# Getting Work Done: A Comparison of Waterfall and Agile as Project Management Frameworks

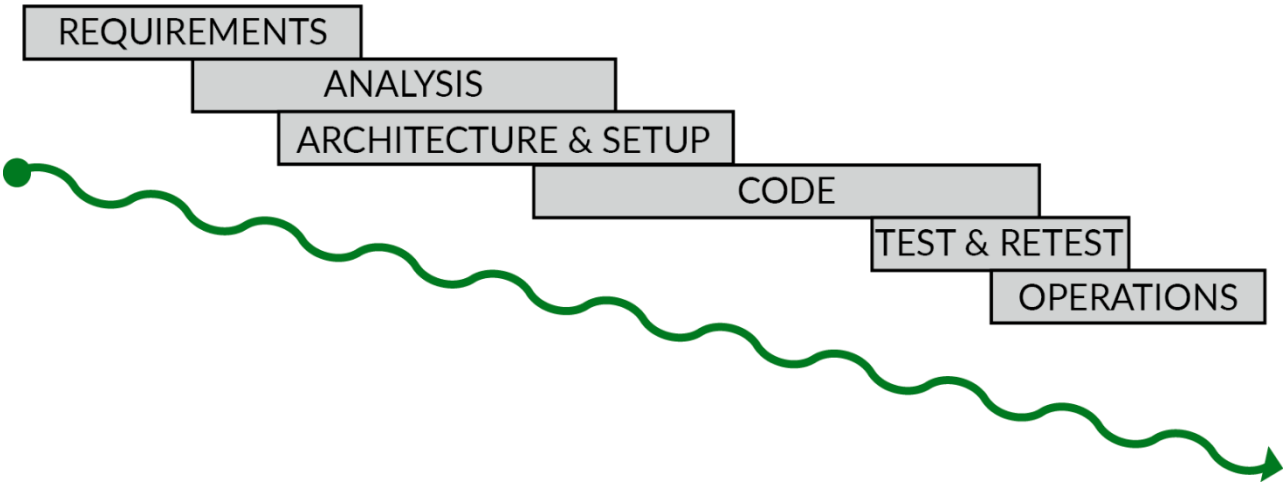
BY ROB SMITH, PMP, PMI-ACP

As the saying goes, there's more than one way to skin a cat – but surely there's a most efficient way. Software developers and IT project managers have spent decades answering this question, searching for the best way to meet ever-evolving demands from a tech-hungry business community.

Initially, The Waterfall Method (waterfall) emerged as software development's standard project management process. It was accepted as flawed from the beginning: The method's cumbersome phases cause an enormous lag time, and frustration with waterfall only grew as the speed of business accelerated.

In response, like-minded professionals determined to find a more timely solution came together in the early 2000s to discuss fresh ideas for a development process. The product of their conversations was coined the Agile method.

We've seen Agile rise to prominence in software development and beyond, but is it right to entirely dismiss the waterfall approach? Might there be a time when it's traditional structure is still beneficial? Keeping in mind the ultimate goal of delivering working software, let's compare and contrast how these frameworks enable teams to get work done.



## What is Waterfall?

Waterfall is a sequential project management process. In other words, a product's life span can be defined by a series of specific phases and each phase is completed in consecutive order. Progress flows steadily downward through the phases – like a waterfall. When the final phase of the sequence is complete, the project is considered done.

As you can see from the diagram on the previous page, all requirements gathering, planning and design work is completed before any coding takes place. Testing of developed code doesn't occur until all or most of the coding is complete, and there is no opportunity to receive feedback or generate incremental value as the project advances. In short, risk is carried throughout the duration of the project, often resulting in a product that is unproven or unfinished.

The waterfall approach typically includes an **initiation** phase for preemptive project groundwork followed by a **planning** phase, an **execution** phase and a **closing** phase. There may also be a series of processes for managing work packages, procurement, risks, reporting, stakeholder management and so on.

## Pros of the Waterfall Approach

Despite its faults, waterfall does move work forward in its own way, and supporters believe the amount of planning involved can be beneficial.

1. When using waterfall for development, potential problems are addressed during the early phases of the process, meaning alternate solutions can be found for obvious issues before any code is written.
2. The waterfall process necessitates extensive documentation during the requirements and design phase. The perceived benefits of this, though often untrue, include the following:
    - Documentation helps new developers assimilate quickly, even in the maintenance phase,
    - All parties have a clear understanding of the needs and expected product outcome,
    - Low chance of surprises when the final product/project is complete,
    - A timeline and budget can be estimated with a high-degree of accuracy.
3. The linear nature of waterfall makes it easy to understand, especially for non-developers or those new to software development, so teams often feel more comfortable and confident with this approach.

## Cons of the Waterfall Approach

Complaints about the waterfall method stem from its rigidity. The process tends to prioritize completion of the plan over delivering value.

1. Solution designers often can't foresee problems that will arise from the implementation of their designs, so early planning stages are often counterproductive.
2. At the beginning of a project, clients don't always know exactly what they

want or need. They'll have a problem that needs fixing and ideas for the final product, but their ideas can be limited by a lack of technological knowledge. Because waterfall requires developers to capture all requirements upfront, incomplete and inaccurate is often gathered, resulting in wasted time and, potentially, the wrong product.

3. Changes to requirements (*e.g.*, adapting for new technologies, changes in a market or updated business goals) can't easily be incorporated with the waterfall method and there are often laborious change control procedures to go through when this happens, meaning the team might be working on the wrong project at a very slow pace.

## What is Agile?

Agile is a software development methodology that eschews a linear, sequential approach in favor of an incremental, iterative one. The most comprehensive and concise definition comes from Scott Ambler in his book "Disciplined Agile Development:"

> *"[Agile is] an iterative and incremental (evolutionary) approach to software development which is performed in a highly collaborative manner by self-organizing teams within an effective governance framework with 'just enough' ceremony that produces high-quality software in a cost effective and timely manner which meets the changing needs of its stakeholders."*

Agile methods are adaptive rather than predictive, adjusting to the inevitable changing circumstances of business and technology. In Agile, time generally spent on upfront planning is dedicated to developing initial prototypes that will help both developers and customers understand the true problem and best solution. Teams and stakeholders learn along the way and incorporate new knowledge into a plan that is made "just in time" with "just enough" detail.

This method is people-oriented. Agile contends that no process will ever make up for the development team's skill, so the process must support the team in doing their work. Agile accomplishes this by requiring close, daily cooperation between business people and developers, advocating for face-to-face conversation, building projects around motivated individuals and allowing teams to self-organize.

An Agile approach allows for changing requirements over time by using cross-functional teams – incorporating planners, designers, developers and testers – which work on successive iterations of the product over short, fixed time periods, called sprints. The work is organized by the team and a representative of the business (the product owner) in to a backlog that is prioritized by business (or user) value. Since cross-functional teams have all the skillsets necessary to produce the product, the goal of each iteration is to produce a potentially shippable increment of working product, which can be demonstrated to stakeholders. Feedback can then be incorporated into the next or future iterations.

## Pros of Agile Methods

1. Working software is delivered frequently, released in successive iterations that can be delivered at a consistent pace.
2. There is closer collaboration between developers and the business.
3. Changes to requirements can be incorporated at any point, even late in development.
4. There is opportunity for continuous improvement in live systems.
5. The entire process is highly transparent.

## Cons of Agile Methods

1. Agile methodologies (*e.g.*, Scrum, XP, Kanban, etc.) are initially more difficult to understand than linear ones.
2. Because the emphasis is on working software, some believe documentation is neglected. The goal is appropriate documentation for the audience that needs it, but if not implemented well, this isn't always the case.
3. When implemented poorly, Agile introduces extra inefficiencies in large organizations or causes teams to struggle against long standing organizational processes.

## Is Waterfall Obsolete?

In today's business and technological landscape, it's a safe bet that Agile is the best option for your project or organization. Agile methods are more flexible than the waterfall methods, so the true needs of a customer are more likely to be met – and is that not the primary motivation for building software?

Agile's constant focus on value means that even when things change, the product you've delivered thus far is or was at some point worthwhile. Plus, the rhythm that Scrum creates helps build highly-motivated teams where productivity increases over time.

Having said all that, there are still circumstance when the waterfall method can work, specifically when requirements are guaranteed to be unchanging, when there is very little uncertainty or when the project if very simple. Mike Cohn, a well-known Agile author, simply stated it this way:

> *"The most appropriate projects for Agile are ones with aggressive deadlines, a high degree of complexity, and a high degree of novelty (uniqueness) to them. We want to use Agile when we are doing something that is new, or at least new to the team building it. If it's something the team has done before over and over, then the team probably doesn't need an Agile approach.*

### Well-Known Waterfall Approaches

**PRINCE2**

Also known as Projects In Controlled Environments, PRINCE became the UK standard for all government information systems projects in 1989.

**PMI PMP**

The Project Management Institute's globally accepted project management certification that follows The PMBOK® Guide. The guide does not dictate a waterfall approach, but its standards are generally associated with the process.

## Putting It in Context

Everything you know about developmental frameworks is useless until applied to a project, and each project has its own challenges and goals. Instead of choosing Agile or waterfall by default, consider which best fits the need. The brief quiz below is designed to help with your decision.

| Question | Yes (1 Pt) | No (2 Pts) |
|---|---|---|
| 1. Do we have clear and complete requirements for the finished product? | | |
| 2. Can we easily predict the amount of effort required to complete the project? | | |
| 3. Have we successfully completed a similar project? | | |

If the score is three points, consider using waterfall. If the score is four or more points, an Agile approach may be more appropriate.

If you determine Agile is a more appropriate approach for your project, use the quiz below determine which style, such as Lean-Kanban or Scrum, is more appropriate.

| Question | Yes (1 Pt) | No (2 Pts) |
|---|---|---|
| 1. Are we developing a new product, new features and/or new functionality that does not currently exist? | | |
| 2. Is a plan required? Do we have some information to start planning? | | |
| 3. Will regular feedback likely clear up customer requirements and/or the development team's understanding of the product to be built? | | |

If the score is three, consider using an iterative approach like Scrum. If the score is four or more, a Kanban approach may be more appropriate.

## Choosing the Right Pilot Project

Many organizations are interested in an Agile transformation, but most are unsure how to start. A pilot project is an ideal way to kick-start the learning process. Finding the right project to use, though – one that will not be easily dismissed nor present an overwhelming amount of challenges – is key. Everyone will have their own opinion about which project to choose, but consider the following:

### Duration

If the project selected is too short, skeptics may claim that Agile works only on short projects. Alternatively, if the project is too long, you risk not being able to claim success until the project is complete. A good rule of thumb is to select a mid-size project within the context of the organization, typically three or four months.

### Size

Ideally, the project should consist of no more than four to five teams. Scaling Agile (multiple teams on one project) introduces additional challenges and requires further practice. If you must scale, start with one or two teams and slowly add more. Also, understand that choosing a project with minimal dependencies on external teams increases the chances of success.

### Importance

It can be tempting to select a low-priority, low-risk project to safeguard the organization and the idea of an Agile transformation among its employees. Don't give into this temptation. Instead, start with a project that is valuable to your company and therefore motivates the team to overcome the challenges a new framework brings. Of course, avoid mission-critical projects. Agile development is usually a disruptive process, and employees should feel comfortable making mistakes as they grow.

# Next Steps

For more information about how these project management frameworks playout in your organization or how to best initiate an Agile transition, contact Louie Bernstein to sign up for a strategy session with one of our Agile coaches.

louie@nTierTraining.com | 404-496-6454

**About nTier Training**

Founded in 2006 on the core belief that there *is* a superior way to train technical teams, nTier designs fully-customizable learning solutions of all sizes and delivers training and coaching around the globe.