

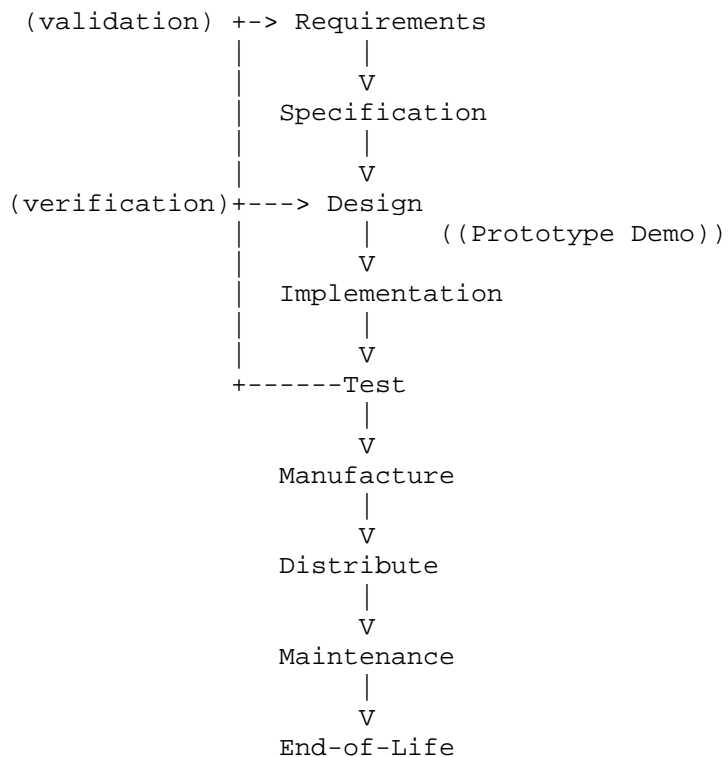
>>> SOLUTIONS <<<

Welcome to your 10% mini-exam for *Senior Project*. The exam is open book and notes, also open Internet (you may google, etc.). **However, you may not receive or give help from or to another person by discussing the exam in any way.** If you copy the answer directly out of the textbook (or some other source, such as a website), be sure to properly “quote” the answer (otherwise it is assumed that any answer is given in your own words). There are six questions (each worth 16 points – there are 4 points overall for “free”) and one extra credit question worth 10 points. The exam is targeted for 75 minutes – about 12 minutes per question. Please fill-in this cover sheet and submit it on top of your answer sheets. Please use a separate sheet of paper for each question. It is OK (actually, I would prefer it ☺) to type the answers (i.e., to use a word processor on a computer). **Please read the entire exam before starting to answer any questions – there are two pages.**

Problem #1

Answer the following questions about the product development process...

a) Sketch the product development process as we have discussed and followed it in this class.



b) What is incremental development?

Incremental development - or progressive refinement - is one where every day you are measurably and visibility closer to the goal of shipping your product by building your product one step at a time. Typically, you will have many sub-release phases or versions (say, version 0.1, 0.2, and so on) where each phase implements more function (i.e., covers more requirement or specification items). The idea of inchstones fits well with incremental development. The idea of iterating between the spec/design and implementation fits well with incremental development.

c) What is “big bang” development?

Big bang development is where the design is 100% perfect before implementation (e.g., coding) begins. Implementation is of the entire design/specification and the first running software and/or hardware contains functionality for all requirement and

specification items. This is in contrast to incremental development where there are lots of partial functionality sub-releases and iteration between phases. An argument for big bang development is that it has less overhead and wasted work than incremental development. There is, however, substantial risk that nothing is completed by the scheduled ship date. With incremental development, something (even if not 100% covering all design/specification items) is ready to ship on the end date.

d) What is (are) the purpose(s) of a prototype demo?

A prototype demo serves several purposes. It lets the customer see what he or she will be getting and to request changes in a way (and time) when they can still be accommodated. It gives the customer (and the developer!) confidence that the final deliverable can be build and shipped. It gives the developer experience (and thus training) in all aspects of the project and creates the Brooks' "throw the first one away" program (See Chapter 11 of Brooks).

Problem #2

What are requirements (and the requirements document) all about? What are the key "ingredients" in a requirements document?

The requirements define the problem. The requirements answer the questions: What is the problem and do you really understand the problem? Who is the user? What is important (e.g., cost, performance, delivery date)? Important ingredients include: measurable and numbered items. Each item describes a unique (non-overlapping) capability, function, or parameter of the problem.

Problem #3

What are the specifications (and specifications document) all about? What are the key "ingredients" in a specification document?

The specification defines an implementation. That is, the specification answers the question: How will you solve the problem? In a specification each measurable aspect of a system is covered. The input, output, and transformation (and any data) must be described in measurable terms. The specification must be traceable to the requirements via a traceability matrix to ensure that no requirement item has been missed and/or no extra unneeded work (extra function) is being done.

Problem #4

What is the test plan (and test plan document) all about? What are the key "ingredients" in a test plan document?

The test plan answers the questions "how do you know we built it right?" (verification) and "how do we know we built the right thing?" (validation). A test plan contains descriptions of test cases. A numbered/named test case describes the system configuration, the input, and the exact expected output. A traceability matrix (or other mechanism) traces each test case to a requirement and/or specification item.

Problem #5

Give at least two examples of partionable tasks and discuss how overhead can result in a less than "an even trade of men for months."

A partionable task can be completed sooner when more workers (people) are added to the task. Two examples are picking strawberries and writing a dictionary. Clearly, adding more people to a field of strawberries or to the researching and writing of word definitions will result in the end task being completed sooner. However, if the number of workers (in either case) is large, a single person (or a large percentage of the effort of a single person) must be dedicated to coordinating the task. The coordination and communications overhead is in assigning non-overlapping parts of the field to pickers and assigning non-overlapping words to researchers/writers.

Problem #6

Is code reuse a silver bullet? Explain and discuss carefully.

This is a tough question. Obviously, the best way to be productive is to not write software but just use an existing solution. However, what is the cost of the existing solution if it is truly reusable? Brooks debates this with himself on page 222 (of MMM (1995)) and does not conclude that a 10x improvement is achievable. It is noted that to make a component reusable it may take 2 to 3 times the effort than if producing it for one-shot use. Thus, a component must be used many times to gain a productivity improvement. Even to reuse a component requires time and effort for a programmer (think about learning .NET or other re-use libraries!!!).

Extra Credit

So... suppose you have a team member who is not carrying their weight, what should you do? Note: A well thought-out answer is needed, not just "in my opinion" and "I think". A good answer will cite authoritative sources. I don't know the right answer, but I know many wrong answers!

I am looking for evidence of clear thought and some research in the literature.

I hereby state that I did not discuss this exam with anyone. I did not receive help from anyone or give help to anyone. I may have found resources on the web, in my notes, and/or in my book. I also state that I did not take more than 2 hours to complete this exam.

Signed: _____

Date: _____

Name (printed): _____