Scrum

Problems with Waterfall mode –

#### What’s Wrong with Traditional Software Development?

The traditional way to build software, used by companies big and small, was a sequential life cycle of which there are many variants (such as the V-Model). Commonly, it is known as “The Waterfall”.

1. It typically begins with a detailed planning phase, where the end product is carefully thought through, designed, and documented in great detail.
2. The tasks necessary to execute the design are determined, and the work is organized using tools such as Gantt charts and applications such as Microsoft Project. The team arrives at an estimate ￼of how long the development will take by adding up detailed estimates of the individual steps involved.
3. Once stakeholders have thoroughly reviewed the plan and provided their approvals, the team starts to work.
4. Team members complete their specialized portion of the work, and then hand it off to others in production-line fashion.
5. Once the work is complete, it is delivered to a testing organization(some call this Quality Assurance), which completes testing prior to theproduct reaching the customer. Throughout the process, strict controls areplaced on deviations from the plan to ensure that what is produced isactually what was designed.

This approach has strengths and weaknesses. Its great strength is that it is supremely logical – think before you build, write it all down, follow a plan, and keep everything as organized as possible. The Waterfall has just one great weakness: humans are involved. Hence a lot of problems occur:

##### Creativity is inhibited

This approach requires that the good ideas all come at the beginning of the release cycle, where they can be incorporated into the plan. But as we all know, good ideas appear throughout the process – in the beginning, the middle, and sometimes even the day before launch. A process that does not permit change will stifle this innovation. With the waterfall, a great idea late in the release cycle is not a gift, it’s a threat.

##### Written documents have their limitations

The waterfall approach places a great emphasis on writing things down as a primary method for communicating critical information. The very reasonable assumption is that if I can write down on paper as much as possible of what’s in my head, it will more reliably make it into the head of everyone else on the team; plus, if it’s on paper, there is tangible proof that I’ve done my job. The reality, though, is that most of the time these highly detailed 50-page requirements documents just do not get read. When they do get read, the misunderstandings are often compounded. A written document is an incomplete picture of my ideas; when you read it, you create another abstraction, which is now two steps away from what I think I meant to say at that time. It is no surprise that serious misunderstandings occur.

##### Bad timing

Something else that happens when you have humans involved is the handson“aha” moment – the first time that you actually use the working product. Intraditional software development,you immediately think of 20 ways you could have made it better. Unfortunately, these very valuable insights often come at the end of the release cycle, when changes are most difficult and disruptive – in other words, when doing the right thing is most expensive, at least when using a traditional method.

##### No crystal balls

Humans are not able to predict the future. For example, your competition makes an announcement that was not expected. Unanticipated technical problems crop up that force a change in direction. Furthermore, people are particularly bad at planning uncertain things far into the future – guessing today how you will be spending your week eight months from now is something of a fantasy. It has been the downfall of many a carefully constructed Gantt chart.

##### Too much work and no fun

In addition, a sequential life cycle tends to foster an adversarial relationship between the people that are handing work off from one to the next. “He’s asking me to build something that’s not in the specification.” “She’s changing her mind.” “I can’t be held responsible for something I don’t control.” And this gets us to another observation about sequential development – it is not much fun. The ￼waterfall model is a cause of great misery for the people who build products. The resulting products fall well short of expressing the creativity, skill, and passion of their creators. People are not robots, and a process that requires them to act like robots results in unhappiness.

##### Sub-optimized results

A rigid, change-resistant process produces mediocre products. Customers may get what they first ask for (at least two translation steps removed), but is it what they really want once they see the product? By gathering all the requirements up front and having them set in stone, the product is condemned to be only as good as the initial idea, instead of being the best once people have learned or discovered new things. Many practitioners of a sequential life cycle experience these shortcomings again and again. But, it seems so supremely logical that the common reaction is to turn inward: “If only we did it better, it would work, and ifwe just planned more, documented more, resisted change more, everything would work smoothly”.

**Definition of Scrum**

Scrum (n): A framework within which people can address complex adaptive problems, while productively and creatively delivering products of the highest possible value.

Scrum is:

* Lightweight
* Simple to understand
* Difficult to master

Scrum is a process framework that has been used to manage complex product development since the early 1990s. Scrum is not a process or a technique for building products; rather, it is a framework within which you can employ various processes and techniques. Scrum makes clear the relative efficacy of your product management and development practices so that you can improve.

The Scrum framework consists of Scrum Teams and their associated roles, events, artifacts, and rules. Each component within the framework serves a specific purpose and is essential to Scrum’s success and usage.

The rules of Scrum bind together the events, roles, and artifacts, governing the relationships and interaction between them. The rules of Scrum are described throughout the body of this document.

Specific tactics for using the Scrum framework vary and are described elsewhere.

**Scrum Theory**

Scrum is founded on empirical process control theory, or empiricism. Empiricism asserts that knowledge comes from experience and making decisions based on what is known. Scrum employs an iterative, incremental approach to optimize predictability and control risk. Three pillars uphold every implementation of empirical process control: transparency, inspection, and adaptation.

**Transparency**

Significant aspects of the process must be visible to those responsible for the outcome. Transparency requires those aspects be defined by a common standard so observers share a common understanding of what is being seen.

For example:

* A common language referring to the process must be shared by all participants; and,
* Those performing the work and those accepting the work product must share a common definition of “Done”

**Inspection**

Scrum users must frequently inspect Scrum artifacts and progress toward a Sprint Goal to detect undesirable variances. Their inspection should not be so frequent that inspection gets in the way of the work. Inspections are most beneficial when diligently performed by skilled inspectors at the point of work.

**Adaptation**

If an inspector determines that one or more aspects of a process deviate outside acceptable limits, and that the resulting product will be unacceptable, the process or the material being processed must be adjusted. An adjustment must be made as soon as possible to minimize further deviation.

Scrum prescribes four formal events for inspection and adaptation, as described in the Scrum Events section of this document:

* Sprint Planning
* Daily Scrum
* Sprint Review
* Sprint Retrospective

**The Scrum Team**

The Scrum Team consists of a Product Owner, the Development Team, and a Scrum Master. Scrum Teams are self-organizing and cross-functional. Self-organizing teams choose how best to accomplish their work, rather than being directed by others outside the team. Cross-functional teams have all competencies needed to accomplish the work without depending on others not part of the team. The team model in Scrum is designed to optimize flexibility, creativity, and productivity.

Scrum Teams deliver products iteratively and incrementally, maximizing opportunities for feedback. Incremental deliveries of “Done” product ensure a potentially useful version of working product is always available.

**The Product Owner**

The Product Owner is responsible for maximizing the value of the product and the work of the Development Team. How this is done may vary widely across organizations, Scrum Teams, and individuals.

The Product Owner is the sole person responsible for managing the Product Backlog. Product Backlog management includes:

* Clearly expressing Product Backlog items;
* Ordering the items in the Product Backlog to best achieve goals and missions;
* Optimizing the value of the work the Development Team performs;
* Ensuring that the Product Backlog is visible, transparent, and clear to all, and shows what the Scrum Team will work on next; and,
* Ensuring the Development Team understands items in the Product Backlog to the level needed.

The Product Owner may do the above work, or have the Development Team do it. However, the Product Owner remains accountable.

The Product Owner is one person, not a committee. The Product Owner may represent the desires of a committee in the Product Backlog, but those wanting to change a Product Backlog item’s priority must address the Product Owner.

For the Product Owner to succeed, the entire organization must respect his or her decisions. The Product Owner’s decisions are visible in the content and ordering of the Product Backlog. No one is allowed to tell the Development Team to work from a different set of requirements, and the Development Team isn’t allowed to act on what anyone else says.

**The Development Team**

The Development Team consists of professionals who do the work of delivering a potentially releasable Increment of “Done” product at the end of each Sprint. Only members of the Development Team create the Increment.

Development Teams are structured and empowered by the organization to organize and manage their own work. The resulting synergy optimizes the Development Team’s overall efficiency and effectiveness.

Development Teams have the following characteristics:

* They are self-organizing. No one (not even the Scrum Master) tells the Development Team how to turn Product Backlog into Increments of potentially releasable functionality;
* Development Teams are cross-functional, with all of the skills as a team necessary to create a product Increment;
* Scrum recognizes no titles for Development Team members other than Developer, regardless of the work being performed by the person; there are no exceptions to this rule;
* Scrum recognizes no sub-teams in the Development Team, regardless of particular domains that need to be addressed like testing or business analysis; there are no exceptions to this rule; and,
* Individual Development Team members may have specialized skills and areas of focus, but accountability belongs to the Development Team as a whole.

**Development Team Size**

Optimal Development Team size is small enough to remain nimble and large enough to complete significant work within a Sprint. Fewer than three Development Team members decrease interaction and results in smaller productivity gains. Smaller Development Teams may encounter skill constraints during the Sprint, causing the Development Team to be unable to deliver a potentially releasable Increment. Having more than nine members requires too much coordination. Large Development Teams generate too much complexity for an empirical process to manage. The Product Owner and Scrum Master roles are not included in this count unless they are also executing the work of the Sprint Backlog.

**The Scrum Master**

The Scrum Master is responsible for ensuring Scrum is understood and enacted. Scrum Masters do this by ensuring that the Scrum Team adheres to Scrum theory, practices, and rules.

The Scrum Master is a servant-leader for the Scrum Team. The Scrum Master helps those outside the Scrum Team understand which of their interactions with the Scrum Team are helpful and which aren’t. The Scrum Master helps everyone change these interactions to maximize the value created by the Scrum Team.

**Scrum Master Service to the Product Owner**

The Scrum Master serves the Product Owner in several ways, including:

* Finding techniques for effective Product Backlog management;
* Helping the Scrum Team understand the need for clear and concise Product Backlog items;
* Understanding product planning in an empirical environment;
* Ensuring the Product Owner knows how to arrange the Product Backlog to maximize value;
* Understanding and practicing agility; and,
* Facilitating Scrum events as requested or needed.

**Scrum Master Service to the Development Team**

The Scrum Master serves the Development Team in several ways, including:

* Coaching the Development Team in self-organization and cross-functionality;
* Helping the Development Team to create high-value products;
* Removing impediments to the Development Team’s progress;
* Facilitating Scrum events as requested or needed; and,
* Coaching the Development Team in organizational environments in which Scrum is not yet fully adopted and understood.

**Scrum Master Service to the Organization**

The Scrum Master serves the organization in several ways, including:

* Leading and coaching the organization in its Scrum adoption;
* Planning Scrum implementations within the organization;
* Helping employees and stakeholders understand and enact Scrum and empirical product development;
* Causing change that increases the productivity of the Scrum Team; and,
* Working with other Scrum Masters to increase the effectiveness of the application of Scrum in the organization.

**Scrum Events**

Prescribed events are used in Scrum to create regularity and to minimize the need for meetings not defined in Scrum. All events are time-boxed events, such that every event has a maximum duration. Once a Sprint begins, its duration is fixed and cannot be shortened or lengthened. The remaining events may end whenever the purpose of the event is achieved, ensuring an appropriate amount of time is spent without allowing waste in the process.

Other than the Sprint itself, which is a container for all other events, each event in Scrum is a formal opportunity to inspect and adapt something. These events are specifically designed to enable critical transparency and inspection. Failure to include any of these events results in reduced transparency and is a lost opportunity to inspect and adapt.

**The Sprint**

The heart of Scrum is a Sprint, a time-box of one month or less during which a “Done”, useable, and potentially releasable product Increment is created. Sprints best have consistent durations throughout a development effort. A new Sprint starts immediately after the conclusion of the previous Sprint.

Sprints contain and consist of the Sprint Planning, Daily Scrums, the development work, the Sprint Review, and the Sprint Retrospective.

During the Sprint:

* No changes are made that would endanger the Sprint Goal;
* Quality goals do not decrease; and,
* Scope may be clarified and re-negotiated between the Product Owner and Development Team as more is learned.

Each Sprint may be considered a project with no more than a one-month horizon. Like projects, Sprints are used to accomplish something. Each Sprint has a definition of what is to be built, a design and flexible plan that will guide building it, the work, and the resultant product.

Sprints are limited to one calendar month. When a Sprint’s horizon is too long the definition of what is being built may change, complexity may rise, and risk may increase. Sprints enable predictability by ensuring inspection and adaptation of progress toward a Sprint Goal at least every calendar month. Sprints also limit risk to one calendar month of cost.

**Cancelling a Sprint**

A Sprint can be cancelled before the Sprint time-box is over. Only the Product Owner has the authority to cancel the Sprint, although he or she may do so under influence from the stakeholders, the Development Team, or the Scrum Master.

A Sprint would be cancelled if the Sprint Goal becomes obsolete. This might occur if the company changes direction or if market or technology conditions change. In general, a Sprint should be cancelled if it no longer makes sense given the circumstances. But, due to the short duration of Sprints, cancellation rarely makes sense.

When a Sprint is cancelled, any completed and “Done” Product Backlog items are reviewed. If part of the work is potentially releasable, the Product Owner typically accepts it. All incomplete Product Backlog Items are re-estimated and put back on the Product Backlog. The work done on them depreciates quickly and must be frequently re-estimated.

Sprint cancellations consume resources, since everyone has to regroup in another Sprint Planning to start another Sprint. Sprint cancellations are often traumatic to the Scrum Team, and are very uncommon.

**Sprint Planning**

The work to be performed in the Sprint is planned at the Sprint Planning. This plan is created by the collaborative work of the entire Scrum Team.

Sprint Planning is time-boxed to a maximum of eight hours for a one-month Sprint. For shorter Sprints, the event is usually shorter. The Scrum Master ensures that the event takes place and that attendants understand its purpose. The Scrum Master teaches the Scrum Team to keep it within the time-box.

Sprint Planning answers the following:

* What can be delivered in the Increment resulting from the upcoming Sprint?
* How will the work needed to deliver the Increment be achieved?

**Topic One: What can be done this Sprint?**

The Development Team works to forecast the functionality that will be developed during the Sprint. The Product Owner discusses the objective that the Sprint should achieve and the Product Backlog items that, if completed in the Sprint, would achieve the Sprint Goal. The entire Scrum Team collaborates on understanding the work of the Sprint.

The input to this meeting is the Product Backlog, the latest product Increment, projected capacity of the Development Team during the Sprint, and past performance of the Development Team. The number of items selected from the Product Backlog for the Sprint is solely up to the Development Team. Only the Development Team can assess what it can accomplish over the upcoming Sprint.

After the Development Team forecasts the Product Backlog items it will deliver in the Sprint, the Scrum Team crafts a Sprint Goal. The Sprint Goal is an objective that will be met within the Sprint through the implementation of the Product Backlog, and it provides guidance to the Development Team on why it is building the Increment.

**Topic Two: how will the chosen work get done?**

Having set the Sprint Goal and selected the Product Backlog items for the Sprint, the Development Team decides how it will build this functionality into a “Done” product Increment during the Sprint. The Product Backlog items selected for this Sprint plus the plan for delivering them is called the Sprint Backlog.

The Development Team usually starts by designing the system and the work needed to convert the Product Backlog into a working product Increment. Work may be of varying size, or estimated effort. However, enough work is planned during Sprint Planning for the Development Team to forecast what it believes it can do in the upcoming Sprint. Work planned for the first days of the Sprint by the Development Team is decomposed by the end of this meeting, often to units of one day or less. The Development Team self-organizes to undertake the work in the Sprint Backlog, both during Sprint Planning and as needed throughout the Sprint.

The Product Owner can help to clarify the selected Product Backlog items and make trade-offs. If the Development Team determines it has too much or too little work, it may renegotiate the selected Product Backlog items with the Product Owner. The Development Team may also invite other people to attend in order to provide technical or domain advice.

By the end of the Sprint Planning, the Development Team should be able to explain to the Product Owner and Scrum Master how it intends to work as a self-organizing team to accomplish the Sprint Goal and create the anticipated Increment.

**Sprint Goal**

The Sprint Goal is an objective set for the Sprint that can be met through the implementation of Product Backlog. It provides guidance to the Development Team on why it is building the Increment. It is created during the Sprint Planning meeting. The Sprint Goal gives the Development Team some flexibility regarding the functionality implemented within the Sprint. The selected Product Backlog items deliver one coherent function, which can be the Sprint Goal. The Sprint Goal can be any other coherence that causes the Development Team to work together rather than on separate initiatives.

As the Development Team works, it keeps the Sprint Goal in mind. In order to satisfy the Sprint Goal, it implements the functionality and technology. If the work turns out to be different than the Development Team expected, they collaborate with the Product Owner to negotiate the scope of Sprint Backlog within the Sprint.

**Daily Scrum**

The Daily Scrum is a 15-minute time-boxed event for the Development Team to synchronize activities and create a plan for the next 24 hours. This is done by inspecting the work since the last Daily Scrum and forecasting the work that could be done before the next one. The Daily Scrum is held at the same time and place each day to reduce complexity. During the meeting, the Development Team members explain:

* What did I do yesterday that helped the Development Team meet the Sprint Goal?
* What will I do today to help the Development Team meet the Sprint Goal?
* Do I see any impediment that prevents me or the Development Team from meeting the Sprint Goal?

The Development Team uses the Daily Scrum to inspect progress toward the Sprint Goal and to inspect how progress is trending toward completing the work in the Sprint Backlog. The Daily Scrum optimizes the probability that the Development Team will meet the Sprint Goal. Every day, the Development Team should understand how it intends to work together as a self-organizing team to accomplish the Sprint Goal and create the anticipated Increment by the end of the Sprint. The Development Team or team members often meet immediately after the Daily Scrum for detailed discussions, or to adapt, or replan, the rest of the Sprint’s work.

The Scrum Master ensures that the Development Team has the meeting, but the Development Team is responsible for conducting the Daily Scrum. The Scrum Master teaches the Development Team to keep the Daily Scrum within the 15-minute time-box.

The Scrum Master enforces the rule that only Development Team members participate in the Daily Scrum.

Daily Scrums improve communications, eliminate other meetings, identify impediments to development for removal, highlight and promote quick decision-making, and improve the Development Team’s level of knowledge. This is a key inspect and adapt meeting.

**Sprint Review**

A Sprint Review is held at the end of the Sprint to inspect the Increment and adapt the Product Backlog if needed. During the Sprint Review, the Scrum Team and stakeholders collaborate about what was done in the Sprint. Based on that and any changes to the Product Backlog during the Sprint, attendees collaborate on the next things that could be done to optimize value. This is an informal meeting, not a status meeting, and the presentation of the Increment is intended to elicit feedback and foster collaboration.

This is a four-hour time-boxed meeting for one-month Sprints. For shorter Sprints, the event is usually shorter. The Scrum Master ensures that the event takes place and that attendants understand its purpose. The Scrum Master teaches all to keep it within the time-box.

The Sprint Review includes the following elements:

* Attendees include the Scrum Team and key stakeholders invited by the Product Owner;
* The Product Owner explains what Product Backlog items have been “Done” and what has not been “Done”;
* The Development Team discusses what went well during the Sprint, what problems it ran into, and how those problems were solved;
* The Development Team demonstrates the work that it has “Done” and answers questions about the Increment;
* The Product Owner discusses the Product Backlog as it stands. He or she projects likely completion dates based on progress to date (if needed);
* The entire group collaborates on what to do next, so that the Sprint Review provides valuable input to subsequent Sprint Planning;
* Review of how the marketplace or potential use of the product might have changed what is the most valuable thing to do next; and,
* Review of the timeline, budget, potential capabilities, and marketplace for the next anticipated release of the product.

The result of the Sprint Review is a revised Product Backlog that defines the probable Product Backlog items for the next Sprint. The Product Backlog may also be adjusted overall to meet new opportunities.

**Sprint Retrospective**

The Sprint Retrospective is an opportunity for the Scrum Team to inspect itself and create a plan for improvements to be enacted during the next Sprint.

The Sprint Retrospective occurs after the Sprint Review and prior to the next Sprint Planning. This is a three-hour time-boxed meeting for one-month Sprints. For shorter Sprints, the event is usually shorter. The Scrum Master ensures that the event takes place and that attendants understand its purpose. The Scrum Master teaches all to keep it within the time-box. The Scrum Master participates as a peer team member in the meeting from the accountability over the Scrum process.

The purpose of the Sprint Retrospective is to:

* Inspect how the last Sprint went with regards to people, relationships, process, and tools;
* Identify and order the major items that went well and potential improvements; and,
* Create a plan for implementing improvements to the way the Scrum Team does its work.

The Scrum Master encourages the Scrum Team to improve, within the Scrum process framework, its development process and practices to make it more effective and enjoyable for the next Sprint. During each Sprint Retrospective, the Scrum Team plans ways to increase product quality by adapting the definition of “Done” as appropriate.

By the end of the Sprint Retrospective, the Scrum Team should have identified improvements that it will implement in the next Sprint. Implementing these improvements in the next Sprint is the adaptation to the inspection of the Scrum Team itself. Although improvements may be implemented at any time, the Sprint Retrospective provides a formal opportunity to focus on inspection and adaptation.

**Scrum Artifacts**

Scrum’s artifacts represent work or value to provide transparency and opportunities for inspection and adaptation. Artifacts defined by Scrum are specifically designed to maximize transparency of key information so that everybody has the same understanding of the artifact.

**Product Backlog**

The Product Backlog is an ordered list of everything that might be needed in the product and is the single source of requirements for any changes to be made to the product. The Product Owner is responsible for the Product Backlog, including its content, availability, and ordering.

A Product Backlog is never complete. The earliest development of it only lays out the initially known and best-understood requirements. The Product Backlog evolves as the product and the environment in which it will be used evolves. The Product Backlog is dynamic; it constantly changes to identify what the product needs to be appropriate, competitive, and useful. As long as a product exists, its Product Backlog also exists.

The Product Backlog lists all features, functions, requirements, enhancements, and fixes that constitute the changes to be made to the product in future releases. Product Backlog items have the attributes of a description, order, estimate and value.

As a product is used and gains value, and the marketplace provides feedback, the Product Backlog becomes a larger and more exhaustive list. Requirements never stop changing, so a Product Backlog is a living artifact. Changes in business requirements, market conditions, or technology may cause changes in the Product Backlog.

Multiple Scrum Teams often work together on the same product. One Product Backlog is used to describe the upcoming work on the product. A Product Backlog attribute that groups items may then be employed.

Product Backlog refinement is the act of adding detail, estimates, and order to items in the Product Backlog. This is an ongoing process in which the Product Owner and the Development Team collaborate on the details of Product Backlog items. During Product Backlog refinement, items are reviewed and revised. The Scrum Team decides how and when refinement is done. Refinement usually consumes no more than 10% of the capacity of the Development Team. However, Product Backlog items can be updated at any time by the Product Owner or at the Product Owner’s discretion.

Higher ordered Product Backlog items are usually clearer and more detailed than lower ordered ones. More precise estimates are made based on the greater clarity and increased detail; the lower the order, the less detail. Product Backlog items that will occupy the Development Team for the upcoming Sprint are refined so that any one item can reasonably be “Done” within the Sprint time-box. Product Backlog items that can be “Done” by the Development Team within one Sprint are deemed “Ready” for selection in a Sprint Planning. Product Backlog items usually acquire this degree of transparency through the above described refining activities.

The Development Team is responsible for all estimates. The Product Owner may influence the Development Team by helping it understand and select trade-offs, but the people who will perform the work make the final estimate.

**Monitoring Progress Toward a Goal**

At any point in time, the total work remaining to reach a goal can be summed. The Product Owner tracks this total work remaining at least every Sprint Review. The Product Owner compares this amount with work remaining at previous Sprint Reviews to assess progress toward completing projected work by the desired time for the goal. This information is made transparent to all stakeholders.

Various projective practices upon trending have been used to forecast progress, like burn-downs, burn-ups, or cumulative flows. These have proven useful. However, these do not replace the importance of empiricism. In complex environments, what will happen is unknown. Only what has happened may be used for forward-looking decision-making.

**Sprint Backlog**

The Sprint Backlog is the set of Product Backlog items selected for the Sprint, plus a plan for delivering the product Increment and realizing the Sprint Goal. The Sprint Backlog is a forecast by the Development Team about what functionality will be in the next Increment and the work needed to deliver that functionality into a “Done” Increment.

The Sprint Backlog makes visible all of the work that the Development Team identifies as necessary to meet the Sprint Goal.

The Sprint Backlog is a plan with enough detail that changes in progress can be understood in the Daily Scrum. The Development Team modifies the Sprint Backlog throughout the Sprint, and the Sprint Backlog emerges during the Sprint. This emergence occurs as the Development Team works through the plan and learns more about the work needed to achieve the Sprint Goal.

As new work is required, the Development Team adds it to the Sprint Backlog. As work is performed or completed, the estimated remaining work is updated. When elements of the plan are deemed unnecessary, they are removed. Only the Development Team can change its Sprint Backlog during a Sprint. The Sprint Backlog is a highly visible, real-time picture of the work that the Development Team plans to accomplish during the Sprint, and it belongs solely to the Development Team.

**Monitoring Sprint Progress**

At any point in time in a Sprint, the total work remaining in the Sprint Backlog can be summed. The Development Team tracks this total work remaining at least for every Daily Scrum to project the likelihood of achieving the Sprint Goal. By tracking the remaining work throughout the Sprint, the Development Team can manage its progress.

**Increment**

The Increment is the sum of all the Product Backlog items completed during a Sprint and the value of the increments of all previous Sprints. At the end of a Sprint, the new Increment must be “Done,” which means it must be in useable condition and meet the Scrum Team’s definition of “Done.” It must be in useable condition regardless of whether the Product Owner decides to actually release it.

**Artifact Transparency**

Scrum relies on transparency. Decisions to optimize value and control risk are made based on the perceived state of the artifacts. To the extent that transparency is complete, these decisions have a sound basis. To the extent that the artifacts are incompletely transparent, these decisions can be flawed, value may diminish and risk may increase.

The Scrum Master must work with the Product Owner, Development Team, and other involved parties to understand if the artifacts are completely transparent. There are practices for coping with incomplete transparency; the Scrum Master must help everyone apply the most appropriate practices in the absence of complete transparency. A Scrum Master can detect incomplete transparency by inspecting the artifacts, sensing patterns, listening closely to what is being said, and detecting differences between expected and real results.

The Scrum Master’s job is to work with the Scrum Team and the organization to increase the transparency of the artifacts. This work usually involves learning, convincing, and change. Transparency doesn’t occur overnight, but is a path.

**Definition of "Done"**

When a Product Backlog item or an Increment is described as “Done”, everyone must understand what “Done” means. Although this varies significantly per Scrum Team, members must have a shared understanding of what it means for work to be complete, to ensure transparency. This is the definition of “Done” for the Scrum Team and is used to assess when work is complete on the product Increment.

The same definition guides the Development Team in knowing how many Product Backlog items it can select during a Sprint Planning. The purpose of each Sprint is to deliver Increments of potentially releasable functionality that adhere to the Scrum Team’s current definition of “Done.” Development Teams deliver an Increment of product functionality every Sprint. This Increment is useable, so a Product Owner may choose to immediately release it. If the definition of "done" for an increment is part of the conventions, standards or guidelines of the development organization, all Scrum Teams must follow it as a minimum. If "done" for an increment is not a convention of the development organization, the Development Team of the Scrum Team must define a definition of “done” appropriate for the product. If there are multiple Scrum Teams working on the system or product release, the development teams on all of the Scrum Teams must mutually define the definition of “Done.”

Each Increment is additive to all prior Increments and thoroughly tested, ensuring that all Increments work together.

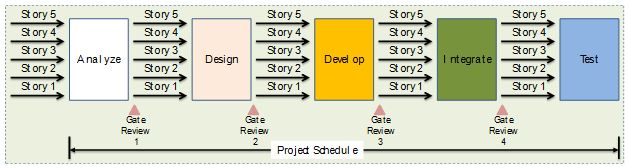
As Scrum Teams mature, it is expected that their definitions of “Done” will expand to include more stringent criteria for higher quality. Any one product or system should have a definition of “Done” that is a standard for any work done on it.

# A Sprint is Not a Mini Waterfall

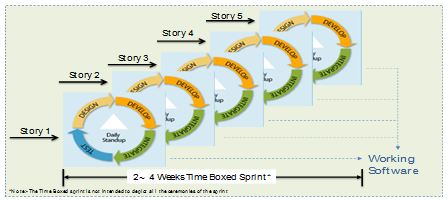
#### 1. Continuous design, development, integration, and testing

The design, development, integration, and testing (DDIT) stage within a Sprint is a continuous activity, not the sequential process it is in a Waterfall project. The graphics below illustrate this difference:

Waterfall Project



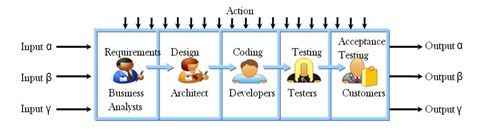
Agile Project



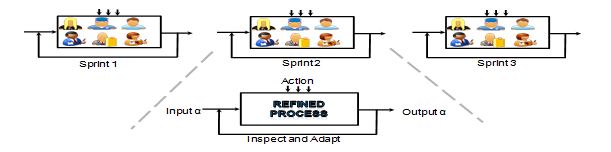
#### 2. Cross-functional team members

A Sprint group is one cross-functional team working in a dynamic, often fluid manner. It is not a set of different teams working separately and handing off work from one to another, as is the case with Waterfall projects:

Waterfall Project Team



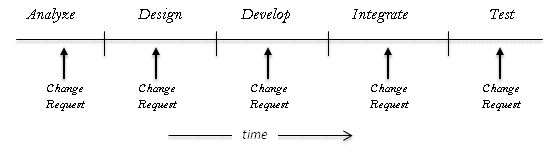
Agile Project Team



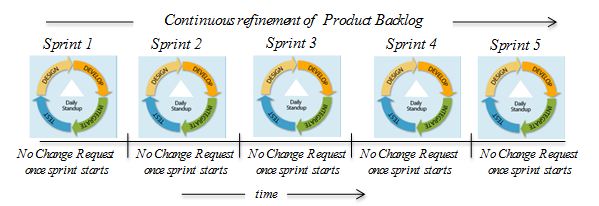
#### 3. No change in scope during Sprint

Waterfall projects define a rigid scope-change control process to manage any changes to work already underway. In contrast, Agile scope-change control, by design, is virtually nonexistent. This gives the product owner the flexibility to change his or her mind between Sprints. However, once the Sprint begins, the scope is frozen and no change request is allowed until that Sprint is complete.

Waterfall Project



Agile Project

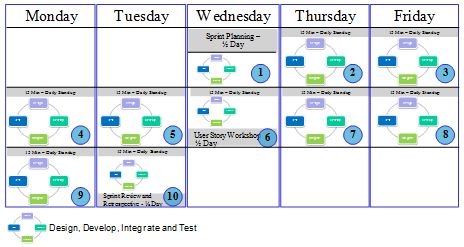


#### 4. Time Boxing

A Sprint is always time-boxed, no matter how the Sprint progresses. In the Waterfall world, if the project is behind schedule, the project timelines are usually extended. In the Agile world, the Sprint duration does not change–it is time-boxed–even if all the committed scope could not be completed during the duration of the Sprint. If, at the end of planned Sprint duration, there are scope items that are not complete, they are put back into the product backlog, prioritized, and made ready for future Sprints.

#### 5. A strictly defined cadence

In a Waterfall project, the cadence cannot be defined up front. A Sprint, in contrast, always works on a well-defined cadence. The graphic below depicts a typical two-week Sprint cadence:



3 – 5 – 3 - 3 Roles, 5 Ceremonies, 3 Outputs.

Roles –

1. Development Team
2. Product Owner
3. Scrum Master

Ceremonies –

1. Sprint Planning
2. Daily Scrum
3. Sprint Retrospective
4. Product Backlog Grooming
5. Sprint Review- Demo/Delivery of the product Increment.

Outputs -

1. Shippable Product Increment
2. Product Backlog
3. Sprint Backlog

Presentation -

<https://34slpa7u66f159hfp1fhl9aur1-wpengine.netdna-ssl.com/wp-content/uploads/2015/06/Scrum-Fundamentals-Webinar-v3.pdf>