American International University Bangladesh

Department of Computer Science and Engineering

Cover Sheet

Course Name	SOFTWARE REQUIREMENT ENGINEERING
Section	**A**
Group Members	
Student Name	**Student ID**
NUSRAT JAHAN	**17-34562-2**
MD. TANVIRUL HASAN	**18-37078-1**
SHIHAB HOSSAIN SHUVO	**18-36385-1**
MOHAIMENUR RAHMAN	**19-40338-1**

Title: Assignment on Requirement Collection

Suppose you are a project manager of a software company. Your company got a big software project to develop for a poultry and frozen food company. Project is to automate their company with an ERP (Enterprise Resource Planning).

Now plan and describe how you can successfully collect full requirements from the company and start development. We all know without collecting requirements properly software cannot be developed perfectly.

To develop a software program that satisfy the client's necessities, at the beginning their requirements must be accrued correctly before begin planning and development. The maximum crucial element is to understand or to select what to develop. If we will not figure out what to develop and the goal of improvement, then the destiny levels could be effected fairly. No other matters are tough extra than accurate later. Many product designers debate that things will become clear as they come together, that project partners will only want to understand need after inspecting early stages of the product, the designers also have debate on that many things change so quickly that any attempt to fully cover requirements is unsuccessful, the reality is delivering a working system, and everything else is optional. Necessities engineering creates a strong foundation for planning and development. Without it, the ensuing programming is quite likely to fail to meet the needs of the client.

To implement proper requirement engineering, seven different tasks need to be cover by us. Those are:

1. Inception
2. Elicitation
3. Elaboration
4. Negotiation
5. Specification
6. Validation
7. Management

These are the terms used to describe these processes. Here, we'll discuss the elicitation element of our project. The process looks like simple enough-

1. Ask the customer, the users, and the others what the objectives
   for the system or product are
2. What is to be accomplished
3. How the system or product fits into the needs of the business
4. How the system or product fits into the needs of the business
5. How the system or product is to be used on a dayto-day basis

But it isn't as easy it look like; in fact, it is really difficult. Establishing business goals is an important aspect of elicitation. Our responsibility is to engage stakeholders and encourage them to be open and honest about their objectives. A prioritizing process should be built once the goals have been documented, and a design rationale that meets stakeholder goals can be created.

Elicitation of requirements (also known as requirements collecting) is a process that incorporates issue solving, elaboration, negotiation, and specification. Stakeholders work together to define the problem, offer aspects of the solution, negotiate different approaches, and specify a preliminary set of solution requirements in order to encourage a collaborative, team-oriented approach to requirements collecting.

1. We have to arrange meetings are conducted and attended by both software engineers and other stakeholders.
2. Some rules should be fixed by us for preparation and participation among developers and stakeholders.
3. We have to create a plan that is formal enough to cover all important points but informal enough to encourage the free flow of ideas.
4. We have to choose a customer or a developer or an outsider,who controls the meeting.
5. We have to create a prototype mechanism which can be work sheets, or design sheet.

Our primary purpose is to identify the problem, suggest solution aspects, negotiate different approaches, and define a first set of solution requirements. During our inception, we will generate product requests. Then, we will choose a suitable meeting location, time, and date for stakeholders and software team members to attend a discussion. We will make a product request, which will be distributed to all attendees prior to the meeting date. During the requirements gathering meeting, everyone could contribute to this statement, and a lot more information would be available. The attendees must create a list of objects that are part of the system and other objects that the system will produce, also those objects that the system will use to perform its function. Furthermore, each attendee is asked to create a new list of processes or functions that manipulate or interact with the objects. Finally, a list of constrains that satisfy cost, size, business rules and performance criteria is created. The attendees are informed that the lists are not intended to be exhaustive, but rather to reflect each individual's perception of the system. This list of objects will be pinned to the walls of the room with large sheets of paper, or written on a wall board. Alternatively, the lists could have been posted for review prior to the meeting on a group forum, an internal website, or in a social networking environment. Using these platforms or methods, we could express our motivations or desires. Ideally, each listed entry should be able to be manipulated independently, allowing lists to be combined, entries to be deleted, and additions to be made. Critique and debate are strictly prohibited at this stage. We must resist the temptation to dismiss a customer's suggestion as "too expensive" or "impractical." The goal here is to come up with a list that everyone can agree on. To work on this we must maintain an open mind. Following the presentation of individual lists in one topic area, the group creates a combined list by removing redundant entries, adding any new ideas that arise during the discussion, but not deleting anything. Following the creation of combined lists for all topic areas, a discussion led by the facilitator takes place. To properly reflect the product or system to be developed, the combined list is shortened. The goal is to create a consensus list of objects, constraints, and performance for the upcoming system. To explain more about, we have created mini-specifications for each item on the list, as well as a use case involving the object or service. This mini-specs are presented for discussion to all stakeholders. There are additions, deletions, and further elaboration. It has the potential to discover new objects, services, constraints, or performance requirements that will be added to the original lists. The team working with may raise an issue that cannot be resolved during the meeting at any time. A list containing issue is kept so that, these suggestions can be followed up on later. We must consider non-functional system requirements such as accuracy, data accessibility, and security. As stakeholders express their concerns, we must consider them in the context of the system that will be built. Some questions have to be satisfy while working on the project those are:

1. Are we able to build the system?
2. Will this development process help us to perform better than our competitors in the market?
3. Are sufficient resources available to build and maintain the proposed system?
4. Will the system's performance meet our customers' expectations properly?

Including these other questions will be answered over time. The Quality function deployment is a quality management technique that translates customer needs into technical software requirements. That defines requirements in such a way that customer satisfaction is maximized. We must work within the framework of QFD .To gather the requirements, QFD uses customer interviews and observations, surveys, and examination of historical data such as problem reports including raw data. These data are compiled into a requirements table known as the customer voice table, which is reviewed with the customer and other stakeholders. Once requirements are gathered, an overall vision of system functions and features come out. The work products generated as a result of requirements elicitation will differ according to the size of the system or product to be built. A statement of need and feasibility and a bounded statement of scope for the system or product, a list of customers, users, and other stakeholders who participated in requirements elicitation, a description of the system's technical environment, a list of requirements and the domain constraints that apply to each, a set of usage scenarios that provide insight into the use of the system or product under different operating conditions, and any prototypes are all part of the product works. Each of these work products is reviewed by everyone who participated in the requirements elicitation process. In the context of an agile process, requirements are elicited by requesting user stories from all stakeholders. Requirements elicitation is a service-oriented development technique that focuses on the definition of services that an application will provide. We must implement Use Cases in order to clear the actor's perspective. A use case is a stylized story that describes how an end user interacts with the system under a specific set of conditions. First and foremost, if the attendees are unfamiliar with one another, we must introduce ourselves. We need to go over the agenda, remind attendees of the session objectives, and address any pre-session or concerns. During the elicitation session, all our conversation must keep focused on the goal. We must prepare questions to guide the discussion. We must make suggestions based on their responses. Suggesting ideas and alternatives during elicitation is a sign of a good organization, we must propose ideas. Business analysts can think outside the mental box that people who are too close to the problem domain are limited by. We must demonstrate patience by providing verbal feedback, when something is unclear, or not understandable .We've agreed on some fundamental operating principles, such as start and end time must be decided , hold one conversation at a time, expect everyone to contribute, and direct comments and criticism toward issues rather than individuals. After the rules are established, we must ensure that everyone abides by them. One person, known as the facilitator, must ensure that people in the workshop cover note taking, time keeping, and scope management. A scribe might keep track of what's going on while someone else keeps an eye on the time. We must develop a clear plan and workshop agenda ahead of time and communicate them to participants so that they are aware of the objectives and can prepare accordingly. We must ensure that the proposed user requirements are within the scope of the current project. Each workshop must remain focused on the appropriate level of abstraction for the session's objectives. Groups can easily become engrossed in distracting detail requirements discussions. These discussions take up time that the group could be using to develop a higher-level understanding of user requirements. To keep the elicitation participants on track, the facilitator will need to bring them back in at regular intervals. During elicitation sessions, we must keep an eye out for off-topic discussion, such as design explorations. We must keep participants focused on the session objectives while assuring them that they will have future opportunities to work through any other issues that arise. Quality attributes, business rules, user interface ideas, and other information will emerge during the elicitation discussion. We need to organize the information in a organizing way. We need to show respect for the participant who brought it up. We must allot a specific amount of time to each topic of discussion. When concluding a timed discussion, summarize the status and steps before leaving the topic. We must keep our group size small. Because small groups can complete tasks much more quickly than larger teams. Only those with knowledge, experience, and the authority to make decisions are eligible to participate in elicitation workshops. We must keep everyone interested. We must try to re-engage the person who has dropped out of the process. And we'll have to take their word for it, if there is one. A focus group is a representative group of users who meet in a facilitated elicitation activity to generate input and ideas on the functional and quality requirements of a product. We must create focus group sessions that are interactive, giving all users a chance to voice their opinions. Focus groups must be organized. We'll need to keep them on topic while not influencing the opinions expressed. We may want to record the session so that we can go back and carefully listen to the comments. We could not expect quantitative analysis from focus groups, but rather a large amount of subjective feedback that will be evaluated and prioritized as requirements are developed. Typically, participants in focus groups do not have decision-making authority over requirements. When we ask users to describe how they do their jobs, they are likely to tell details that may be missing or incorrect. This is frequently because the fact that tasks are complex and it is difficult to remember every minute detail. They may be so accustomed to the task that they do not even consider it. We can sometimes learn a lot by observing how users perform their tasks. Because observations take time, they are not appropriate for every user or every task. Limit each observation time to two hours or less to avoid interfering with the users' regularly assigned work activities. We must choose important or high-risk tasks as well as multiple user classes for observations. When using observations in agile projects, have the user demonstrate only the tasks related to the next iteration. By observing a user's workflow in the task environment, Business analysts can validate information gathered from other sources, identify new interview topics, identify problems with the current system, and identify ways in which the new system can better support the workflow.

To ensure that the requirements captured apply to the user class as a whole, not just to that individual, the Business analyst's must abstract and generalize beyond the observed user's activities. A skilled Business analysts can frequently suggest ways to improve the user's current business processes. Throw or ask questions are effective method of surveying large groups of users in order to better understand their needs. They are inexpensive, making them an obvious choice for gathering information from large user populations, and they can be easily administered across geographical boundaries. The analyzed questionnaire results from our survey could be used as input to other elicitation techniques. For example, we could ask question to the user to identify their major complaints about an existing system, then use the results to discuss prioritization with decision makers in a workshop. We can also use questionnaires to solicit feedback from commercial product users. The most difficult aspect of questionnaires is creating well-written questions. We will use questions that include this condition that can help us to cover important requirements.

1. While asking question we have to provide answer options that cover the full set of possible responses.

2. Answer choices that will be provide by us must be both mutually exclusive and exhaustive.

3. Question should not be phrase in a way that implies a "correct" answer.

4. Before asking we have to ensure that we ask the right questions of the right people.

5. It's really frustrating to discover too late that a question was phrased ambiguously or to realize that an important question was omitted. So we have to recheck or test them before moving to the distribution.

6. We should not ask too many question which make people interest less to response properly.

Interface analysis is an independent elicitation technique that entails examining the systems to which your system connects. System interface analysis reveals functional requirements regarding the exchange of data and services between systems. Context diagrams and ecosystem maps are an obvious choice to begin finding interfaces for further study. In fact, if we find an interface that has associated requirements and that is not represented in one of these diagrams, if the diagrams are incomplete. For each system that interfaces with ours, identify functionality in the other system that might lead to requirements for our system. These requirements could describe what data to pass to the other system, what data is received from it, and rules about that data, such as validation criteria. We should also discover existing functionality that you do not need to implement in your system. Through system interface analysis, we may be learning that multiple systems pass orders to the order-management system, which performs the validation, so we don't need to build this function. Document analysis entails examining any existing documentation for potential software requirements. The most useful documentation includes requirements specifications, business processes, lessons learned collections, and user manuals for existing or similar applications. Documents can describe corporate or industry standards that must be followed or regulations with which the product must comply. For packaged-solution implementations, the vendor documentation mentions functionality that our users might need, but you might have to further explore just how to implement it in the target environment. Comparative reviews point out shortcomings in other products that you could address to gain a competitive advantage. Problem reports and enhancement requests collected from users by help desk and field support personnel can offer ideas for improving the system in future releases. Document analysis is a way to get up to speed on an existing system or a new domain. Doing some research and drafting some requirements beforehand reduces the elicitation meeting time needed. Document analysis can reveal information people don't tell us, either because they don't think of it or because they aren't aware of it. We should use the results of this analysis as input to user interviews.

We can divide the activities into three categories to perform the elicitation session those are:

1. Preparation for elicitation,
2. Perform elicitation activities
3. Follow up after elicitation.

Enterprise resource planning (ERP) is a process used by companies to manage and integrate the important parts of their businesses. ERP software applications are important to companies because they help them to implement resource planning by integrating all of the processes needed to run their companies with a single system. Elicitation techniques are divided into several categories such as interviews, workshops, focus groups, observations, questionnaires, system interface analysis, user interface analysis, document analysis. These steps we have covered in our discussion that how these are going to be use by us... Because of our clarification about system requirement questionnaires stage will not be follow by us. As this corporate software going to have a specific requirement to interact with the system so user interface is not coming in our frame. Different user perspective are not going to be map by us in this user interface analysis. These are not required for this elicitation as we have described about the reason behind. While we are going to plan elicitation on our project, we have always keep some steps in our mind those are elicitation objectives, elicitation strategy and planned techniques, schedule and resource estimates, documents and system needed to independent elicitation, expected products of elicitation efforts and elicitation risks. So we are to going to run elicitation in our project on this way to fulfill our objectives and requirements.