Name:	**Software Engineering**	**Topic**
Surname:	Homework 2, Solution	Agile Software Development

Q1) Suggest a small-scale software project that best fits for plan-driven approach. Explain the project very briefly. Also define the customer-side and the software-development company-side situation for this software project. Note that, I expect a novel project suggestion from you.

Plan-driven approaches (such as waterfall) are best suited for large-scale project or contract-based project where requirements are well-understood. But, plan-driven approaches can be still usable (in some scenarios must be used) even if the project is small in case software needs to be safety-critical. For example, a target tracker for a missile or a dose injector for a health device.

Assume that you are working in the research and development department of Ministry of the Health (Sağlık Bakanlığı) of Turkish government. They want to develop a handheld device which will measure various sensor data and decide if the person has a certain virus for a possible future epidemic. If person has that virus then the system should certainly detect it (at least with 99% detection rate). However, a few false alarms can be acceptable. Also, system should work fast. So, most of the development of this handheld device is hardware development and a small embedded software will be also developed. Although developed software is small one, it is very critical, and it should run as expected otherwise your fight against that virus will be harmed.

Q2) Suggest a small-scale software project that best fits for incremental approach, specifically an Agile approach. Explain the project very briefly. Also define the customer-side and the software-development company-side situation for this software project. Note that, I expect a novel project suggestion from you. Which Agile approach you will use and why?

Now, lets talk abut our current situation with Covid-10 virus epidemic. Now, the development is not for a possible future epidemic, but you are already fighting against it. This time Ministry of the Health of Turkish government decide to hire several research companies to develop such a handheld device with maximum detection performance with minimal development time. Assume that you are working in a small innovative research and development company that wants to develop this handheld device. Success in such a development has many benefits: it is transcendental because it will help saving human lives, it can provide huge profit, and it is a very good advertisement of the company. However, time is gold and you can not wait for 1 year to complete the development. Instead, you will use agile approaches to develop this device and maybe in 4 months you will deliver the hardware with first version of the software. Then government can start using it. Meantime, you can improve the software and this handheld device can update their software online since they are connected to the Internet with 3G. And, if your hardware design is also flexible then you can even update the hardware in future. You can use Scrum for the development of such a device since it provides incremental development and several releases of the software. You can also consider eXtreme Programming (XP) if you have a team with the working knowledge of XP, but Scrum is more than enough.

Q3) Explain the documentation load for Q1 and Q2. What are the necessary documentations for Q1 and Q2, when and how they will be produced?

In Q1, plan-driven approach (i.e. waterfall) is used. And, in waterfall, you need the generate comprehensive documents in sequence. First you should create Software Requirement Specification (SRS) document, then Design document and Test document. Since Design document and Test document uses SRS they can be created simultaneously.

In Q2, an agile approach (i.e. Scrum) is used. And in agile approaches, documentation load is very light, almost none. In agile approaches, you should create software-architecture design in a comprehensive way since refactoring the software-architecture is difficult and expensive. Then you will have user stories, tasks that acts as requirements that will be collected in Scrum backlog. In each iteration these contents will get larger and larger. There will be no design document at all since software itself is thought as the design document in agile approaches. You will have small test documents for each sprint (i.e. if you are using test-driven methodology). And since you will refactor the code in each iteration your design (which is the code itself) will be also improved in time.

Q4) Explain why the software project given in Q2 can be developed best using the Agile Manifesto. Give explanation for each item in the Agile Manifesto.

Individuals and interactions over processes and tools → Processes and tools are too rigid to overcome problems through which the team encounter. Therefore, it is much more productive to solve them by interacting. Also, in this way, the team can come up with different ideas.

Working software over comprehensive documentation → For both customer and the team, taking feedback based on a working software is much better and more understandable than taking feedback based on a comprehensive documentation.

Customer collaboration over contract negotiation → The team and customer should collaborate to find the best way by focusing on the customer satisfaction.

Responding to change over following a plan → There are various features that can be added to the product in Q2 today. New developments and discoveries about Covid-19 virus should reflected to the developed handheld device and the embedded software contained in it.

Q5) Which practices of Extreme Programming (XP) are useful for Scrum? Why?

Refactoring: It is useful because it enables developers to generate high-quality, clear and understandable codes, therefore reducing the need for additional documentation.

Small releases: Even though release period is longer with respect to XP, the team obtains a potentially shippable product at the end of each sprint. This enables users to use the released product; presents an opportunity to get a feedback from the product owner.

Continuous integration: It is beneficial because it enables us to detect and fix error faster as well as improving quality and testability.

Sustainable pace: It is useful because keeping the team active through projects is essential for productivity.

Pair programming: It is useful because it enables the developers working on the same software to understand and have a comprehensive knowledge of the project. They can help themselves to enhance code quality by sharing their knowledge.

Test-first development: It is beneficial because it strengthens the understanding of user stories (requirements). The tests written and tested previously are tested as new test cases are implemented. So, the cases are tested more than once and become more mature in time.

Simple design: It can be useful because it ensures that each team member can easily understand code, therefore decreasing documentation need.

Incremental planning: It is useful because it reinforces the understanding of the user needs by getting feedback from the released product and it reduces the risk of missing changes in the business. Moreover, it is got more manageable by the project team.