Software Engineering Definitions

1. **Software Engineering** – the use of tools, techniques, procedures, and paradigms to enhance the quality of software products.
2. **Software** – A computer program that directs the operation of a computer to accomplish a specific task.
3. **Enterprise** **Software** – Software that is built for a specific business process
4. **Behavior** – Something the software does (function, use case, etc)
5. **Objects** – Data used by the activities (records, classes, fields, etc)
6. **Relationships** – Which objects are used in which activities
7. **Abstraction** – A simplification that allows focus on some pertinent aspect of the software (model, blueprint of a home showing electrical wiring)
8. **Modularization** – Divide and conquer; break entire behavior into independent, logical parts. (design/build/test/integrate)
9. **Activity** – A type of work performed during software development
10. **Stakeholders** – Anyone who benefits from the software being produced (i.e., has a stake in the success of the software; clients, investors, developers, end-user)
11. **Software Engineering Process** – A series of steps involving activities, constraints, resources that produce an intended output.
12. **Constraint** – A restriction imposed on the process (deadlines, a schedule, finances, target platform)
13. **Requirements** – What the system is supposed to do in terms of behavior, data, and constraints.
14. **Specifications** – Developer interpretation of what the system is supposed to do.
15. **Design** – Blueprints of how the system will be built
16. **Implementation** – The working software product
17. **Process Model** – A specific configuration of process activities that can guide real software development.
18. **Risk** – The chance that something bad will happen. An activities work product is not 100% correct.
19. **Validation** – Check if all requirements are accounted for. “Is this what the customer wants?”
20. **Verification** – Everything works correctly according to the specifications.
21. **Agility** – A software engineering characteristic that acknowledges change as a necessary part of software engineering and promotes efficient, appropriate response to change.
    1. Based on iterative software engineering process model
       1. (Iteration = Sprint)
22. **Requirement** – In general, an expression of desired behavior. Something the system should or should not do
23. **Requirements** **Engineer** **(RA)** – Developer who collects and analyzes requirements, produces and validations specifications
24. **Specification** – Precise description of what software is supposed to do
25. **Entity** – A very general term for something that interacts with the system or is acted upon