

# Guidelines of the Scrum Development Process

by Ernie Paschall

The Agile Manifesto of 2001 made it clear that individuals and interactions are valued over processes and tools (Beck et al.). Does this mean that processes and tools are counter productive for agile teams? The answer is not simply yes or no. This is one of those classic gray areas of software development that is based on empirical studies. It turns out that processes and tools are not bad if they are simple and helpful for all stakeholders of the software being created. In fact, when both of these factors are true, then iterations and releases become easier to manage.

Since a customer is a very important stakeholder, it stands to reason that she be involved in the product definition, the product planning, and the product development. This collaboration with the customer allows her to identify herself as a team member involved in producing the software. Therefore, the customer becomes aware that the team will succeed collectively or the team will fail collectively (Cohn 2005). This is a very positive side effect of agile development methodologies from the point of view of the software development organization.

The Agile Manifesto of 2001 also indicates that responding to change is more valuable than following a plan (Beck et al.). Therefore, if a plan has been created and conditions change after its creation, then the plan should be allowed to change in order to compensate. Many software developers have interpreted this statement in the manifesto to mean that a plan is not necessary in agile development. Having no plan is analogous to another common development methodology known as the “Chaos Methodology”. In fact, planning is very important to agile development methodologies. There are three levels of planning: release planning, iteration planning, and daily planning (Cohn 2005). With these three levels of planning, agile software development plans change in step with the changing conditions of satisfaction of the customer. The level of detail gets more precise from the release plan down to the daily plan.

The purpose of Scrum Time is to help agile development teams and customers collaborate in the production of software. Scrumtime.org seeks to stay true to the intentions of the Agile Manifesto authors by using the Manifesto and other published works as the requirements of the Scrum Time application.

Scrum Time 1.0 will have four main features: Define, Plan, Execute, and Monitor. The details of these features may be found below. If you find that this implementation of the Scrum process works for you, then you will benefit by using Scrum Time for your development projects.

## Define

The application user stories must be created and estimated.

- Brainstorming – identify the user stories or features of the application.
  - “As [a user], I want to [perform some task] so that I can [reach some goal].”

- Estimate the magnitude of each story
  - Story points (Cohn 2005)
  - Ideal Days (Cohn 2005) [Post 1.0 Release](#)
  - Planning Poker (Grenning 2002) [Post 1.0 Release](#)
- Identify Themes (Cohn 2005) – Themes should be identified such that each defines a discrete set of customer valued functionality. Stories are aggregated under a relevant theme. i.e.
  - User management.
    - As a user, I need to register so that I may gain access to the system.
    - As a registered user, I need to login so that I may access my system information.

## Plan

- Prioritize Themes (Cohn 2005)
  - Factors – These factors are mentioned as notes in the margins of Scrum Time.
    - Financial value
    - Development and support cost
    - Amount and significance of learning and new knowledge
    - Amount of risk removed by creating the themes
- Prioritize Desirability (Cohn 2005)
  - Relative weighting (Wiegers 1999)
    - relies on expert judgment instead of questionnaires
    - considers both the positive benefit of the presence of the feature and the negative impact of its absence.
  - Kano model (Griffin and Hauser 1993) [Post 1.0 Release](#)
    - allow customer to fill out online questionnaires to receive feedback on themes or stories
- Release Planning
  - Notes
    - Typically 3-9 months
    - Items are user stories
    - Estimated in user points or ideal days ([Post 1.0 Release](#))
  - Determine the conditions of satisfaction (Cohn 2005)
    - schedule
    - scope
    - resources
  - Allow re-estimation of the user stories
  - Allow re-prioritizing of the user stories
  - Define the standard sprint duration
  - Define the scrum team
    - Estimate an initial velocity in story points
    - Velocity estimation helper [Post 1.0 Release](#)

- Assign stories to the release
- Create a date for the release.
  - Auto-sprint creation - given the time between today and the first release, the system will automatically create sprints containing the prioritized user stories using the initial velocity of the scrum team. If the stories do not fit in the release, then the conditions of satisfaction have not been met.
- Sprint Planning (Velocity-Driven Planning (Cohn 2005))
  - Notes
    - Typically 2-4 weeks
    - Items are tasks
    - Estimated in ideal hours
  - Adjust story priorities
  - Determine target velocity
  - Identify a sprint goal
  - Assign user stories to the sprint
  - Break-down user stories into tasks
    - Scrum Time should not dictate who breaks tasks down
    - Scrum Time dictates that all tasks be estimated before a sprint may be started
    - Task sizes should be such at each developer is able to finish an average of one per day (Schwaber and Beedle 2002; Rising 2002)
- Additional Planning Tools
  - Planning Buffer Helper (Post 1.0 Release)
  - Scrum-of-Scrum Planning (Post 1.0 Release)

## Execute

- Release kick-off
  - an official acknowledgment of the beginning of the release cycle
- Sprint kick-off
  - an official acknowledgment of the beginning of the sprint cycle
  - this enables the daily scrum features of Scrum Time
- Daily Scrum
  - Scrum team meets daily during the sprint
  - Task board (Cohn 2005)
- Sprint completion
  - an official acknowledgment of the end of a sprint
  - Sprint review meeting
- Release completion
  - an official acknowledgment of the end of a release
  - Release review meeting

## Monitor

- Release burn down chart

- Sprint burn down chart
- Parking-Lot chart (DeLuca 2002) Post 1.0 Release
- Read-only task board