NAME: NELOY SAJJAD MAHADI

STUDENT ID: 2021290010

CLASS : SOFTWARE THEORY AND METHODOLOGY

MAJOR: COMPUTER SCIENCE

COUNTRY: BANGLADESH

I. 1. **Formal application types description**

Refer to the fairly broad variety of theoretical computer science fundamentals.
Described as :

- Logic calculi
- Formal languages
- Automata theory
- Discrete event dynamic system
- Program semantics
- Types of systems
- Algebraic data types

2. Quality Standards of software product are:

Reliability - enduring and consistent performance in real world conditions.
Availability - The availability of a service.
Usability - ease of use.
Consistency
Performance - requirements such as the responsiveness and speed of a user interface.
Maintainability - requirements that things be easy to maintain and fix.

3. Requirements elicitation and analysis

- It's a process of interacting with customers and end-users to find out about the domain requirements, what services the system should provide, and the other constrains.
- May involve end-users, managers, engineers involved in maintenance, domain experts, trade unions known as stakeholders.
- It has 4 main process.

a) Requirements discovery.
b) Requirements classification and organization.
c) Requirements prioritization and negotiation.
d) Requirements specification.

II. 1. Description for the user story.

- A description of the starting situation;
- A description of the normal flow of events;
- A description of what can go wrong;
- Information about other concurrent activities;
- A description of the state when the scenario finishes.

Name	Money transactions
Actor	Customer and Bank system
Description	It shows the balance amount, the money which can be withdrawn, and deposit and transfer options.
Pre-condition	The customer is logged in
Post- condition	The customer deposit,and transfer or withdrawal.
Actions(Main Scenario)	1. The customer will press the deposit from home page. 2. The customer select to transfer the money . 3. The customer withdraw the money. 4. Press confirm 5. Confirmation message upon success.
Exceptions	#User entered invalid input, thus an error message will be displayed

2. The extreme programming release cycle:

Plan Release : This is the first stage of the Extreme Programming development life cycle.Its main task is to set goals of the entire project and certain iterative cycles. At this stage the team meets with the customer and asks him on all aspects of the future software.

Develop: At this stage of the project the team must define the main features of the future code. The main thing is to create a simple design, because simplicity is one of the main principles of XP methodology. Extreme Programming developers often share responsibilities at the stage of designing.

Release software: the sum of the stages of development and maturity for a piece of software.

Evaluate system: measuring the final system against its initial performance goals as well as performing ongoing testing to see that the system continues to meet those goals

User stories for this release: a set of functionality that makes sense to deliver to users.

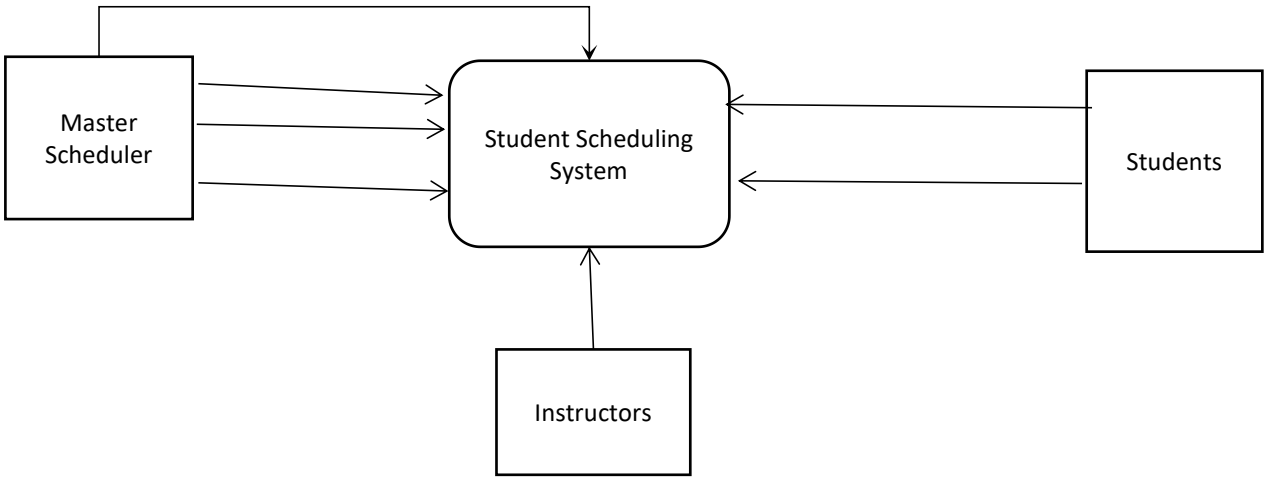
3. The process improvement cycle:

Step 1: Define. Identify the target process.
Step 2: Identify. Identify the process customers and suppliers.
Step 3: Select. Establish desired performance goals.
Step 4: Implement. Develop an action plan to achieve the goals.
Step 5: Evaluate. Establish ongoing feedback

III.

1. Context model of a student scheduling system and description:

The student scheduling system that handles : inputs information about courses, instructors and sections to the system.
Students can make inquires about the available sections for each course , and can also add or drop a section.
Student schedules are available to the students, course rosters are available to the instructors and management reports are available to the master schedule.



2. Structural models , with example:

- Structural models is the architectural map for a large software system or family of systems.
- It display the organization of a system in terms of the components that make up that system and their relationships.
- **Diagrams which can be used to describe structural models:**

I. Structural diagrams
II. Package diagrams
III. Profile diagrams
IV. Class diagrams

3.Description for the weather station state diagram

- A weather station is a package of software controlled instruments which collects data, performs some data processing and transmits this data for further processing.
- The instruments include air and ground thermometers, an anemometer, a wind vane, a barometer and a rain gauge. Data is collected periodically.
- When a command is issued to transmit the weather data, the weather station processes and summarises the collected data.
- The summarised data is transmitted to the mapping computer when a request is received.

IV. 1. Research interest directions:

Progress has been made in the Creating and research program for better software engineering systems, but many important research questions and technical support needs remain. In Chapter 3, a framework is suggested for evaluating banking system initiatives that gives priority to research that improves response and recovery and/or develops risk reduction or consequence mitigation measures. The research also produce tools with a reasonable likelihood of implementation and, where feasible, dual-use benefits. Based on this framework and the review of security efforts already under way, two important security research gaps are identified and discussed briefly in this chapter. In addition, short- and long-term system security needs to be upgraded and from here my research recommendations are made.

3. Apply some knowledge of this course "Software Theory and Method" to my future scientific researches:

Software Theory and Methodology engineering is knowledge-intensive and intellectual capital is crucial. Intellectual capital may be divided into human, social and organizational capitals. A theory of software engineering is formulated based on these three capitals. The theory is based on industrial observations and illustrated in a case study. The theory is used by both researchers and practitioners in real life and I am going to make use of this software theory and method in my future research papers.There exists no generally accepted theory in software engineering, and at the same time a scientific discipline needs theories.Several researchers and initiatives emphasize the need for theory in the discipline.The theory is generated from empirical observations of industry practice, including several case studies and many years of experience in working closely between academia and industry. The theory captures the balancing of three different intellectual capitals: human, social and organizational capitals, respectively. The theory is formulated using a method for building theories in software engineering. It results in a theory where the relationships between the three different intellectual capitals are explored and explained.With the base line of powerful theory and methods , I conclude that I will use this knowledge in all the researches I am going to do in future since this was the main base (foundation) of software engineering at large.