Mirza Ovcina 200272124
CS373 Assignment #1

1.

***Define the fielded of Software Engineering*.**

Software engineering is an approach to software development that borrows practices from other engineering fields by placing emphases on scalability, reliability, maintainability, and safety of large-scale software projects. These practices include project evaluation, requirement specification, rigorous testing and finally long-term maintainability. Software produced by following strict software engineering practices should, in theory, be safer, maintainable and modular when compared to software developed using conventional development practices.

2.

Software Engineering means "Programming in the Large". Explain what does this mean.

Programming in the large describes large-scale software development projects that require many thousands of lines of code that are written by teams of programmers over long periods of time.

3.

What are all the problems related to the software crisis?

Amongst many problems that plugged software industry (trends which more or less continues today) software crisis describes the following:

Projects which are not respecting time constraints
Projects that are not respecting budget constraints
Release of untested or unsafe software (Therac-25)
Non-modular and unmaintainable software
Software that fails to solve the problem

4.

Explain why do we still need Software Engineering.

Today's software is more complicated than ever and developing it is no longer a task that could be accomplished by single or even small group of programmers. In addition to large teams of programmers, modern software projects require a wide range of experts that need to work together in a cohesive manner in order to produce high-quality software that meets all the constraints, is on budget and delivered on time.

5.

Using real-life examples, explain the differences between generic and customized software products.

Generic software is generally distributed to the public as compiled program that can be obtained either for free or at a cost. Specifications of such program are made as general as possible in order appeal to many users as possible. Some practical examples include email clients, web browsers, word processors and many more.

Bespoke software, on the other hand, is any software which is made under specifications that are aimed at fulfilling needs of a particular application or a particular business. For example, assembly line robots that are found inside car manufacturing plants would be running software specifically designed for such purpose.

6.

Explain concretely the two problems related to the year 2000.

Early software designer underestimated the lifecycle of their products and in an attempt to save resources they allocated only two least significant digits to the year in the date data. The problem with their strategy became apparent in the mid-1990s when many people were left wondering how the software will react when the year 2000 finally rolled around. Mush of fear came from banking industry especially when it came to calculations involving interest. Many speculated whether computers would correctly assume 00 to mean the year 2000 or 1900.

The second problem was that software developers worked under incorrect assumption that leap years do not happen if a year is evenly divisible by 100. The year 2000 is obviously divisible by 100 and is also happens to be a leap year which lead to a contraction and the discovery of the flaw.

The two before mentioned flaws have cost a combined estimate of $2 billion in downtime and code repairs.

7.

Identify the major activities in the software management.

In software engineering, software management is mainly concerned with feasibility and successful execution and completion of a project. Specific management duties include:

Ensure software development is “worthwhile” or feasible.

Estimate time needed to complete and cost of the project.

Continuously track and ensure development progress is in line with estimates.

Make sure specific requirements and budget constraints are respected during development.

8.

What is the difference between software development and software maintenance?

Software development can be seen as an entire process from initial consumer problem description, all the way to the development and implementation phase. Maintenance, on the other hand, is a continued effort that encompasses existing software improvements through bug fixing, optimizing as well as new feature implementation.

9.

Describe in detail an example of software failure but not covered in the lecture*.*

On February 26th of 1994, 27 American soldiers lost their lives and further 98 were injured in one of the deadliest attacks (on American Forces) of the Gulf War. The attack was executed by a lone Iraqi Scud missile which managed to penetrate military air defenses, striking and exploding inside of a crew housing barrack. Following the attack, much of the blame was attributed to the ineffective operation of Patriot air defense system. To farther complicate matters, knowledge of a software bug has been known for some time before the attack, however, the crucial update, which would have prevented the disaster, was not deployed in time. It was discovered that a computers running Patriot targeting systems relied on a flawed time representation and storage. Patriot hardware had maximum floating point accuracy of 24 bits; this was very problematic constraint since rest of the system needed timekeeping precision of up to 1/10 of a second ("Lethal Software Defects: Patriot Missile Failure « Barr Code"). Because 1/10 cannot be properly represented in binary, the resulting value is prone to a small amount of rounding error. The error would compound linearly with each hour the system was in standby operation. Finally, during the missile launch, the targeting system would lock onto enemy missile using radar and would continue to track its path using algorithms in order to estimate its position while in radar blind spot. Just prior to engaging the target, Patriot would steer towards the position it thinks the enemy missile would be, but because of the timing error, its algorithms would guide it off course just enough to miss its target and fail.

10.

Define “Legacy code”.

Although there exists more than one concrete definition of legacy code, there appears to be an informal definition among programmers that defines legacy code as original source code which is flawed, hard to maintain and is a general nuisance to work with. George Olivetti, a programmer who coined the term, used “legacy” it to describe any code which one maintains but is not the original author of. Michael Feathers on the other hand defined legacy code as codebase which has not been tested and is therefore difficult to maintain (J. Salé, Michael).

11.
List the four types of artifacts produced in the software development process.

1. The software: source code and/or binary executable
2. User manuals
3. Requirement specification documents
4. Quality testing documents

12.
Based on a concrete example, describe the role of the different parties in the software process:

-User: This is a person that will ultimately be the end user of the software suite. For example, the user of a flight management systems (FSM) in an airplane is almost always a pilot. Apart from being users, pilots and users in general, also be involved in drafting of initial specifications as well as testing (black box).

-Customer: Customer is an entity who contracts the development firm and ultimately provides specification for the project. Role of a customer may also include testing (black box). The customer is also responsible for the cost of the development efforts.

-Developer: Developers are software writers and engineers that follow software requirements and produce software which is released to the customer. Developers are also included in the early specification drafting as well as testing (black and white box).

-Manager: Managers work with customers and developers in order to ensure the project is developing in a timely manner, facilitate communication between all parties and oversee any general employee/customer needs.

13.
Why do we need the feasibility study of software systems?

Before work on the project can begin, a feasibility study is needed in order to ensure that solution to the problem exists, resources and domain experts can be accessed and the overall project profitability expectations are met.

If, for example, the project is expected to be unprofitable cancellation may be required at an early stage in order to prevent further losses.

14.

Choose a problem from any application domain and produce its requirements specification document.

1. Problem Definition

Create an online classified advertisement website that caters to buyers and sellers of used vehicles and automotive parts.

2. Requirements elicitation and analysis

Website should have three user types: Buyer, Seller and Administrator

2.1 Functional requirements from buyer

View advertisements: text, images, price
Search for items by: type, make, model, price and year.
View seller details: email and phone number

2.2 Functional requirements from seller

Same as buyer in addition to:
Account creation
Log in/Log out
Post advertisement with textual description, images and pricing
Post contact details that include phone number, email or both
Choose category of advertisement: car, truck, van …
Choose make model of vehicle from database
Edit or delete his/her advertisement

2.3 Functional requirements from administrator

Log in/Log out.
View member list and their history.
View advertisement list.
Remove or alter advertisements.
Remove and/or ban users from creating future user account.
View server status and statistics such as: load and user demographics.
Change basic look of website including background color, logos and basic body messages.
Change form requirements for account creation and advertisement posting.

3.0 ***Quality requirement***

Correctness
Advertisement should be properly categorized based on user input (make, model, year).
Advertisement must properly display contact information.
Buyer messages must be forwarded to seller email address.

Robustness
System should tolerate incorrect or missing user login data.
During creation of an advertisement, system must be able to deal with missing or incorrect user input data and be able to accept and handle all common image formats and sizes.
Website should remain accessible even if single server is taken offline

Performance
System should handle many concurrent logins, postings and account creations.
System should authenticate user in under 3 seconds.
Advertisement must appear on website within 5 minutes of posting.
System should automatically resize seller provided images in order to limit page load times to under 5 seconds (on high speed internet connection).

Security
Passwords and user details have to see encrypted in case of physical data breach.
Users may only access data allowed to his/her respective user type (elaborated in user roles above).
User form fields should accept alphabetic and numerical data only.
User passwords must be at least 8 characters long and must include one or more numerical data type.

User Friendliness
User interface should be familiar and intuitive as possible.
Forms should have helpful requirement explanation.
System should have informative error messages.

15.
Explain in detail the software design phase*.*

Software design phase is one of the most important steps in the development of quality code that is delivered on time and budget. The initial steps of design phase include developing a plan which dictates how the code will be written. In order to come up with the plan, complex systems are decomposed and attacked individually until such time that a model can be created. Representation and/or models can be expressed via the use of graphical modeling languages such as UML or Behaviour Trees and more formally though FSM and mathematical analysis in case of safety/mission critical software applications. This is particular step is considered the top-level design and is primarily concerned with components and communication interfaces between them. On a lower level (detailed design), the underpinnings of the components are refined and formally described. This includes algorithms as well as data structures and their use. Finally, architecture style is chosen and team members are assigned their respective roles.

References

J. Salé, Michael. "A Case Study On Improving Quality During Legacy Software Maintenance Using A Heuristic." IEEE Software, 2017, Institute Of Electrical And Electronics Engineers (IEEE).

"Lethal Software Defects: Patriot Missile Failure « Barr Code." Embeddedgurus.Com, 2018, https://embeddedgurus.com/barr-code/2014/03/lethal-software-defects-patriot-missile-failure/.

Vliet, Hans van. Software Engineering Principles And Practice - 3Rd Ed. John Wiley & Sons, 2008,.