Requirement Engineering Best Practices

Simon Codrington III

Grand Canyon University

SWE-520: Advanced Software Engineering Fundamentals

Dr. Wibbenmeyer

April 10, 2024

The purpose of this section is to take the following requirement and state what parts are ambiguous. The requirement is as follows, “An automated ticket machine sells rail tickets. Users select their destination and input a credit card and a personal identification number. The rail ticket is issued, and their credit card account charged. When the user presses the start button, a menu display of potential destinations is activated, along with a message to the user to select a destination and the type of ticket required. Once a destination has been selected, the ticket price is displayed, and customers are asked to input their credit card. Its validity is checked, and the user is then asked to input his or her personal identifier (PIN). When the credit transaction has been validated, the ticket is issued.” (Sommerville, 2015, p.122)

The first part that is ambiguous would be the line that states, “the rail ticket is issued”. There is absolutely no description about what is a rail ticket and what is expected to be on the rail ticket. Some of the information might be self-explanatory, name, date, destination, departure, and arrival times. But what about customer service numbers? Do these tickets need anything scanned? Will these tickets be used in a different system and therefore do we need to check people in? The next ambiguous line is the “type of ticket required”. There is no explanation of the different types of tickets. Also, there is no reference to how the physical system operates. Are these tickets printed? What kind of interface will the client use? Are these touch screens or physical buttons? How are we dispensing tickets? Are they printed or will an email suffice? None of these questions are answered and need to be to make this story clearer.

The next section is used to describe a set of use cases for an ATM machine. All uses cases have actors (the person or system interacting with system) (Sommerville, 2015, p.112) and interactions. The first will be used to describe a customer making a cash withdrawal. “The customer shall upon prompt enter their debit card into a card reader and then enter their PIN number on the touch screen. They will then be displayed with their account balance and prompted to enter the desired amount. The customer will then be prompted with an are you sure message and if the amount requested is equal or less than the balance, the money will be dispensed.” The next will be for a customer to deposit cash. “The customer upon selection from menu on touch screen shall be shown a list of all accounts associated with the card that was inserted into the card reader. The customer will select the account and the system will then open the cash collector for the user to place money wished to be deposited. The system will count and display the amount collected to the customer for verification. If a customer agrees to the validity of the count, they will be asked if they would want to complete the transaction. If not, the system will dispense the money that was deposited. Otherwise, the system will update the account balance and display the new balance to the customer.” The final use case I will discuss the use case of owner unlock safe through system. “The owner upon insert key card that is associated with physical ATM shall be shown a different option on menu touch screen from customers called “unlock ATM”. Upon selection the owner will be prompted to enter owner personal identification number and then prompted to confirm. After proper validation and confermation, the system will unlock the ATM lock that is holding the cash inside. There are a plenty other amount of usecases that could be chosen for an ATM to help with requirements such as a customer to transfer money, a customer to check account balances, and an owner to see available cash inside.

The final section is devoted to the process management of making an emergency change to the system before requirement changes are approved and how to keep the requirement documentation consistent. This is a crucial problem to be addressed and the Spiral model is good to deal with these changes (Sommerville, 2015, p.98). Using this model, first, using a tool like Azure DevOps, we can flag this emergency change and give it an ID and add it to the traceability matrices(Sommerville, 2015, p.98). The next step is risk analysis. We want to know how urgent and critical the emergency is. This is documented and shared and data feedback is gathered. Once deemed emergency, a quick requirement elicitation is conducted with a focus on what system changes need to occur. Next a prototype(Sommerville, 2015, p.98), which can be crude and quick, is used focus on implementation of crucial changes to system. Evaluating the prototype with stakeholder is next and data is gathered. This is helpful is assuring that there is no discrepancies with the old requirements and the emergency changes. Based on feedback, we further refinements and changes are implemented and risk is further assessed. If conflicts arose with older requirements, adjustments are made and documented. After this process, the new requirements are finalized and implementation can begin. After implementation the final steps are to test the new system changes and deploy. Monitoring the system and gaining feedback is the final step.

*Citations*

OpenAI. (2024, March 26). GPT-3.5 (or ChatGPT). https://www.openai.com/

Sommerville, I (2015). *Software engineering* (10th ed.). Pearson.