Jose Grijalva

9/9/2024

SER415 Fall A 2024

Reflection #2: Use Cases

1) After looking through the Use Case 1 description, many use cases were easy to identify. Because placing an order, verifying stock, and approving restricted products are all basic elements of an online ordering and warehouse system, they felt simple to do. It was difficult for me to imagine how each actor—the consumer, the warehouse manager, the delivery guy, and the Level 10 manager would communicate with the system, though. Understanding how each actor's involvement affects the workflow was necessary to determine which use cases are most relevant. Particular consideration was needed for the restricted item approval process because it contains a rare but important system component.
2) Handling the temperature shutdown feature and the penguin proximity check were challenging in Use Case 2. Because of the differences between the two circumstances, it is important to carefully evaluate how the robot would perform in the provided setting. It was difficult to specify the robot's behavior in regard to the researcher because it introduced a conditional flow that modifies the robot's actions, particularly when a penguin is present. The temperature threshold feature also caused me to pause on deeply considering since I had to clarify, without going into much technical detail, how the robot

should respond to environmental data and how it should relay this information back to the researcher in an emergency.

3) I had enough basic understanding of Use Case 3 to comprehend how customers place orders and communicate with the kitchen. But I had to think about a few more details that might affect the use case. For example, it wasn't apparent if there was a procedure for specific requests (such as allergies) or if customers were able to modify their purchases after submitting them. I additionally wanted to ask the customer about the system's handling of out-of-stock items and system faults that occur during order submission. By making these elements clear, the use case may be improved, and edge cases and system failure areas can be taken into consideration.
4) In the work field of software development, use cases are very useful, especially when outlining the interactions that actors have with the system. By making the developer consider certain workflows and the ways in which various roles access the system, they help to avoid misunderstandings during the development process. One disadvantage, however, is that use cases sometimes get too specific, emphasizing parts that might not be important to the system as a whole. This could end in over-complication or missing chances to consider the system design from more perspectives. Another disadvantage is that non-functional requirements like performance and scalability, which are just as crucial to a system's success, are sometimes not covered by use cases alone. I think there are usually more advantages to clearly defined use case development than disadvantages, even with these possible disadvantages.