**Assignment 1**

**Part 1:**

1-10:

**A passenger aircraft is composed of several millions of parts and requires thousands of persons to assemble. A four-lane highway bridge is another example of complexity. The first version of Word for Windows, a word processor released by Microsoft in 1989, required 55 person-years, resulted into 249,000 lines of source code, and was delivered 4 years late. Aircraft and highway bridges are usually delivered on time and within budget, whereas software is often not. Discuss what are, in your opinion, the differences between developing an aircraft, a bridge, and a word processor that would cause this situation.**

First of all, this question seems to imply that software is unique in its propensity to be delivered over budget and late. In my experience laying a new highway almost always takes longer than they say it will. However, it is fair to say software is more often over budget and more often late, and there are several reasons for this.

First, if we are talking about examples like Microsoft’s first iteration of word, we are talking about novel solutions that are often being designed and created either for the first time, or as a bespoke product to fulfill specific requirements. In contrast, the simple manufacturing of a passenger airline is following a design that has already been created, perhaps it needs a different coat of paint, or certain features customized to the needs of the airline, but at its essence it is still the same plane. Likewise with a highway, highways have been designed and created in the past, the same techniques and basic technologies will be reused, just applied to the specific landscape, and needs of that particular stretch of highway. This is more accurately compared to modular software solutions that can be customized for a client’s specific needs. Examples include Salesforce, EDI solutions, and other software suites that are forks from an established base, just with certain features being added, or the appearance being changed. This is very similar to how an airline will order an airplane with custom paint and with or without specific features.

However, software also has its own unique set of difficulties. When dealing with the creation of a new piece of software, you are creating something intangible with many hands working on it. Often times, the parts are all interconnected. Changing how one-part functions may have unanticipated consequences for the rest of the software. Tangible products are typically more compartmentalized. Changing one stretch of a highway is unlikely to completely break a stretch that has already been built miles down the road. Good software engineering practices seek to minimize unintended consequences like this by establishing standard practices and behaviors, just like manufacturing and construction adhere to standardized safety measures and manufacturing practices. However, software engineering is a relatively new practice when compared with the hundreds or even thousands of years that manufacturing and construction disciplines have had to evolve.

2-4

A group of white ovals with black text

Description automatically generated

2-13

A black background with white lines

Description automatically generated

3.6

(I was a little confused about this one. Did you want us to do 3-6 or just read 3.6? Also my book says I should use the meeting described in 3.4.1 but there were several types of interactions described in that section and none of them were titled “meeting” I did my best creating an UML activity diagram utilizing swim lanes based on the description of a meeting later on in the chapter)

**A black screen with white rectangles

Description automatically generated**