chap11).

8. Specification using functions does not allow ordering to be specified and sequences are used for specifying ordered collections. A sequence is a mapping from _____ **consecutive** integers to associated values (Chap11).

9. Operations may be specified in Z by defining their effect on the system _____ **state** (Chap11).

10. Specifications can be used as building blocks. Specifications should be structured into small, comprehensible which are put together to form more _____**complex** specifications (Chap11).

11. A Z specification is presented as a number of _____ **schemas**, which may be combined to make new _____ **schemas** (Chap11).

12. Describe what the following schema is saying (read the specification to yourself and write down the words that are expressed in the predicate) (Chap11).

```
┌─Storage Tank──────────────────────────────────────
│ contents: ℕ
│ capacty: ℕ
│ reading: ℕ
│ danger_level: ℕ
│ light: {off, on}
├──────────────────────────────────────────────────
│ contents ≤ capacity
│ light = on <=> reading ≤ danger_level
│ reading = contents
│ capacity = 5000
│ danger_level = 50
```

---

13. Define the six Z conventions (Chap11):

N versus N' means:

A variable name decorated with ! means (Chap11):

A variable name decorated with a ? means (Chap11):

If a schema name is prefixed with the Greek character Xi (Ξ) this means (Chap11):

If a schema name is prefixed with the Greek character delta (Δ) this means (Chap11):

14. Give a diagram showing the range and domain of the function SmallSquare(one → 1, three → 9, seven → 49) and give your definition of the *range* and *domain* of a function (Chap11).

15. Formal specification forces an analysis of the system requirements at an _____ _____ **early stage** (Chap11).

16. Given the two different specifications shown below answer the following questions: (a) name the two types (left one) _____ **Algebraic-based** (right one)_____ **Model-based (Zed => half credit)**. Name the three parts of the specification on the left and the four parts of the specification on the right (Chap11).

```
Storage Tank (A)
contents: ℕ
capacty: ℕ
reading: ℕ                    (B)
danger_level: ℕ
light: {off, on}
---------------------------------------
contents ≤ capacity
light = on <=> reading ≤ danger_level
reading = contents
capacity = 5000               (C)
danger_level = 50
```

A._____

B._____

C._____

D._____

E._____

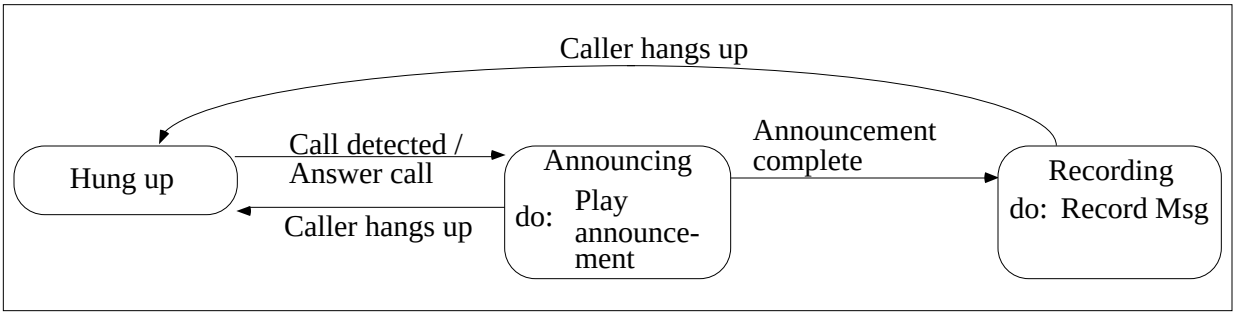
F._____

G._____.

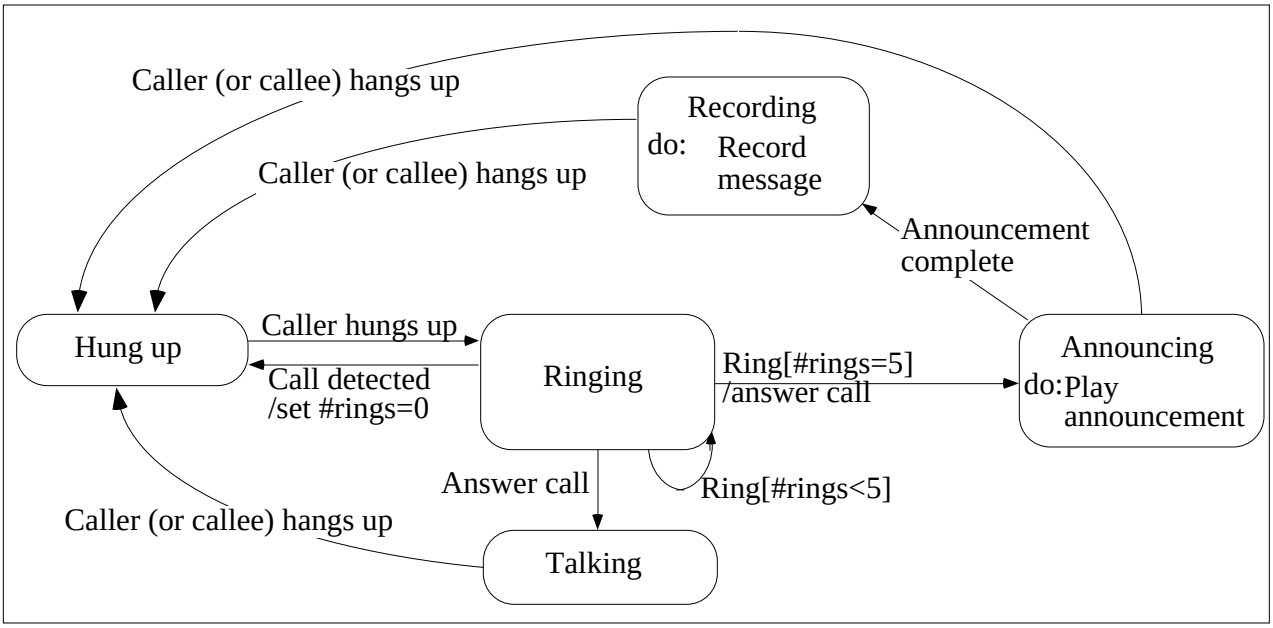
```
ARRAY (Elem: [Undefined -> Elem])
sort   Array                                  (D)
imports  INTEGER
---------------------------------------------------
Arrays are collections of elements of generic type
Elem. They have a lower and upper bound (discovered
by the operations First and Last). Individual elements
are accessed via their numeric index.
Create takes the array bounds as parameters and
creates the array, initializing its values to Undefined.
Assign creates a new array which is the same as its
input with the specified elementassigned the given   (E)
value.  Eval reveals the value of a specified element.
If an attempt is made to access a value  outside the
bounds of the array, the value is undefined.
---------------------------------------------------
Create  (Integer, Integer) → Array
Assign  (Array, Integer, Elem) → Array
First  (Array) →  Integer                         (F)
Last  (Array) →  Integer
Eval  (Array, Integer) →  Elem
---------------------------------------------------
First (Create (x, y)) = x
First (Assign (a, n, v)) = First (a)
Last (Create (x, y)) = y
Last (Assign (a, n, v)) = Last (a)                (G)
Eval (Create (x, y), n) = Undefined
Eval (Assign (a, n, v), m) =
  if m < First(a) or m > Last(a) then
    Undefined  else
    if m = n then v else Eval (a, m)
```

a) Schema name, b) Schema signature, c) Schema predicate, d) Introduction naming the sort (or type) of the entity being specified, (e) Informal description, (f) Signature defines the syntax and operations, and (g) Axioms that defines the semantics.

17. Consider the Telephone Answering Machine (TAM) shown in the diagram below (Chap11).



The TAM above is activated on the first ring. Revise the state diagram so that the TAM answers after five rings. However, if the telephone is answered before five rings, the machine should do nothing (except to allow Talking). Be careful to distinguish between calls where the callee answers before the fifth ring and a call that rings five times.



The process of writing a formal specification based on design artifacts may be called _____ _____ [1pt]. **Reverse Engineering**

Choose one or both of the two types of formal specification methods discussed in Chapter 10 (Algebraic) and Chapter 11 (Zed) and to specify the TAM.

Note, there is definitely one of the two methods more suited for this particular problem (at lease from the stand point of how it is described. Think of what is being given to you. Then ask yourself, which method is based on specifying what is fundamentally being provided. First, give your choice of which method most easily maps to the way this problem is stated _____ **(model-based or Zed)** method (if you just skim the two chapters it should become obvious).

TAM

Machine: {wait, play, record}

Phone : {on_hook, off_hook}

Caller : {on_hook, off_hook}

Ring_det? : {true, false} --Boolean

Ann_done? : {true, false}

Ring_cnt : N

Ring_cnt' = 0 <=> Caller ==on_hook

Ring_cnt' = Ring_cnt + 1 <=> Ring_det == true ^ Phone ==on_hook

Machine = play <=> Ring_cnt = 5 ^ Caller==off_hook^ Phone==on_hook

Machine = record <=> Ann_done? ==true ^Caller==off_hook

Machine = wait <=> (Caller==on_hook) or (Caller==off_hook ^ Ring_cnt < 5) or

(Caller==off_hook ^ Phone==off_hook)

Explain (in generic terms) what is being said in the first half of this schema and do the same for the second half.

Algebraic Method (give a simple try and show all four parts):

18. Compare and contrast the four parts in a Zed Schema and the three parts an Algebraic Specification. Describe these elements (i.e., parts) in terms of: syntax, semantics, predicates, (properties), names, types, sort, imports, set of objects, formal and informal, easier to understand. signatures, operations, axioms, and characterize the behavior (Chap10 and 11).

    Guaranteed test question (no key will be provided prior to exam)

19. What is Software Cohesion and Software Coupling?

    Measure of how closely the parts of a component relate to each other (see slide page 44 for the types of cohesion and there definitions).

    Measure of strength of component interactions.

20. How do the terms cohesion and coupling figure in to (i.e., help or hinder) making software systems (a) more adaptable, (b) more extensible *and* (c) more maintainable?

Chap 12 (including the handouts given (which covered some parts of Brooks), the quiz 2, and Pg. 223, key pts

Guaranteed test question (no key will be provided prior to exam).

21. (Chap 12) Name the 3 main types of notation used in design descriptions:

    Graphical, PDL (i.e., pseudo-code) and informal text Pg. 214.

22. (Chap 12-15) Discuss the differences between object oriented and function oriented design.
    Chap 12

23. (Chap 13) Explain why it may be necessary to design the system architecture before the specifications are written. **Exercise 13.1, Pg. 244**

24. (Chap 14) Explain why adopting a design approach based on information hiding should lead to a design which may be readily modified? **Exercise 14.1, Pg. 272.**

25. (Chap 15) Data flow diagrams are a means of documenting end-to-end flow through a system. They do not include what? **Pg. 294, key pts.**

26. (Project) The design methodology we used in the project is known as SSA/SD which stands for _____ **Structured Systems Analysis and Structured Design**. Name the five parts that were delivered in the design notebook (not including the Requirements Traceability Matrix [RTM]). There is some flexibility in terms of the ordering but list them in the order that would be most suitable and appropriate.

(a) **Context diagram and DFDs must be first**

(b) **Transaction analysis**

(c) **Structure chart (transformation analysis)**

(d)

(e)

P-Specs and Data dictionaries can be sprinkled anywhere in there except first.

27. (Handout on design) In expressing your ideas about design, there are three major principles. The roles of representation in capturing, explaining and checking design information include:

(a) Capturing the designer's ideas for a solution

(b) Explaining the designer's ideas to others (customers, implementers and managers).

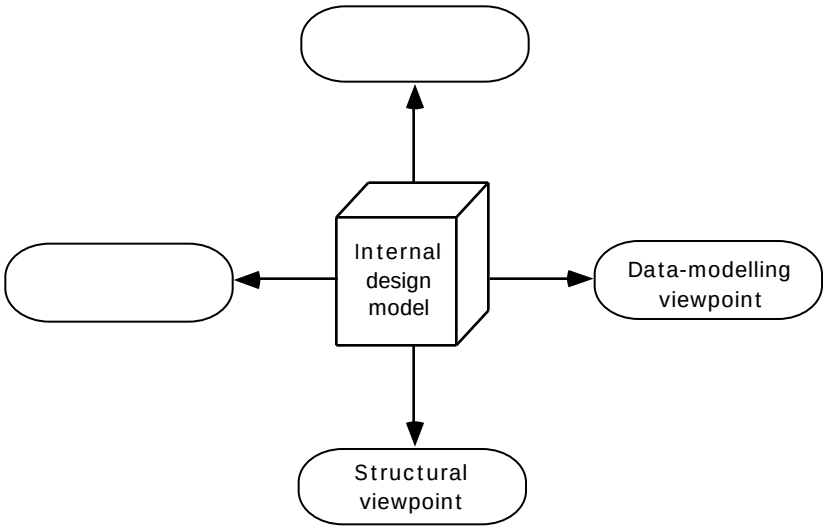
(c) Checking for consistency and completeness in a solution.

Design viewpoints for software give a means for capturing a particular set of design attributes. The type of attribute is projected through the use of a representation form of notation.

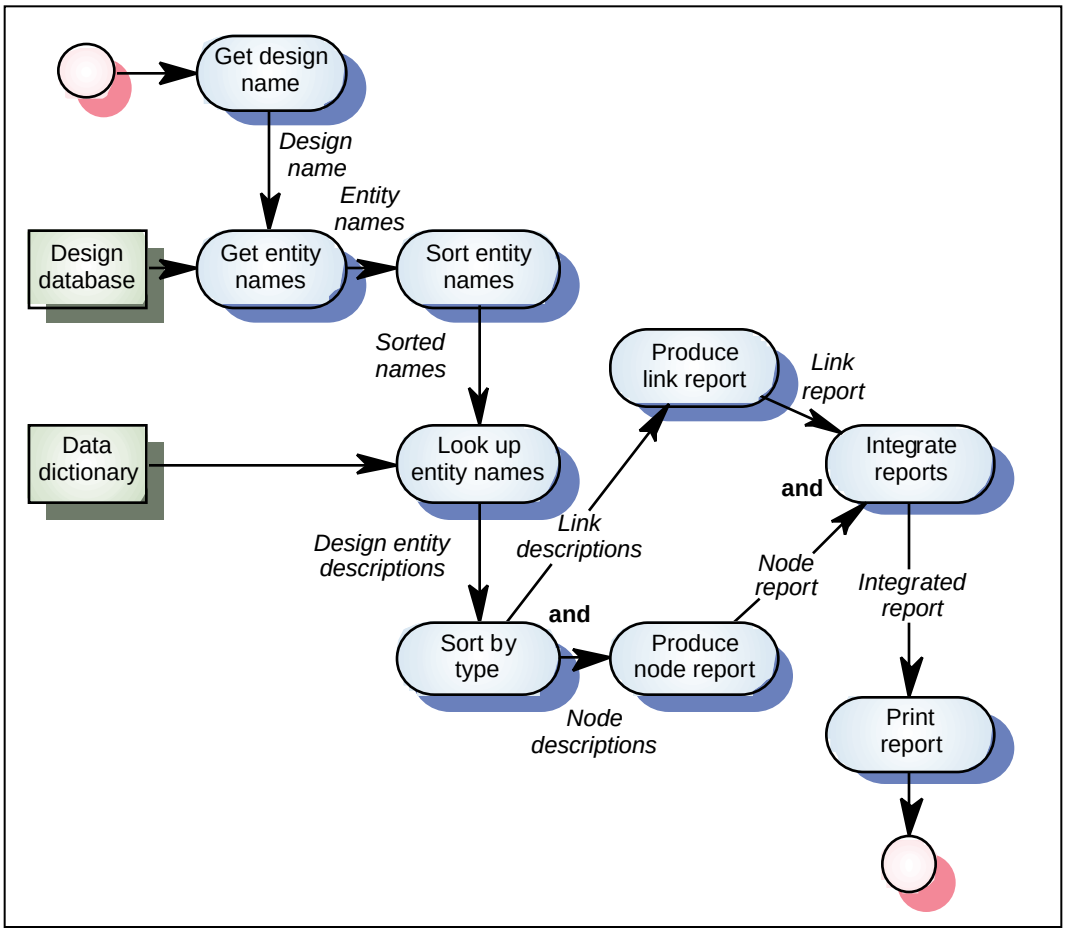
Forms of notation: There are four principal classes of design viewpoints (that use various notations dependent on their inventor). Write in the diagram the missing viewpoints

Guaranteed test question (no key will be provided prior to exam).

Answer should show four: structural, behavioral, functional and data-modeling forms



28. Structure charts are a way of representing the hierarchical organization of a system. Give a structure chart for Figure 15.3. **Answer is Figure 15.4-6.**



29. (Chap 14) In structural decomposition there are several rules of thumb which may be applied to help the designers (1) Many systems can be considered to have three stages: input some data (perhaps with validation and checking), process the data and then output the data (perhaps in report form); (2) If data validation is required, such functions used to implement the checking should be subordinate to the input function and output formatting follows in this same vein; (3) The role of functions near the top of the structural hierarchy may be to control and coordinate a set of lower-level functions; (4) The objective of the design is to have loosely coupled, highly cohesive components. Functions therefore should do one and only one thing; (5) Each node in the structure chart should have between two and seven subordinates. If there is only one it implies the unit represented by that node may have a low degree of cohesion. Too many subordinates indicates that the design may have been developed to too low a level at this time. Three process steps, which follow these guidelines, can be identified for the transformation process from data-flow diagrams to structure charts. Name the three steps:

    (a)_____**identify system processing transformations**

    (b)_____**identify input transformations**

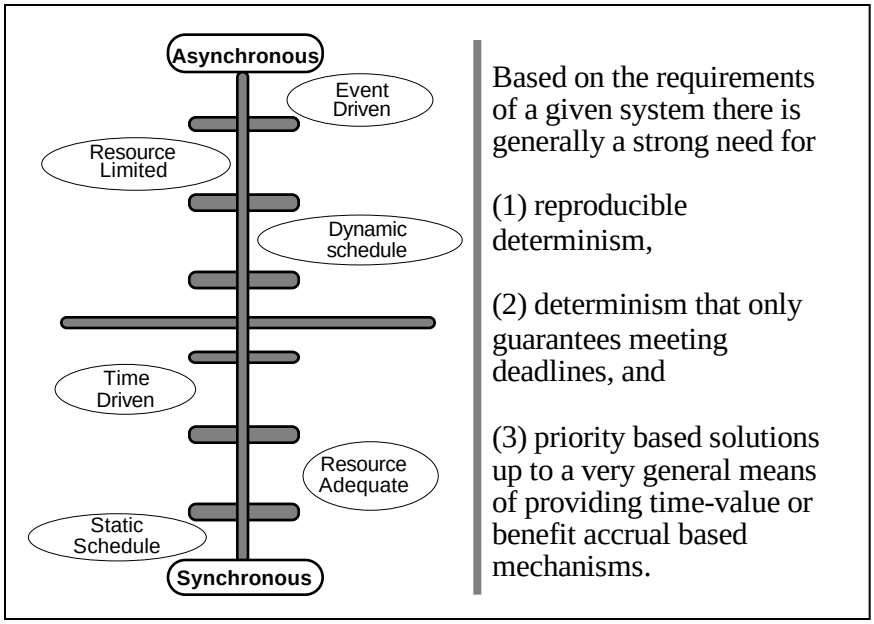
    (c)_____**identify output transformations**

30. (Chap 16) A real-time system is a software system that must _____ respond to events in real-time. The correctness does not only depend on the accuracy of the results the system produces, but also on the _____ **time** when these results are produced.

31. (Chap 16) There are two different types of real-time systems _____ _____ systems and _____ _____ systems. Explain the difference between the two.

    (hard/soft real-time – distinguished by the property of whether they can tolerate lateness in meeting deadlines).

32. Show where these two different types of real time systems fit within the gradient shown in the figure below (Chap 16). Guaranteed test question (no key will be provided prior to exam).



Hard real time systems tend to be designed using a more deterministic approach which puts them near to the bottom of the gradient using static scheduling approaches while soft real time systems can afford to take a more non-deterministic approach which places them nearer the top of the gradient in an asynchronous event driven" environment

33. (Chap 17) To study for this chapter carefully review the key points (slides 57-58). Pay particular attention to the following slides (suggest that you write down some thoughts [and specific details] about each of those slides here below):

Slide 9

Slide 11-12

Slide 21-23

Slide 31

Slide 33

Slide 37-38

Slide 46

Slide 48-49

Slide 56

Slide 57-58