Print

Glossary

This is a Glossary for defining Week 1 terms.

1. **Software Engineering**: Systematic application of engineering approaches to the development of software.
2. **Software Development**: The process of conceiving, specifying, designing, programming, documenting, testing, and bug fixing involved in creating and maintaining applications, frameworks, and other software components.
3. **Continuous Integration:** The practice of merging all developer working copies to a shared mainline several times a day.
4. **Continuous Deployment:** A software release method that automatically, and frequently, releases any code commit that passes automated testing into the production environment.
5. **Rapid application development:** a software development methodology, which favors iterative development and the rapid construction of prototypes instead of large amounts of up-front planning. The "planning" of software developed using RAD is interleaved with writing the software itself. The lack of extensive pre-planning generally allows software to be written much faster, and makes it easier to change requirements. The rapid development process starts with the development of preliminary data models and business process models using structured techniques. In the next stage, requirements are verified using prototyping, eventually to refine the data and process models. These stages are repeated iteratively; further development results in "a combined business requirements and technical design statement to be used for constructing new systems".
6. **Waterfall:** The waterfall model is a sequential development approach, in which development is seen as flowing steadily downwards (like a waterfall) through several phases, typically: Requirements analysis resulting in a software requirements specification, Software design, Implementation, Testing, Integration, if there are multiple subsystems, Deployment (or Installation), Maintenance
7. **Agile:** "Agile software development" refers to a group of software development frameworks based on iterative development, where requirements and solutions evolve via collaboration between self-organizing cross-functional teams. Agile processes fundamentally incorporate iteration and the continuous feedback that it provides to successively refine and deliver a software system. Agile model also include following software development processes: Dynamic systems development method (DSDM), Kanban, Scrum, Crystal, Atern, Lean software development
8. **Scrum:** An agile process that helps to deliver the business value in the shortest time. It rapidly and repeatedly inspects actual working software. It emphasizes on teamwork and iterative progress of the software. Its goal is to deliver new software every 2-4 weeks.
9. **Kanban:** A visual system for managing work. It visualizes both the process and the actual work passing through that process. The main objective of implementing Kanban is to identify potential bottlenecks in the process and fix them. Kanban goal is that work flow should proceed smoothly at an optimal speed.

10. **Software Development Lifecycle (SDLC):** A well-defined sequence of steps or phases in software engineering to develop the intended product. Phases include requirements, design, development and coding, Integration and testing, implementation and deployment, review.
11. **Dev-Ops aka Development and Operations:** a process framework that ensures collaboration between development and operations teams to deploy code to production environments faster in a repeatable and automated way. Principles include frequent releases, team awareness and shared goals, end to end responsibilities, continuous improvement, and automation. DevOps lifecycle begin at Dev which includes Plan, code, build and test; the Ops include release, deploy, operate, monitor.
12. **CI/CD pipeline aka continuous integration/continuous delivery or deployment:** A series of steps for integration and delivery/deployment, also includes software build (compilations) process, testing and more.
13. **Continuous Planning:** Have a plan in place that can be quickly and frequently reprioritized and adjusted.
14. **Continuous Development:** Small development cycles, iterative development.
15. **Continuous Testing:** Tests the completed build from the previous stage.
16. **Continuous Release:** Product is made ready for release to deployment.
17. **Continuous Deployment:** Software deployed to production environment, available to end-users.
18. **Continuous Operating:** Ensure software runs smoothly for end-users, implement continuous feedback for use in next iteration.
19. **Continuous Monitoring:** Monitor the deployed software for issues.
20. **Permissive Licenses**: Provide software as is, with no warranties; use and alter as you wish, at your own risk. Ex.// Berkeley Software Distribution (BSD), MIT, Apache 2
21. **Copyleft Licenses:** Extra requirements to the permissive license; source code must be included in distribution of binaries. Ex.// GPL, MPL, EPL, CDDL
22. **TDD (Test Driven Development):** Software development method focusing on an iterative development cycle where the emphasis is placed on writing test cases before the actual feature or function is written - currently how workshops deliver assignments