Requirements are needed because they outline the wants and needs of the stakeholders for the outcome of their desired software product. Defining requirements can be challenging, especially due to communication gaps between technical staff and stakeholders. Requirements engineering bridges this gap by ensuring that stakeholder needs are properly understood and communicated. Requirements engineering is the systematic process around gathering, defining, and refining requirements from stakeholders for a system or software product. Requirements can be categorized into two groups: Functional and Non-Functional. Functional requirements specify the system's behaviors and actions, like role-based access controls. Non-functional requirements are quality attributes that can be assigned to the new software system. Non-functional requirements are often referred to as the "-ilities". Some examples would be maintainability, scalability, reliability, testability, etc.

Once requirements are gathered, it is crucial to document them effectively. There are many qualities that make a good requirements document. One of the first tenets of a successful requirements document is understanding user need, or user story. To do so, the "three W's" must be defined. For *whose* benefit is this requirement generated? *What* needs to be done to accomplish this requirement? *Why* is this requirement needed? All of these questions help further define the requirement, provide clarity, and remove ambiguity. Requirements should also be prioritized so that critical requirements can be emphasized first. Finally, requirements should be separate from design. The definition of the software product's utility should be distinct from its implementation. This helps reduce scope creep and allows the design to be up to the software engineer fulfilling the development of the requirement (Deepanshu Natani, 2020).

Gathering and producing high quality requirements is not a simple task. There are many potential challenges that can arise while generating requirements. One of them was mentioned in the beginning of this document, which is communication between the technical staff and the stakeholder. There are many ways to mitigate this complication, the first being separating the technical staff from the stakeholders altogether. Many times, there is a role in between the stakeholder and the engineer called a "Product Owner". The product owner may have some technical knowledge but is primarily an expert in the domain of the product being developed and in software requirements engineering and scrum/agile techniques. This individual is responsible for compiling these requirements from the stakeholders so that the engineers don't risk creating solutions while requirements are gathered (keeping requirements and design separated).

Another potential complication during the requirements engineering process is evolving requirements. Sometimes stakeholders may request additional features or make

changes during development. In software development this is often called “scope creep”. To mitigate the impact of scope creep, there needs to be a change management process, or change control plan. It’s unrealistic to think that requirements won’t change during development of the product, but this change needs to be managed and approved. Without this approval process, deadlines and budgets could be at risk (Westland, 2022).

Deepanshu Natani. (2020, September 15). The art of writing good requirements. Atlassian Community. https://community.atlassian.com/t5/Jira-articles/The-art-of-writing-good-requirements/ba-p/1482103

Westland, J. (2022, February 14). What is change control in project management? ProjectManager.com. https://www.projectmanager.com/blog/what-is-change-control-in-project-management