Stephen Smith
CMSI 402
Assignment #1
4 February 2015

Problem 1.9
Designing a dishwasher with few than 10 different washing cycles could be easily accomplished using the waterfall paradigm. It does not require the refinement that the iterative approach would provide. Designing a dishwasher can be done with a straight-forward design, implement, and maintenance approach. A dishwasher is not an item that requires updating (at least for now) and evolving.

Problem 1.10
Designing a website for a medium-sized company would work best with an iterative approach. It has a high enough volatility in both the technology used and the knowledge of the project that it would require multiple evolutions.

Problem 2.8
The term *life cycle* is misleading because it gives the impression that somehow, when a program lifespan comes to an end, another replaces it. Even though this is obviously not the case, the term *life cycle* is used more today than *life span model.*

Problem 2.9
The V-model can be used when the waterfall model can be used, but requires, and puts a large emphasis on, testing individual components of the larger system.

Problem 2.10
The prototyping model can help with solving the issue of requirements volatility. By creating a fast prototype and getting feedback from the users the prototyping model eliminates a fair amount of the back-and-forth a project has when deciding what it needs. However, this model also maintains all the other volatilities. It also uses a fair amount of time on the prototyping process.

Problem 3.13
Having an is-a relationship is part of OO technology and is called a relationship of inheritance. Inheritance is a relation between 2 classes: one being the superclass, or base type, and the other being the subclass, or derived class. The base class defines the relationships that are shared by all members of it and its subclasses. For instance, one could model a person's job as a superclass. The subclasses of the "job" superclass would be an real occupation such as a police officer, doctor, programmer etc.

Problem 3.14
A class is a blueprint for an object. It provides both class-wide methods and attributes. An object is an actual instance of a class. In OO technology, all objects must belong to a class.

Problem 3.15
Polymorphism is allows the use of a single interface to be provisioned to entities of different types. An example would be if we had a Animal class with a method called “makeSound”. This function is not implemented with the exception of its signature; instead, we would then have subclasses, like Cow for example, that inherited from the Animal interface and implemented the “makeSound” method.

Problem 3.16
Information hiding involves hiding the details of an object and its class or a function. It is a powerful tool and helps reduce the complexity of a program. It also preserves data by preventing programmers (or hackers) from intentionally or unintentionally changing parts of a program.

Problem 4.1

Problem 4.9
The activity diagram is one of a version control system. In this diagram you checkout the latest source code. You then get an opportunity to edit the code and make changes. When you are satisfied with what you then are posed with the question of whether you are working with a team or not. If you are, then you will want to pull down the latest changes to make sure there is not a conflict with the changes you have made. If there is then you resolve the conflict. In the 3 instances that you are working on a team and have no conflicts, working on a team and have resolved the conflicts, or not working on a team and are done making changes, you may then commit and be done with the activity of make a commit in a version control system.