CMPE 285 – Software Engineering Processes

Homework 3

First Name: Harish

Last Name: Marepalli

SJSU ID: 016707314

Professor: Richard Sinn

Book: **Software Engineering: A Practitioner's Approach (Eighth Edition)**

Problem 16.1: Discuss the three "parts" of a design pattern and provide a concrete example of each from some field other than software.

Answer:

A design pattern is delineated by a three-part rule that articulates the correlation between a specific context, a problem, and its solution.

- The context serves to elucidate the setting within which the problem exists and guides towards an appropriate solution within that framework.
- A set of requirements, encompassing constraints and limitations, acts as a conglomerate of forces shaping how the problem can be understood within its context and how the solution can be effectively applied.

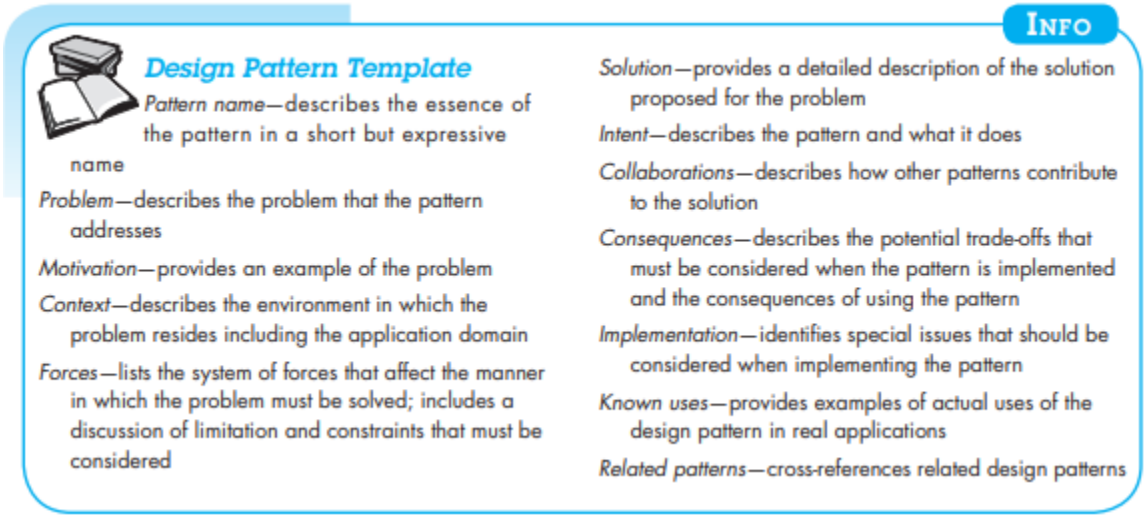
Consider a scenario where an individual needs to travel between two locations.

- In this context, the travel takes place within a developed country utilizing the existing transportation infrastructure such as roads, airlines, and railways.
- The array of forces influencing the approach to solving the travel problem encompasses factors like the urgency to travel from New York to Los Angeles, inclusion of sightseeing or stopovers, financial constraints, the specific purpose of the trip, and the available modes of personal transportation.
- Considering these forces, the problem (traveling from and to) can be more precisely defined. For instance, upon investigation (requirements gathering), it's revealed that the individual possesses limited financial resources, only owns a bicycle, intends to embark on the trip to fundraise for a preferred charity, and has ample time available.
- The solution to this problem, considering the context and the influencing forces, might manifest as a cross-country bike journey. If the forces were to change, an alternative solution might prove more suitable.

Problem 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.

Answer:

The design pattern template presented in Section 16.1.3 is as below.



INFO

Design Pattern Template

Pattern name—describes the essence of the pattern in a short but expressive name

Problem—describes the problem that the pattern addresses

Motivation—provides an example of the problem

Context—describes the environment in which the problem resides including the application domain

Forces—lists the system of forces that affect the manner in which the problem must be solved; includes a discussion of limitation and constraints that must be considered

Solution—provides a detailed description of the solution proposed for the problem

Intent—describes the pattern and what it does

Collaborations—describes how other patterns contribute to the solution

Consequences—describes the potential trade-offs that must be considered when the pattern is implemented and the consequences of using the pattern

Implementation—identifies special issues that should be considered when implementing the pattern

Known uses—provides examples of actual uses of the design pattern in real applications

Related patterns—cross-references related design patterns

The kitchen pattern mentioned in Section 16.3 is as below.

> Architectural patterns are a bit different. For example, every house (and every architectural style for houses) employs a **Kitchen** pattern. The **Kitchen** pattern and patterns it collaborates with address problems associated with the storage and preparation of food, the tools required to accomplish these tasks, and rules for placement of these tools relative to workflow in the room. In addition, the pattern might address problems associated with countertops, lighting, wall switches, a central island, flooring, and so on. Obviously, there is more than a single design for a kitchen, often dictated by the context and system of forces. But every design can be conceived within the context of the "solution" suggested by the **Kitchen** pattern.

Pattern description for the Kitchen pattern cited in section 16.3 of the book, using the design pattern template shown in section 16.1.3 is given as.

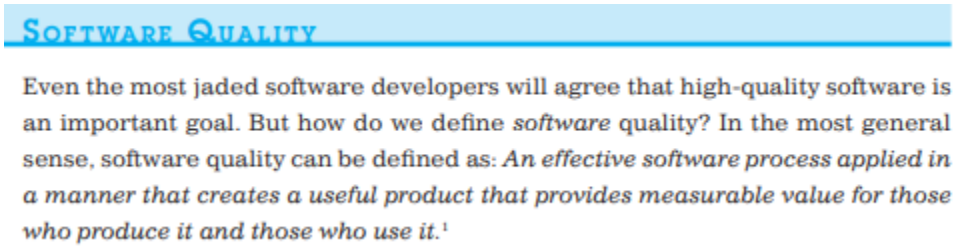
Pattern name	Kitchen pattern
Problem	To store and prepare food, the resources required to prepare food.
Motivation	Problem associated with lighting, wall switches, countertops etc.
Context	The kitchen itself and the appliances, resources already available with the client
Forces	The monetary and time resources the client is willing to spend on the project.
Solution	• List the resources, furniture, and appliances the client already has

	• Check from where the light enters the kitchen. • Keep an eye on the flow of movement in the kitchen. Keep that area vacant. • If some items are to be bought select as per requirement. • Use a computer aided design system to o Sketch the floor plan of the kitchen including, windows, door and wall dimension o Draw the resources, appliances, and furniture layout. o Draw a 3D picture of the kitchen with resources placed and show it to client. • On acceptance of the interior design, implement it.
Intent	The pattern makes the kitchen functional, convenient and looks aesthetically pleasing and the client is happy with the interiors of the kitchen.
Collaborations	If the client needs to buy new appliances, do it.
Related patterns	In the book called 'A pattern language'.

Problem 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.

Answer:

The definition of software quality proposed in Section 19.2 is as follows.



SOFTWARE QUALITY

Even the most jaded software developers will agree that high-quality software is an important goal. But how do we define *software* quality? In the most general sense, software quality can be defined as: *An effective software process applied in a manner that creates a useful product that provides measurable value for those who produce it and those who use it.*[1]

No, it is not possible to create a useful product that provides measurable value without using an effective process. Because,

• Software engineering practices allow the developer to analyze the problem and design a solid solution to build high-quality software.

• Umbrella activities such as change management and technical reviews have as much to do with quality as any other part of software engineering practice.

Problem 19.11: Are quality and security the same thing? Explain.

Answer:

No, quality and security are not the same things, but software security relates entirely and completely to quality. Software that does not exhibit high quality is easier to hack, and as a consequence, low-quality software can indirectly increase the security risk with all of its attendant costs and problems. So, to build a secure system, you must focus on quality.