Alan Chan

010063326

**Homework 3**

16.1 – Discuss the three “parts” of a design pattern and provide a concrete example of each from some field other than software.

The three “parts” are context, a problem, and a solution. Context gives reader the environment that the problem is residing in. And the requirements are the constraints that influence how the problem can be solved in a certain way. As an example, I will use the event of purchasing a car. The problem set is that a student just graduated and needed a car to get around (including to hangout with friends and going to work). That is the problem. The require that that graduate has set three requirements for the car, it’s not too expensive, it’s fun to drive and it has to be practical. First car comes to mind is a Mazda Miata, it is known to be fun to drive and it’s not expensive. Although it is not very practical as it is a two seater car. Second brand comes to mind is BMW. It’s fun to drive, and practice but it is definitely more expensive. So both of those solution does not really fit the context to solve the problem. After much research the user went with a Subaru Impreza, it is fun to drive, not expensive and it can seat 4 people with a big trunk.

16.11 – Using the design pattern template presented in Section 16.1.3, develop a complete pattern description for the Kitchen pattern mentioned in Section 16.3.

* Pattern Name - Kitchen
* Problem – A place that is clean and safe to store, prepare and cook food.
* Motivation – When user is hungry and do not feel like going out or wait for deliver.
* Context – Raw ingredients will be delivered to the kitchen and cooked delicious food will be delivered.
* Forces – The kitchen can only store so much food. When cooking, a mess will be made.
* Solution – Must have a garbage to contain unwanted leftovers.
* Intent – Creates a clean and safe place to cook food.
* Collaborations – Food, mess, container, stove.
* Consequences – Used up a lot of space.
* Implementation – Food comes pre-cooked.
* Related patterns – Pantry, room.

19.3 – Using the definition of software quality proposed in Section 19.2, do you think it’s possible to create a useful product that provides measurable value without using an effective process? Explain your answer.

I think it is always possible, but it requires a huge combination of luck, organization and being smart. But an effective process can make sure developers not to miss any corners and keeping them on track. It can almost be like a checklist too, not always though. Good process also creates a measurable value.

19.11 – Are quality and security the same thing? Explain.

In the old days, I think quality and security are two different things. Efficient codes used to be more important than security. When the internet is not as popular, security is more on the physical side. But nowadays, cyber security is important, codes cannot be consider as good quality without considerable protection.