Activity 2: Software Product Requirements

Why?

We cannot build a software product until we know what it is we are supposed to build. Product requirements provide such specifications.

Learning Objectives

- Understand what stakeholder needs are, what product requirements are, and how the two are related.
- Understand levels of abstraction in needs and requirements.
- Understand how requirements are handled in traditional and agile processes.
- Understand how requirements are stated in traditional and agile processes.
- Be familiar with standard needs elicitation techniques.
- Understand requirements verification and validation techniques.
- Understand requirements management in traditional and agile processes.
- Understand the relationship between product design and requirements analysis.

Success Criteria

- Be able to distinguish between needs and requirements.
- Be able to provide examples of needs or requirements at different levels of abstraction.
- Be able to explain how requirements are frozen and recorded in software requirements specification documents in traditional processes.
- Be able to explain how in Scrum needs are captured in product backlogs and become requirements in sprint backlogs.
- Be able to write traditional requirements specifications that are clear, testable, atomic, and have a unique identifier.
- Be able to write user stories in user voice form.
- Be able to list and explain several needs elicitation techniques.
- Be able to explain how requirements are validated and verified in traditional and agile processes.
- Be able to explain requirements management in traditional and Scrum processes.
- Explain the relationship between product design and requirements analysis.

Resources

Software Product Requirements (pdf from Canvas)

Exercises

1. What is a stakeholder?

2. What is the difference between functional and non-functional requirements?

3. What is a PBI?

4. What is an example of a needs elicitation technique?

5. What is an SRS?

Problem

1. Name three reasons that stakeholder needs cannot simply be recorded as software product requirements.

2. Name three reasons why freezing requirements as is done in traditional (waterfall) processes does not work well.

3. Name three ways that Scrum makes requirements change as easy and cheap as possible.

4. What is the difference between epics, features, and sprintable stories?

5. How can needs elicitation go wrong in Scrum?

6. How are requirements validated and verified in Scrum?

7. What is the relationship between software product design and software requirements analysis?