Summary

Brooke Lampe

Scrum, Scrumban, and Kanban are all flavors of Agile software development. As such, they divide work into small pieces, produce deliverable software throughout the development process, and emphasize client involvement. Scrum produces a deliverable at the end of each Sprint. A Sprint is a fixed-time iteration of development. It is pre-planned, and all work planned for the Sprint should be completed at the conclusion of the Sprint. Work items should be estimated. In contrast, Scrumban and Kanban expect a continuous flow of work: As items progress through the workflow, new items are added as needed. Scrumban and Kanban also place limits on work in progress. Meanwhile, in Scrum, work in progress is controlled by Sprint content. Scrum provides more structure and requires more planning overhead, while Kanban provides smoother flow. Scrumban offers a middle ground between structure and smoothness, and different implementations of Scrumban can lean slightly more toward Scrum or Kanban. Each process expects regular meetings to coordinate team activities. In addition, Scrum expects a meeting at the end of each Sprint to plan the next Sprint. Scrum and Scrumban expect to include review and retrospection in regular meetings. Kanban and Scrumban expect regular meetings to replenish the work items to be selected next. All flavors work to optimize development. In Scrum, Sprint retrospection supports optimization. In Kanban, developers work to minimize and normalize the time to complete one work item, which is called the cycle time. Scrumban utilizes both retrospection and cycle time minimization in its optimization process.