Requirements Engineering

Requirements engineering is part of agile methodology and is the process of determining user expectations for a new or modified product. These features, called requirements, must be quantifiable, relevant and detailed and are learned before building the product. The discovered requirements must be recorded so that they can be agreed upon, prioritised, estimated, and eventually developed, tested and delivered. A good requirement should address the system functionality and satisfy the user’s needs (Requirements analysis (requirements engineering) 2016).

Requirements engineering is usually undertaken in the beginning of the system development lifecycle. Yet in some cases, developing a precise set of requirements that would remain stable throughout the months or years of development has been realised to be unmanageable in practice. As a result requirements engineering is an incremental and iterative process as per Agile and performed to correspond with other system improvement activities such as design and coding (Pandey, et al 2010).

There are two kinds of requirements; functional and non-functional. A functional requirement is a requirement that stipulates an action accomplished by the system without considering physical limitations. A Non-functional requirement is a requirement that specifies system properties such as environmental and implementation constraints, performance, platform dependencies, maintainability, extensibility, reliability (Pandey, et al 2010).

There are four phases in requirements engineering:

**Requirement elicitation and development**

The main aim of this phase is to investigate and collect preferred requirements and objectives for the system from different perspectives (e.g., customer, users, constraints, system's operating environment, trade, marketing and standard etc.). This phase commences with identifying stakeholders of the system and accumulating raw requirements. These are requirements that have not been evaluated and recorded. In a well-formed requirement representation (Pandey, et al 2010).

**Documentation of requirements**

In this phase an official document is prepared after gathering requirements. This comprises of a comprehensive explanation of the external behaviour of the software system. This phase allows for the determining of which functions of the system will be executed by software. Here Non-functional requirements and functional requirements are addressed and implemented into the software requirements specification using flow-down, allocation, and derivation (Pandey, et al 2010).

**Validation and verification of requirements**

In this phase it is determined that the proper requirements are specified (validation) and these requirements are specified correctly (verification). Validation and verification activities include validating the system requirements against raw requirements and verifying the correctness of system requirement documentation. Software requirements need to be validated and verified against system level requirements. Typical procedures for validating requirements are requirements assessments with the stakeholders, and prototyping. Typical procedures for verification comprise of correctness, consistency, unambiguousness and understand ability of requirements (Pandey, et al 2010).

**Requirement management and planning**

This phase facilitates the control and tracking of the fluctuations of the agreed requirements, associations amongst requirements, and assurances between the requirements documents and other documents produced throughout the requirements engineering process. It is a continuous phase and regularly carries out its activities of identification and change control during and after requirements management and planning phase. Requirements management is an unceasing action that can perform after development and during maintenance because requirements may carry on being modified (Pandey, et al 2010).

**Planning poker**

Planning Poker is the secure, fun way for agile teams to guide sprint planning and build accurate consensus estimates. It is used to estimate effort or relative size of [development goals](https://en.wikipedia.org/wiki/User_stories) such as requirements engineering in [software development](https://en.wikipedia.org/wiki/Software_development).

To start a poker planning session, the product owner or customer reads an agile user story or describes a feature to the estimators.

Each estimator is holding a deck of Planning Poker cards with values like 0, 1, 2, 3, 5, 8, 13, 20, 40 and 100. The values represent the number of story points, ideal days, or other units in which the team has agreed upon

The estimators discuss the feature, asking questions of the product owner as needed. When the feature has been fully discussed, each estimator privately selects one card to represent his/her estimate. All cards are then revealed at the same time.

If all estimators selected the same value, that becomes the estimate. If not, the estimators discuss their estimates. The high and low estimators should especially share their reasons. After further discussion, each estimator reselects an estimate card, and all cards are again revealed at the same time.

The poker planning process is repeated until consensus is achieved or until the estimators decide that agile estimating and planning of a particular item needs to be deferred until additional information can be acquired (Planning Poker 2016).

**Links for upskilling**

<http://ieeexplore.ieee.org/xpl/login.jsp?tp=&arnumber=5656776&url=http%3A%2F%2Fieeexplore.ieee.org%2Fxpls%2Fabs_all.jsp%3Farnumber%3D5656776>.

**References**

Pandey, D., Suman, U., & Ramani, A. K. (2010, October). An effective requirement engineering process model for software development and requirements management. In *Advances in Recent Technologies in Communication and Computing (ARTCom), 2010 International Conference on*(pp. 287-291). IEEE.

Planning Poker. (2016). Retrieved from <https://www.mountaingoatsoftware.com/agile/planning-poker>

Requirements analysis (requirements engineering)(2016). Retrieved from <http://searchsoftwarequality.techtarget.com/definition/requirements-analysis>