Agile Notetaker

& Scrum Reference

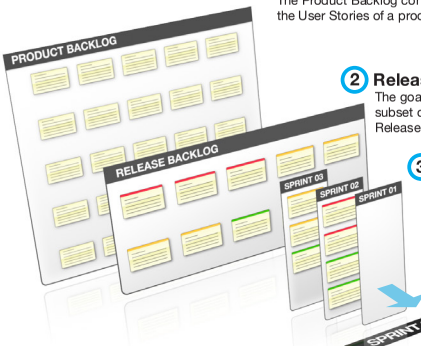
Designed by Axosoft, the creators of OnTime—the #1 selling scrum software.

axosoft

Scrum Diagram:

1 Product Backlog

The Product Backlog contains a wish list of all the User Stories of a product.

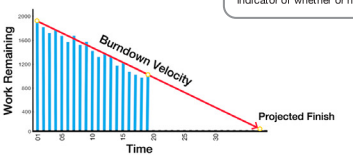


2 Release Backlog

The goal of a given release is to deliver a subset of the Product Backlog, known as the Release Backlog.

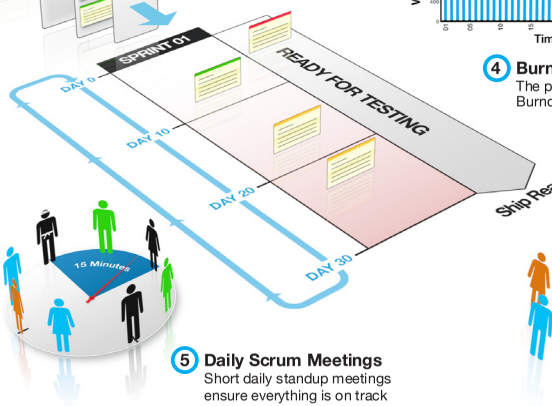
3 Sprint Backlogs

Each Sprint (or short duration milestone) takes a small chunk of the Release Backlog and gets it Ship-Ready!



4 Burndown Chart

The progress of the team is monitored using a Burndown Chart.



5 Daily Scrum Meetings

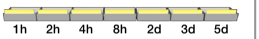
Short daily standup meetings ensure everything is on track and everyone has the tools they need.

6 Sprint Retrospective

After each sprint, a longer retrospective meeting helps fine-tune the process.

Estimation Techniques:

Hours: When estimating work, it's important to have the entire team use some standards. Use 1h, 2h, 4h, 8h, etc. No estimations in-between.



Story Points: You can also estimate work in comparison to the complexity of a well known but simple component.

Remember that in Scrum, the estimates are only part of the story. The Burndown Velocity is the true indicator of whether or not the project is on track.

Team Roles:

Product Owner: Is responsible for what goes into the product backlog and prioritizes it. Would probably make a good dictator if given the chance.

Scrum Master: A team facilitator. Ensures teams have what they need to get the job done. Also, sets up meetings and monitors everything. Also, kicks ass when necessary.

Developers & Testers: They write code and make sure it does what it's suppose to do. Duh!

For more info:

Intro to Scrum Video: www.axosoft.com/scrumvideo

Shipping software on time and on budget is tough. There are a lot of variables with rapidly moving parts. Everything, from the "what" should be delivered to the ideal technologies used to deliver it, is constantly changing. If it takes too long to ship, the product may be obsolete before it gets to market. If it ships early but with poor quality, it will have to fight an uphill battle to overcome its first impression.

To balance these tradeoffs, an iterative agile development technique has become the new standard of shipping software on time. Within the agile world, scrum has emerged as the leading methodology for building efficient self-organizing teams and monitoring the progress of those teams using proven techniques that help make adjustments along the way.

The concepts of agile and scrum are simple. In the first few pages of this agile notetaker, my team and I have broken down these concepts into manageable chunks of bite-sized knowledge. Each time you sit in a conference room waiting for a meeting to start, review a page from the concepts explained in this book. Over time, you will notice that your team will ship software faster and with better quality than ever before.

The concepts are solid and proven by thousands of dev teams worldwide. We have been using these techniques to build software at Axosoft for more than a decade now with fantastic results.

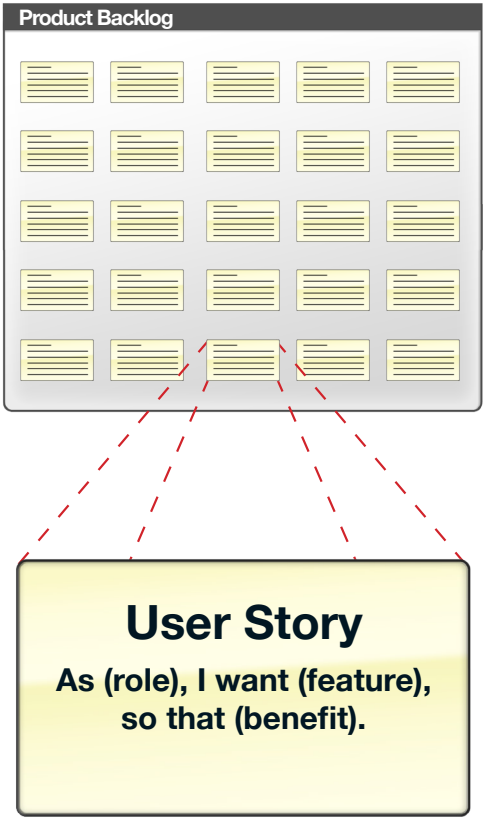
I hope you enjoy this book as much as we enjoyed creating it. As always, we love to hear from you. Feel free to email me directly with your comments about this book.

Sincerely,

Hamid Shojaee
hamids@axosoft.com
Axosoft, LLC
Building Software Solutions for Agile Businesses

1 Product Backlog

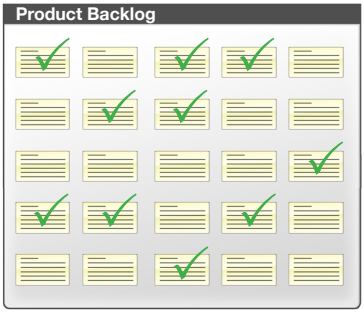
The Product Backlog contains the wish list of all the User Stories that would make the product great.



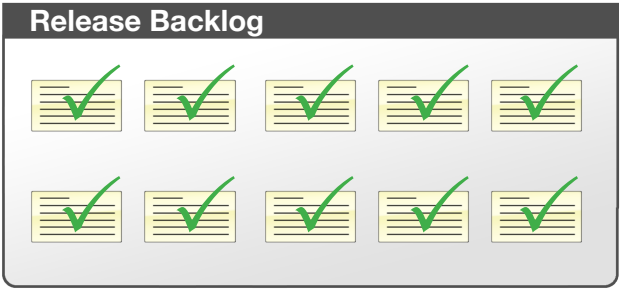
In Scrum, features are known as User Stories and they are written from the perspective of the end-user. The Product Owner, representing the users and customers of the product, decides which User Stories or items make it into the Product Backlog.

2 Release Backlog

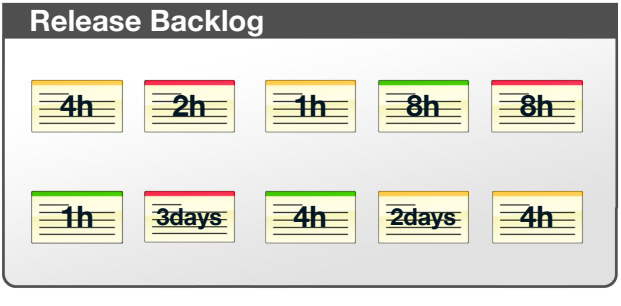
The goal of a given release is to deliver a subset of the Product Backlog, known as the Release Backlog.



After identifying which User Stories will go into a particular Release, the User Stories become part of a Release Backlog...



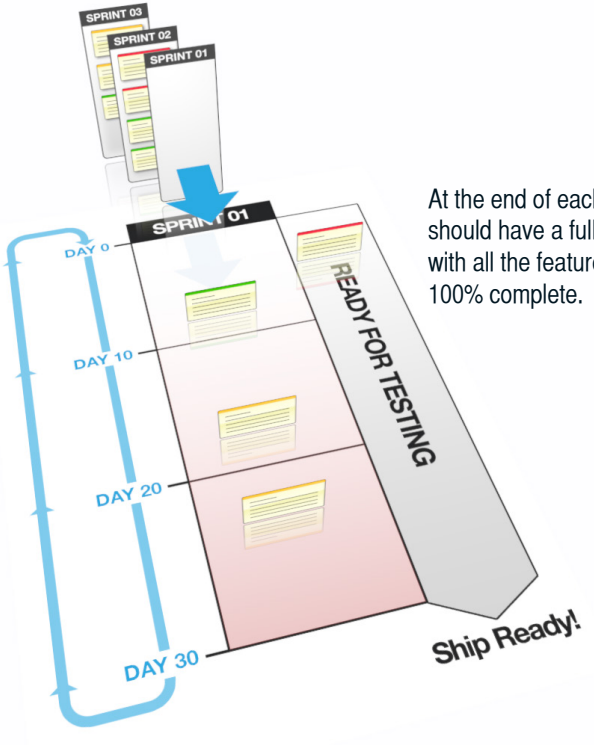
...which are then prioritized by the Development Team, who also estimate the amount of time involved to complete each item. *see Pro Tip- Estimation Techniques*



3 Sprint Backlog

Each Sprint (or short duration milestone) takes a manageable chunk of the Release Backlog and gets it to a Ship-Ready state!

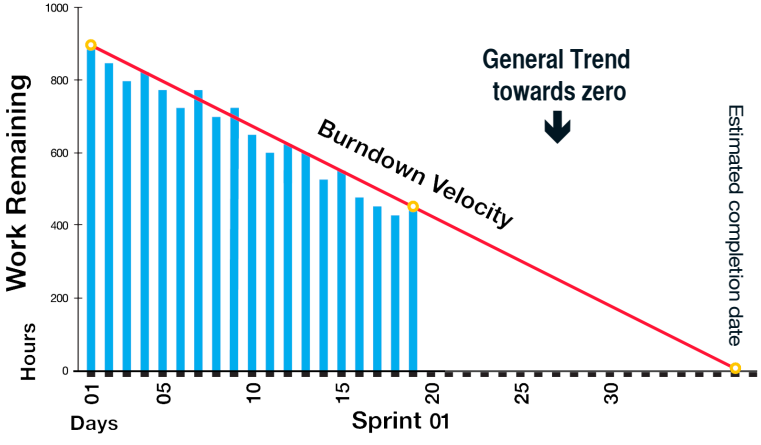
Sprints generally range from a couple of days to 30 days in length. ***Remember- The shorter the Release Cycle, the shorter each Sprint should be, with two to a dozen Sprints in a given Release.***



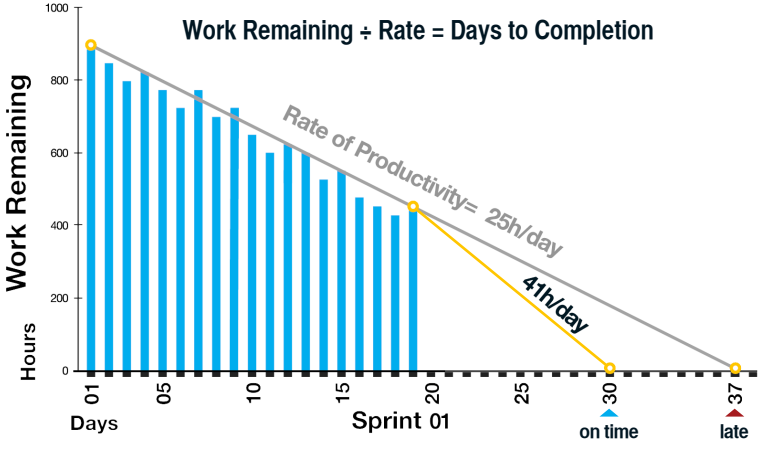
At the end of each Sprint, you should have a fully tested product with all the features of the Sprint 100% complete.

4 Burndown Charts

The progress of the team is monitored using a Burndown Chart, one of the best project visibility tools, to ensure a project is progressing smoothly. The Burndown Chart provides a day-by-day measure of the amount of work that remains in a given Sprint or Release.



The slope of the graph, or Burndown Velocity, is the average rate of productivity for each day.

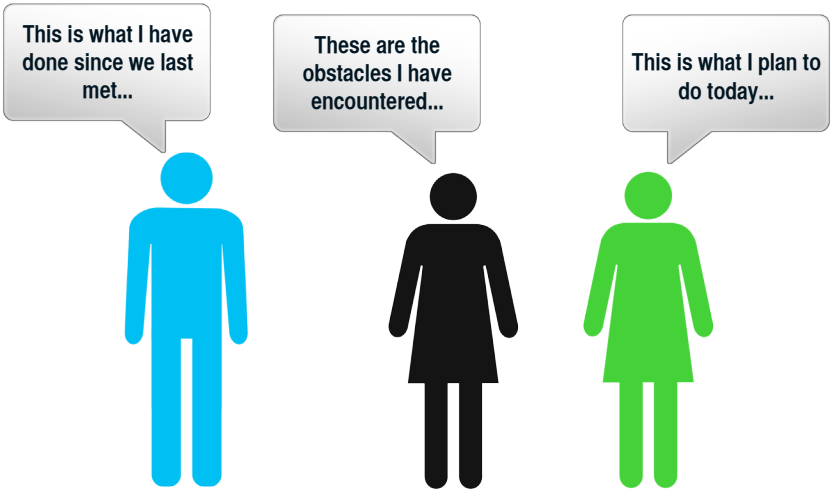


Knowing whether or not the Project is on time early in the schedule, can help teams make the proper adjustments necessary to get the project on-track.

5 Daily Standups

Short daily standup meetings (aka The Daily Scrum) ensure everything is on track and everyone has the tools they need. It is an essential tool to have information flow freely between team members.

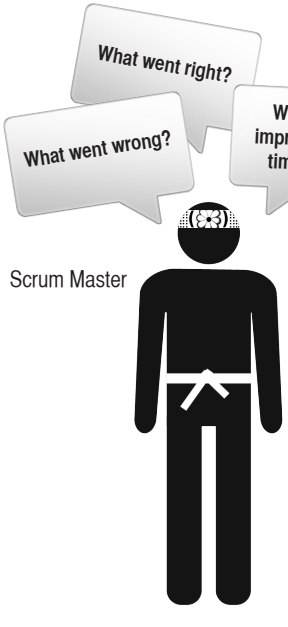
Team members list the work they have completed since the last meeting, any obstacles in their way, and what they are going to do next.



6 Sprint Retrospectives

After each Sprint, a longer Retrospective meeting helps fine-tune the process.

This is a time for each team member to reflect on what went right and areas of improvement.



After all, Scrum is a flexible Agile development method that needs constant improving and tweaking for every team.

Reference:

Team Roles

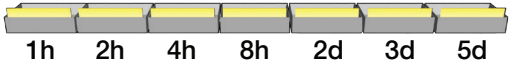
Product Owner: Is responsible for what goes into the product backlog and prioritizes it.

Scrum Master: A team facilitator. Ensures teams have what they need to get the job done. Also, sets up meetings and monitors everything.

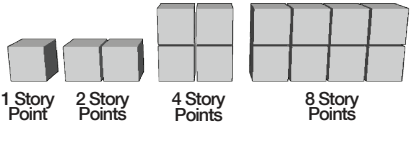
Developers & Testers: They write code and make sure it does what it's suppose to do.

Pro Tip- Estimation Techniques

While most estimates are inaccurate, there are things you can do to help improve your team's estimation techniques. For example, a simple technique is to create estimating buckets where User Stories fall into a complexity bucket such as:



or:



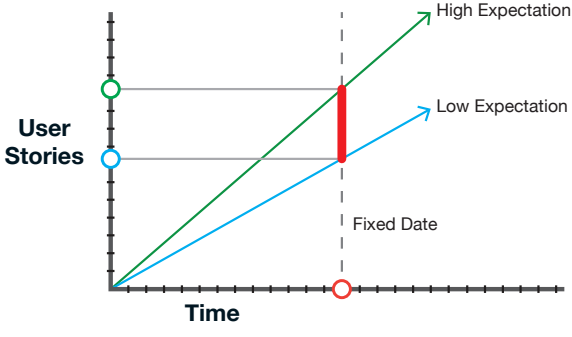
No item would be estimated at fractional time or any numbers in between the buckets. For example, a 6-hour estimate would fall into the "8-hour" bucket and therefore would have an 8-hour estimate. Regardless of whether you use Hours or Story Points for your estimations, be sure to always maintain the same unit for estimating across the board. Remember that in Scrum, the estimates are only part of the story. **The Burndown Velocity is the true indicator of whether or not the project is on track.**

Expectation Charts

One of the most important tasks of any development team is to appropriately set expectations. An important tool to help set expectations with stakeholders is to use the charts below and ask them to choose “Fixed Date Delivery” or “Fixed Scope Delivery” but that they can’t have both.

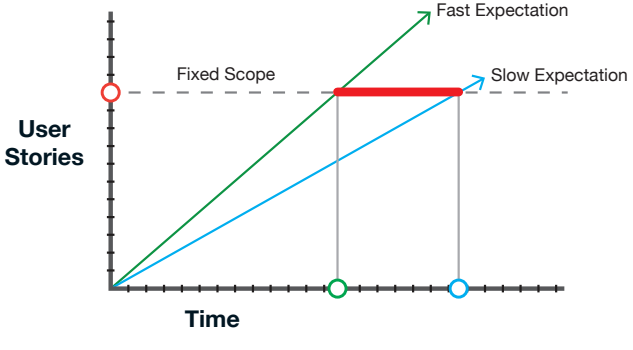
With a fixed date delivery, what can be delivered is unknown, but the team should be able to provide low expectations and high expectations estimates (a lower and upper scope boundary).

Fixed Date Delivery



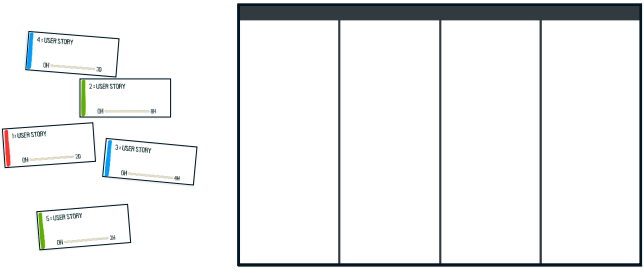
With a fixed scope delivery, when the project will be delivered is unknown, but the team should be able to provide slow expectations and fast expectations estimate (a lower and upper time boundary).

Fixed Scope Delivery

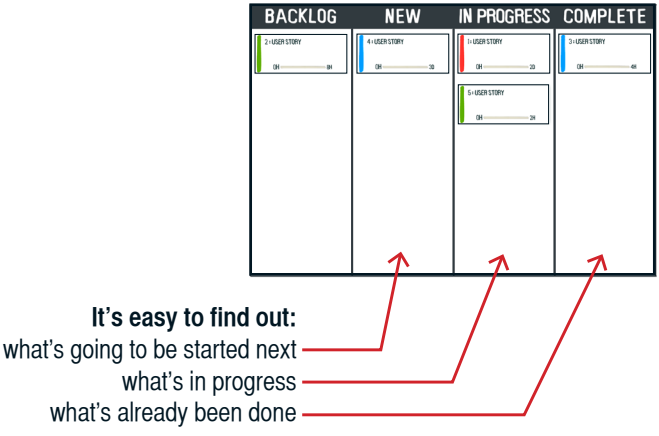


Scrum + Kanban = Agile Awesomeness

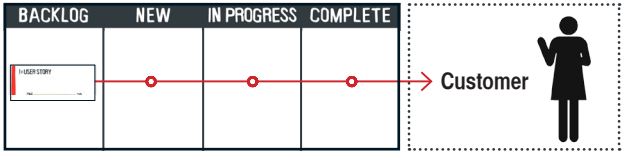
A Kanban is a lean scheduling system in manufacturing that utilizes visual cues. When adapted to software development, Kanban systems usually start with a board and visual cards that represent items in your product backlog.



On the board, you place the cards into columns that represent their current step in the workflow, ranging from “New” to “Complete”. **Pro Tip: The steps in-between are entirely up to you, so keep it simple and efficient.**

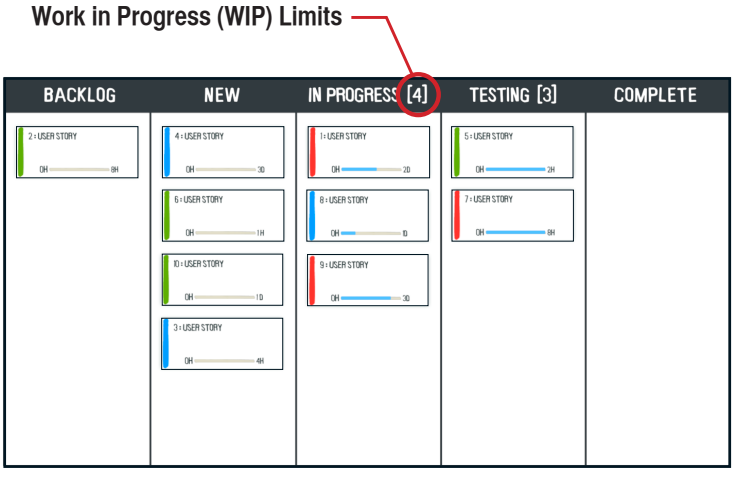


So long as your team keeps finishing work, those cards keep moving to the right, and more importantly, you keep delivering features to your customers.

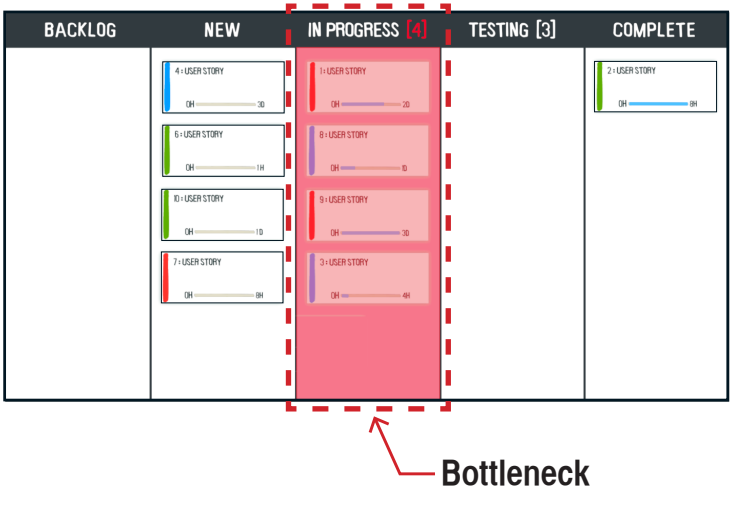


Kanban WIP Limits

To help ensure items are being completed at a steady pace, Kanban can impose limits on the number of items that can live in any one workflow step at any given time.

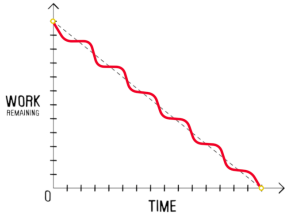


If your team runs into a problem, these limits will bring it to light very quickly by creating a visible bottleneck. This allows the entire team to collaborate or “swarm” on the problem.



Kanban Software Development

WIP Limits help you:

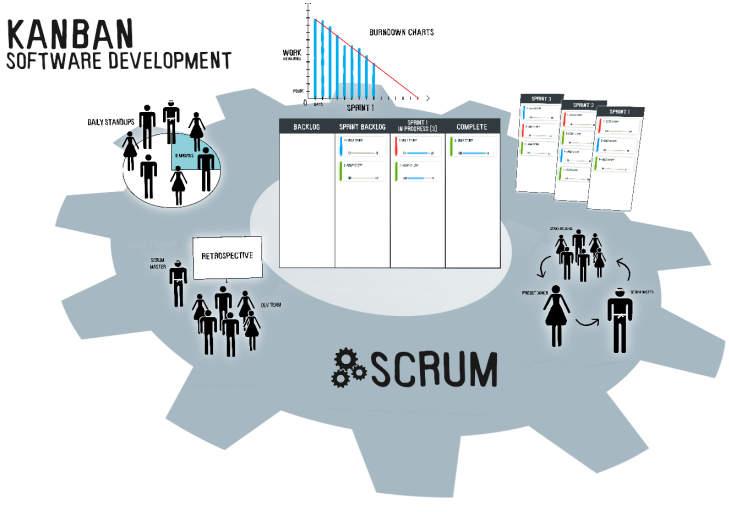


Keep Work Flowing

Save Time by eliminating too much task switching

Complete Tasks

Kanban is fantastic in it's own right and on many projects may be all that you need, but when paired with a good Scrum framework and a great Scrum tool, Kanban really shines.



Scrum provides the structure for organizing feedback, short-term planning, stack ranking, an inspect-and-adapt mindset, and other organizational improvements.

Kanban provides a steady flow of tasks that reach 100% completion by helping your team manage day-to-day development with a minimum of overhead and blocking issues.

Agile Manifesto by Kent Beck, et al.*

We are uncovering better ways of developing software by doing it and helping others do it. Through this work we have come to value:

Individuals and interactions	over	Processes and tools
Working software	over	Comprehensive documentation
Customer collaboration	over	Contract negotiation
Responding to change	over	Following a plan

That is, while there is value in the items on the right, we value the items on the left more.

Customer's Bill of Rights by Steve McConnell**

I have the right:

To set objectives for the project and have them followed.

To know how long the software project will take and how much it will cost.

To decide which features should be included in software.

To make reasonable changes to requirements throughout the course of the project and to know the costs of making those changes.

To know the project's status clearly and confidently.

To be apprised regularly of risks that could affect cost, schedule, or quality, and to be provided with options for addressing potential problems.

To have ready access to project deliverables throughout the project.

Project Team's Bill of Rights by Steve McConnell**

I have the right:

To know the project objectives and to clarify priorities.

To know in detail what product I'm supposed to build and to clarify the product definition if it is unclear.

To have ready access to the customer, manager, marketer, or other person responsible for making decisions about the software's functionality.

To work each phase of the project in a technically responsible way, especially to not be forced to start coding too early in the project.

To approve effort and schedule estimates for any work that I will be asked to perform. This includes the right to provide only the kinds of cost and schedule estimates that are theoretically possible at each stage of the project; to take the time needed to create meaningful estimates; and to revise estimates whenever the project's requirements change.

To have my project's status reported accurately to customers and upper management.

To work in a productive environment free from frequent interruptions and distractions, especially during critical parts of the project.

**(2001) "Manifesto for Agile Software Development". Agile Alliance. Retrieved 14 June 2010.*
*** "Software Project Survival Guide" by Steve McConnell ...a must read.*

Glossary of Agile/Scrum Terms*

Agile Software Development- methods based on iterative and incremental development, where requirements and solutions evolve through collaboration between self-organizing, cross-functional teams.

Burndown Chart- a graphical representation of work left to do over time.

Daily Standup- a short, daily meeting, during sprints, for team communication.

Development Team- a cross-functional group of people responsible for delivering potentially shippable iterations of Product at the end of every Sprint.

Impediment- anything that prevents a team member from performing work as efficiently as possible.

Iteration- the sum of all the Product Backlog Items completed during a Sprint and all previous Sprints.

Kanban Software Development- a method for developing software products and processes with an emphasis on just-in-time delivery while not overloading the software developers.

Product Backlog- a prioritized list of high-level requirements.

Product Owner- the person responsible for maintaining the Product Backlog by representing the interests of the stakeholders, and ensuring the value of the Development Team's work.

Release Backlog- a prioritized subset of the Product Backlog.

Retrospective- a meeting held by a project team at the end of a project to look for ways to improve the process for the next Iteration.

Scrum Software Development- an iterative and incremental Agile Software Development framework for managing software projects, and product or application development.

Scrum Master- the person responsible for the Scrum process, making sure it is used correctly and maximizing its benefits.

Scrum Team- consists of the Product Owner, Scrum Master and Development Team.

Sprint- a time period (typically 1–4 weeks), committed to by the team, in which development occurs on a set of backlog items.

Sprint Backlog- a prioritized list of tasks to be completed during the Sprint.

Sprint Burndown Chart- daily progress for a Sprint over the Sprint's length.

User Story- a feature that is added to the backlog that follows a specific suggested structure: "As (role), I want (feature), so that (benefit)"

Velocity- the rate at which a team gets work done. Example: 50 hours per day.

**http://en.wikipedia.org/...*