Validation Report

Explain in general, how, why, and when you validate requirements in general.

Requirements ***validation and verification*** are one of the four major components of requirements development, the other three being elicitation, analysis, and specification. It is done to reveal ambiguities and errors in the requirements before any code for the system is written. Throughout the lifecycle of the product, performing requirements validation will save money and time by revealing things that could extend the delivery time. ***R equirement validation*** asses whether you have written the right requirements that define the right system as intended by the stakeholders **(doin g the right thing)**. This is different but also related to ***requirement verification***, which is about ensuring that requirements are defined clearly and precisely to the extent that they can be designed, developed, and tested **(doing the thing right).** Requirement validation on the other hand, isn’t a single discrete phase that you perform at a certain point after elicitation and documenting. Instead, some validation activities such as incremental reviews, it is threaded throughout the iterative elicitation, analysis, and specifications processes. There are many things that requirement validation attempts to ensure. Validating requirements allows teams to build the correct solution that meets the objectives of the business. They ensure the requirements accurately describe the intended system capabilities and properties that will satisfy the stakeholders' needs. Validation also ensures that the software requirements are correctly derived from the business requirements, system requirements, business rules, and other sources. Furthermore, it attempts to ensure that the requirements for the product are complete, feasible, and verifiable along with making sure that all the requirements are necessary, and that the entire set of requirements is sufficient to meet the objective of the business. Also, validating requirements attempt to ensure that the requirements representations are consistent with each other and that the requirements provide an adequate basis to proceed with design and construction. Validating requirements will ensure that the requirements are complete, correct, feasible, necessary, prioritized, unambiguous, and verifiable. **Ways to validate requirements include <u>requirements reviews and inspections, prototyping, and performing acceptance tests</u>.**

Requirements reviews and inspection

The most common form of requirements validation is by reviewing and/or inspecting the **requirements document(s).** In this means of validation, a group of reviewers is assigned a brief to look for errors, mistaken assumptions, lack of clarity, and deviation from standard practices. The group composition that performs the review and inspection is important, at least one representative of the customer should be included for a customer-driven project. The requirements review may be carried out once the system definition document, the system specification document, the software requirements specification document (SRS), the baseline specification for a new release, or any other steps in the process are complete.

Prototyping to validate requirements

Another common means for validating requirements in software engineering is prototyping. Prototyping can also be used to elicit new requirements. A huge advantage of prototypes is that they make it easier to interpret the software engineer’s assumptions and, it is here where they will receive feedback if they are wrong with how the requirements are being displayed. For example, a user interface of a program can be better understood with the use of a prototype. Requirements can be very volatile, however, after the use of prototypes, the volatility of the requirements becomes extremely low since there is an agreement between the stakeholder and the software engineer on how the product is being displayed along with its requirements. There is a disadvantage, one of them is that the user may become distracted from the core functionality of the product by the cosmetic issues or problems of the prototype. It is for this reason that prototypes such as flip-chart based mockups are used. Usually, prototypes may be expensive, but the benefits usually outweigh the costs. It is possible for prototypes to be evolutionary as well, as they can contain aspects of the final product.

Acceptance Tests

Software requirements should be able to validate that the finished product satisfies them all, so it is very important to plan how to verify each requirement. Designing acceptance tests does this in most cases. An acceptance test is essentially a “check” for whether a requirement is met or not. It specifies how your client will verify that a requirement has been satisfied. Acceptance tests are based on the acceptance criteria, which is a specific condition that needs to be met. If the acceptance tests are passed for all requirements acceptance criteria, then the requirement is successfully acceptance tested.

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

The requirements engineering process consists of requirements elicitation, specification, analysis, and verification and validation. These four components cannot be separated and be performed sequentially, instead, they are performed repeatedly all throughout the software development life cycle (SDLC). Identifying the stakeholders for the system that is to be developed is very important as this a huge source of requirements that the requirements engineer will elicit from. These requirements must be validated in order to ensure that the software has been built in such a way that it is traceable to the stakeholder’s requirements, in other words, to ensure that the right product has been built. The specifications and analysis of these requirements all help with validating the requirements by providing the requirements documentation and models.

Requirements Memorandum

DATE: April 25th, 2020

TO: Ecological Laboratories Inc

FROM: Stephen Aranda

SUBJECT: SwiftClock Requirements Validation

To summarize my experiences with conducting SwiftClocks' requirements review and inspection, this memo will be written for future uses. The requirements for SwiftClock were elicited from the major stakeholder, Abby Richter. Multiple **interviews** were conducted with Abby in order to form user stories that were used to form requirements for the system. **Scenarios, prototypes, and competitor products** were other techniques utilized to <u>elicit requirements.</u> After elicitation, the System/Software Requirements Specifications document was able to be created. Then, I sent the document to Abby Richter one of the stakeholders to give her a chance to **review/inspect** the document and SwiftClock's requirements to validate them. Abby made sure to look for errors, mistaken assumptions, lack of clarity, and deviation from standard practices and let me know of any changes that needed to be made anywhere. Another form of validating the requirements to ensure that the right product was set to be made was sharing **SwiftClock's prototype** with the Abby, where she informed me of any changes that needed to be made as well. Lastly, **acceptance tests** were created for each requirement to specify how the requirements will be verified. After inspection of the requirements and revisions were made, the complete list of validated requirements was shared with the rest of the stakeholders for them to see.