Elicitation Report

Author: Stephen Aranda

Describe the process of eliciting requirements for a software project in general including at least four sources and four common elicitation techniques.

The process of eliciting requirements is one phase of five to construct a well-done requirement, the others include expressing, prioritizing, analyzing, and managing requirements. For now, we will focus on the first, which is requirement elicitation. The process of eliciting requirements for a software project consists of many steps, each of which is very important in their own way to design, build, install, and maintain a great program that achieves its goal. Requirement elicitation deals with the origin of software requirements and how a Software Engineer can collect them in order to construct a reliable, efficient program. It is the first step of the Requirement Engineering process. They are the range of activities where information is given by stakeholders, users, and customers to design the solution for a given problem or opportunity. Elicitation occurs all throughout the life cycle of a project's development since requirements that are constructed could possibly evolve.

That being said, there are many sources from which requirements can be elicited from; **Stakeholders, goals, business rules, domain knowledge, and organizational environment** are all sources that I have used to gather requirements in my recent project and they are each reliable. Requirements can be elicited from Stakeholders, who is an individual(s) that is affected by or has an effect on the success of a project. This includes end-user, clients, managers of the end-user, and system administrators. Stakeholder Requirements play major roles in systems engineering, they form the basis of system requirements activities, form the basis of system validation and stakeholder acceptance, act as a reference for integration and verification activities, and serve as a means of communication between the development team and the stakeholders.

Requirements can also be elicited from another source, Goals. Goals refer to the high-level objectives that are met and fulfilled by the creation of the product and usually, they are formulated vaguely. It is very important for the Software Engineer at work to assess the value relative to priority and the costs of goals. as it is also a source of requirement elicitation. Business Rules, another source of elicitation, are statements that constrain an aspect of the product structure since the rules represent the behavior of the business itself. They must be taken into consideration when forming requirements. Having and using Domain Knowledge of the application in development is extremely important in the process of eliciting requirements. Using Domain Knowledge will help the systems engineer who coordinates the creation, maintenance and growth of an organization's computer systems, to better understand the elicited requirements and in engineering complete and unambiguous requirements. If the engineer does not have the domain knowledge, then it should be acquired. The organizational environment is another source of elicitation, Usually, the business process of an organization will already be in place so the software solution should not create any unplanned changes in it. Therefore, it is vital for the Software Engineer to bear this source in mind. Finally, the operational environment is

important as a source of elicitation as it is the environment in which the software will be used, so things such as timing constraints in real-time software could be used to elicit requirements.

Now that the sources of elicitation have been discussed, I will now go over the principle techniques that are commonly used to elicit those requirements. There are many techniques to elicit requirements, however, **Interviews, Scenarios, Prototypes, Observations, User Stories, and Analyzing competitor products** are all techniques that I have found to be useful in my SwiftClock project. Technique methods used will vary from project to project, so it is up to the Business Analyst to decide which method will capture the requirements most effectively.

Interviewing stakeholders is the traditional technique used to elicit requirements. It is very important to ask the interviewee open-ended questions to gather useful information from the clients' needs. It is important to understand the advantages and limitations of interviews, along with understanding how they should be conducted properly. Scenarios are good for providing context to the elicitation of user requirements. The most common kind of scenario is a use case description; For example, a use case named “Log in” would have the description, “A user logs in to access the program functionalities.”.

Prototypes are another type of principle technique used to clarify unclear requirements. They range from paper mock-ups of screens to beta-test versions of software products. They should never be too detailed, to avoid the stakeholder from “anchoring” on a quality of it. Observations are a technique method used for elicitation; It consists of the Software Engineer immersing themselves in the organizational environment and observing how users perform their tasks. It enables the Software Engineer to learn more about user tasks. This technique is instructive since it shows that a lot of user tasks and business processes are too subtle or complex to describe easily.

Another very important technique used to elicit requirements as Software Engineers are User Stories. User stories are short, high-level descriptions of required functionality that is expressed in such a way so that the customer can understand it. Usually, they take on the form: “As a <role>, I want <goal/desire> so that <benefit>”. They are constructed to have just enough information for developers to be able to produce an estimate on the effort that needs to be implemented. Using user stories can avoid wasteful activities. Other techniques exist outside of the core principle ones, such as Analyzing a competitors' product. Analyzing competitors' products can help in eliciting complete requirements.

Describe how the requirements engineering process supports the elicitation of behavioral requirements.

The requirements engineering process supports the elicitation of behavioral requirements in a sort of indirect manner, so to speak. During the process of requirement elicitation, there are many types of requirements that may be constructed. Functional requirements, Non-Functional requirements, Product and Process requirements, and System and Software requirements are all vital Software Requirements Fundamentals that can be captured. Functional requirements describe the functions that the software is to execute, and these, along with the other types could be elicited through the process of requirements engineering. **Behavioral Requirements** describe

all the cases where the system uses the functional requirements, which are captured in use cases. Therefore, by using the requirements engineering process which includes using techniques such as interviewing stakeholders, using User Stories, Prototypes, and many more, to elicit requirements, it supports the process of eliciting behavioral requirements by default since they will also be formed in the process.

Describe the fundamental challenges of requirements elicitation.

Many challenges and issues accompany the process of eliciting requirements. First of all, the clearest challenge/issue to me that almost always arises in elicitation occurs when Stakeholders are not clear about their needs. A lot of the times, initial stakeholder concerns do not serve as requirements since they often lack definition, analysis, and even consistency. This is often due to a lack of understanding of what can and cannot be done through software. So, it is up to the requirement engineer to lead stakeholders from these initial thoughts to more formal statements that will eventually take the form of stakeholder requirements. Another challenge commonly faced in requirement elicitation arises when conflicting requirements are expressed from different stakeholders. It is also possible for a single stakeholder to express incompatible requirements.

A problem of scope is another fundamental challenge that could arise in a software product. Having a scope that is not defined properly could cause a project to get out of hand due to unstable requirements. It is important for the requirement engineer to control the scope of the project, as it is what encompasses what can realistically be achieved within the current project. Changes in scope happen in software all the time, and it is a challenge that a Software Engineer must know how to overcome. That is why it is so important to control the scope of a project, by making expectations clear between the client and yourself, drawing the scope with the client using UML such as a use case diagram, ensuring that the client prioritizes the requirements, and asking the question, “Is this in the scope?” when forming requirements.

Another fundamental challenge in eliciting requirements arises when the requirement change. Requirements often change throughout a software's life cycle. This could be due to the fact that the stakeholder will be exposed to more knowledge and information that would result in the maturing of their needs. When these changes are made, it is vital in the elicitation process for the requirement engineer to accommodate those changes, as not doing so could result in an incomplete, unsatisfactory product in the end. This is why it is so important to continue elicitation methods throughout the entire life cycle of software in an iterative manner.

Describe the specific process you took in eliciting requirements for your project in detail. Include your sources (your list of goals, your list of stakeholders, your business rules, etc.). Identify and describe the techniques you used including your interview questions and answers, observation notes, use cases, user stories, etc.

The elicitation process in my software development life cycle (SDLC) for my project “SwiftClock” had many steps to what would eventually result in a core group of requirements for the Employee TimeCard/Payroll Management System I am developing. I began my elicitation process by determining the sources of which I will be eliciting the requirements. From the

sources mentioned above, I found that the **goals** Ecological Laboratories had for the product, my **domain knowledge**, project **stakeholders**, my organization's **business rules**, **operational environment**, and **organizational environment** were all sources I could use to elicit requirements in my project. Ecological Laboratories Inc had a set of goals that were to be achieved from the completion of my product which are as follows:

- Create software that will improve work productivity in the office for the Director of Human resources.
- Create software that will allow employees to view their own schedules.
- Create software that will allow the Director to generate the necessary reports.
- Create software that will allow the Director to manage employee time cards completely, which include the employees' lunch breaks.
- Create software that will allow the Director to assign training to individual employees that need to complete it.

Once these goals were established, I used them to elicit some functional and non-functional requirements that would help accomplish those goals. Domain knowledge was used to elicit non-functional requirements such as the language that was chosen for this project which is Java. As I discovered what I needed the software to do, I realized that Java would be a good choice since it is statically typed, which will yield benefits maintainability and performance optimization. My project stakeholders, the director of Human resources Abby Richter, along with the product & compliance manager whose name is Gayle Richter, the president Barry Richter, and the CEO Michael Richter were all a very important source of requirements that I elicited for my project. The requirements from these sources formed the basis of my systems requirements activities.

Ecological Laboratories Inc had business rules that were already set in place from the beginning of the life cycle of SwiftClock. The Social Media Policy, Electronic Communication System Policy, Ethical Conduct Policy, HIPPA regulations, and employee privacy laws are all forms of business rules that needed to be taken into account when eliciting requirements in all stages of the software. Finally, the organizational environment, which is the business process of the human resources department was closely analyzed throughout the elicitation process in my project. From seeing all the processes carried out in the current software that is being used by Ecological Labs to seeing what reports must be submitted by the director, this source was used to elicit requirements that would not force unplanned changes in the business process. It was important in my particular product as it is what could make the difference in the organization deciding to stay with their current software or to switch to SwiftClock.

This project used a variety of different principal elicitation techniques which include **interviews, scenarios, prototypes, observations, user stories, and analyzing competitor products.** Interviewing the stakeholders was one of the most important techniques used as it gave me the foundations of almost all of my requirements that were elicited in this process. The list of questions asked to the Major stakeholder, Abby Richter along with their answers are the following:

1. **What would a highly successful solution do for you?**

    1. A highly successful solution would increase work productivity and allow the Director of Human resources to manage employees, generate the necessary reports, and manage the timecards of all the employees.
2. **What goals could this product help you accomplish?**
    1. The goals that can be accomplished for this product include having a more efficient business day that will make business operations run more smoothly. The product will allow the Director to manage all employee information, including monitoring their benefits such as PTO (paid time off) and their timecards for example. The product will also allow the Director to manage the times that employees clock in and out which will help with making sure that all the employees are on track and ready to work.
3. **What aspect of the product excites you?**
    1. The fact that the product will generate the reports necessary for the Director is the most exciting part. Currently the director needs to log in certain information on excel to create specific reports, which affects the efficiency of the work environment for the Director. Also, the Director currently needs to physically log in when employees clock out and back in for lunches. The software that is currently in use does not have this functionality, so the Director is excited about this feature as well.
4. **What qualities (e.g., efficiency, security, reliability, etc.) are critical for the specific parts of the product?**
    1. Security is a huge factor in how successful this product will be. Employee information must remain confidential, along with important company information that will reside within the software. The software must also be reliable enough to have no issues arise during use, otherwise, the Director will need to resort to physically logging certain things and/or use third-party software to carry out the responsibilities that the Director must take care of. The database must also be updated live whenever an employee is added or removed, it will play an important role in how accurate that data is.
5. **Can you describe the environment in which the product will be used?**
    1. The product will be used when changes in things such as employee information, employee timecards, and company staff occur.
6. **What should happen if the program is left on idle for too long?**
    1. The program should log the user out if it is left on Idle for too long. This will prevent access to confidential information.
7. **What is most important to you about the product?**
    1. The most important thing about this product is making sure that the director can add/remove employees, adjust timecards, and generate the reports that the Director needs to submit. Other features such as a calendar will also be very beneficial for planning events. Also, the employees must be able to view their schedules when they login to use the program.
8. **How would you judge whether the product is a success?**
    1. The product will be considered a success if the Director or any of the other stakeholders are able to carry out the most important features mentioned in the question before this. If the Director cannot generate the necessary reports or

manage employee information/timecards, then it will greatly hinder the work productivity in the office for the human resource Director.

9. **How should the product be different from the way things are done now?**
    1. The software that is currently in use does not generate reports that are necessary for the daily/weekly business operations. Also, the Director currently cannot input the times that employee's clock in or out for lunches, this must be done physically. Both of these features can be included in the software to improve productivity. The current software also does not have a calendar feature, including a feature like the one mentioned in the mockup will greatly assist in planning for the week for any upcoming business events, along with viewing when employees are scheduled to work more conveniently.
10. **What policies must the product conform to?**
    1. The product must conform to any of the policies mentioned in the employee handbook. The software must also conform to any security laws for data and information such as HIPPA when dealing with things like employee benefits and company insurances if they are included in the software.

Scenarios are another technique that was used to elicit requirements in my process. I created scenarios in which the director or employee was using the software to do things such as generate reports, view a calendar to plan events or view work schedules, manage timecards, and manage payroll. This assisted me in forming the right requirements that are complete and unambiguous. The prototype technique is another one that was used in my process which was used to clarify some of the requirements that I engineered for the system. For example, when the mock-up was shown to Abby, it was during the showing of the prototype that it was made clear that the functionality of assigning training to employees should be done from the employees' tab in the dashboard. It was also during the utilization of this technique where I was informed that the employee should also have a calendar view for viewing an upcoming schedule.

Observations were done by immersing myself in the environment of the organization, by analyzing all the processes that are carried out in a business day. It was during these observations that I was able to discover all the reports that must be included in the software that I am developing. It took up a lot of time, so it is relatively expensive, however, it allowed me to capture some important details in how things should be carried out in the proposed software. User Stories is another technique that was used in my elicitation process, it really helped in visualizing the requirements that needed to be formed to be meet the standard/condition of the user story. They gave me the right amount of information that was needed to elicit requirements that would result in the right product/solution being built for the client. Finally, analyzing competitor products was another technique I used and am currently using in the elicitation process. Analyzing the competitors' product allowed me to stay focused on the features that needed to be improved upon in the software that will be developed. The competitor program has some things that were done correctly, such as the ability to adjust pay rate for example which was analyzed to be implemented in SwiftClock as well, since it is an employee timecard/payroll management system also.