Software Architecture and Design

Lab Manual

Problem Statement and Requirement Gathering

Week 4 Lab

Made By:

M Faizan Khan

Session: Fall 2024

(South China Normal University, China)

Week 4- Lab

Objectives

- Identify the prblems present around you .
- Learn to devise a software Solution to that problem.
- Discuss with team mates in order to know the exact team work working environment.

Introduction to Problem statement

The problem statement is the initial starting point for a project. It is basically a one to three page statement that everyone on the project agrees with that describes what will be done at a high level. The problem statement is intended for a broad audience and should be written in nontechnical terms. It helps the non-technical and technical personnel communicate by providing a description of a problem. It doesn't describe the solution to the problem.

The input to requirement engineering is the problem statement prepared by customer. It may give an overview of the existing system along with broad expectations from the new system.
The first phase of requirements engineering begins with requirements elicitation i.e. gathering of information about requirements. Here, requirements are identified with the help of customer and existing system processes. So, from here begins the preparation of problem statement. So, basically a problem statement describes **what** needs to be done without describing **how**.
Conclusion: The problem statement was written successfully by following the steps described above.

Purpose:

The main purpose of the problem statement is to identify and explain the problem. This includes describing the existing environment, where the problem occurs, and what impacts it has on users, finances, and ancillary activities. Additionally, the problem statement is used to explain what the expected environment looks like. Defining the desired condition provides an overall vision for the process or product. It makes clear the purpose for initiating the improvement project and the goals that it is meant to accomplish.

Another important function of the problem statement is to be used as a communication device. A problem statement helps with obtaining buy-in from those involved in the project. Before the project begins, the stakeholders verify the problem and goals are accurately described in the problem statement. Once this approval is received, the project team reviews it to ensure everyone understands the issue at hand and what they are trying to accomplish. This also helps define the project scope, which keeps the project concentrated on the overall goal.

The problem statement is referenced throughout the project to establish focus within the project team and verify they stay on track. At the end of the project, it is revisited to confirm the implemented solution indeed solves the problem. A well-defined problem statement can also aid in performing root cause analysis to understand why the problem occurred and ensure measures can be taken to prevent it from happening in the future.

It is important to note that the problem statement does not define the solution or methods of reaching the solution. The problem statement simply recognizes the gap between the problem and goal states. It can be said that "a problem well stated is half solved". However, there are often multiple, viable solutions to a problem. Only after the problem statement is written and agreed upon should the solution(s) be discussed and the resulting course of action determined.

Defining the Problem:

Before the problem statement can be crafted, the problem must be defined. It is human nature to want to begin working on a solution as soon as possible and neglecting the definition of the true problem to be solved. However, a poorly defined problem increases the risk of implementing a solution that does not fully meet the expected results. A problem cannot be solved if it is not completely understood.

The process of defining the problem is often a group effort. It starts with meeting with the stakeholders, customers, and/or users affected by the issue (if possible) and learning about their pain points. Since people often struggle with effectively communicating their issues, particularly to someone outside of the process, it is helpful to ask a series of "why" questions until the underlying reasoning is identified. This method, known as the five whys, helps drill down to the core problem as many of the experienced frustrations could be mere symptoms of the actual problem. Asking these additional questions as well as paraphrasing what the stakeholder had said demonstrates a degree of empathy and understanding of the problem.

The information collected from these initial interviews is only one part of problem analysis. Many times the problem extends to multiple areas or functions to which the stakeholders, customers, and users are unaware. They may also be familiar with what is happening on the surface but not necessarily the underlying cause. Therefore, it is just as essential to gather knowledge, information, and insights from project team members and subject matter experts concerning the problem. Additional research materials, including work instructions, user manuals, product specifications, workflow charts, and previous project plans may also need to be consulted. Like most other stages in the process improvement project, defining the problem is often iterative as several rounds of discussions may be needed to get the full picture.

Once the problem is understood and the circumstances driving the project initiation are clear, it is time to write the problem statement.

Writing the Problem Statement:

The problem statement will be used to gain project support and approval from stakeholders. As such, it must be action-oriented. More importantly, the problem statement must be written clearly and accurately in order to deliver successful results. A poorly crafted or incorrect problem statement will lead to a faulty solution, as well as wasted time, money, and resources.

There are several basic elements that can be built into every problem statement to decrease the risk of project failure. First, the problem statement must focus on the end user. A common mistake is focusing on how a problem will be solved rather than the current gap. Second, the problem statement should not be too broad. A benefit of using the five whys approach is that it avoids over-simplicity by providing the details needed for understanding the problem and developing an appropriate solution. Finally, the problem statement should not be too narrow. Solution-bias stifles the creativity that arises while brainstorming a solution, which may result in less-than-optimal experience for the user.

It is useful to design and follow a specific format when writing a problem statement. While there are several options for doing this, the following is a simple and straightforward template often used in business analysis to maintain focus on defining the problem.

- Ideal: This section is used to describe the desired or "to be" state of the process or product. It identifies the goals of the stakeholders and customers as well as assists in defining scope. At large, this section should illustrate what the expected environment would look like once the solution is implemented.
- Reality: This section is used to describe the current or "as is" state of the process or product. It explains pain points expressed by the stakeholders and customers. It should also include the insights and expertise of the project team and subject matter experts provided during problem analysis.
- Consequences: This section is used to describe the impacts on the business if the problem is not fixed or improved upon. This includes costs associated with loss of money, time, productivity, competitive advantage, and so forth. The magnitude of these effects will also help determine the priority of the project.
- Proposal: This section is used to describe potential solutions. Once the ideal, reality, and consequences sections have been completed, understood, and approved, the project team can start offering options for solving the problem. It can also include suggestions by the stakeholders and customers, although further discussions and research will be needed before a specific course of action can be determined.

Lab Tasks

Lab Task

Problem Overview:

The objective of this project is to design and develop a robust e-commerce platform that meets stringent quality standards while delivering a seamless and user-friendly shopping experience. The platform will serve as a web-based marketplace where users can browse various products, add them to their shopping cart, and complete secure purchases. The system should not only meet functional requirements (like product browsing and secure payment) but also excel in non-functional areas, such as high performance, strong security, scalability, usability, and maintainability.

With increasing competition in online retail, a high-quality e-commerce platform is essential for ensuring business success. The platform must be responsive, secure, and scalable to accommodate varying traffic loads and evolving business needs. Additionally, it must be built with long-term maintainability in mind, so that future updates, such as new features or security patches, can be incorporated without disrupting user experience or system performance.

Background:

E-commerce platforms have rapidly become essential in the digital economy. With an increasing number of users engaging in online shopping, the reliability and quality of these systems are critical to business success. Failures such as poor performance during high-traffic periods, security breaches, or difficult-to-navigate interfaces can result in loss of revenue and damage to the brand's reputation. Therefore, developing a solid quality assurance model ensures that key system qualities are met throughout the software development lifecycle.

What you need to do?

Write a Problem Statement:

- Start by defining the key challenges facing e-commerce platforms, such as poor system performance during peak traffic, security vulnerabilities during payment processing, or difficulty in navigating the platform.
- Describe the need for a solution that addresses these problems and ensures the platform is reliable, secure, and easy to use.

Write a detailed problem statement detailing why there is a need to develop this software.

Gather Requirements:

- Interview key stakeholders (Think of you as a stake holder) (e.g., customers, business owners, and developers) to understand the expectations for the platform.
- Define functional requirements, such as product browsing, cart management, and secure payment gateways, and non-functional requirements such as system performance, security measures, and usability standards.
- Make sure to document requirements related to scalability (e.g., handling 10,000 users simultaneously) and maintainability (e.g., ease of adding new products or features).

Gather the software requirements in very detail as mentioned in Course Slides.

Refine Requirements:

- After gathering initial feedback, refine and prioritize the requirements based on feasibility, cost, and criticality.
- Ensure that performance requirements (e.g., page load time under 2 seconds), security needs (e.g., encrypted payment data), and usability standards (e.g., minimal clicks to complete a purchase) are well defined and realistic.
- Consider the trade-offs between conflicting requirements (e.g., security vs. usability) and adjust the priorities accordingly.

Write and refine the requirements in details.

Design the Platform:

- Begin sketching out the architecture of the system, including the front-end user interface and the back-end services (e.g., databases, payment gateways).
- Make sure the platform is designed to allow easy updates and improvements in the future without requiring a complete system redesign.

You just need to create a dummy designs . No need to go in details.

Testing and Evaluation:

- Plan a testing phase that includes performance tests (e.g., load testing), security tests (e.g., vulnerability scanning), and usability tests (e.g., user feedback sessions).
- Define metrics for success, such as system uptime, average response time, and user satisfaction scores, and make adjustments based on the results of the tests.

You need to create a simple plan for testing.