# Securing Agile: Getting Speedy Results Safely

**Authors: Alexandru Mircea Rotaru, Larisa Găbudeanu**

In fields like information security, where every second matters, using anything to save time can mean the difference between success and bankruptcy. The Agile Framework promises greater speed for projects by eliminating redundancies wherever possible. However, many believe that information security is one of the redundancies the organization can do away with. This article will show you why this is not the case, and how you can implement Secure Agile in your organization.

## What is the Agile Framework?

The Agile Framework is a means of distributing tasks and delegating roles in a way that gets the job done as efficiently as possible. Much of how its principles stem from the Agile Manifesto from 2001[1]. Agile spreads the work over multiple cycles, called sprints, that progressively add components that come together to form the finished product; in many ways an Agile project is in itself a sprint-like entity, as they both share many of the same key components.

A sprint begins with definitions: goals (definition of done), prioritizing tasks (must have vs. should have vs. could have), the timeline (the duration of the sprint), resource allocation (in particular, which team goes where), task allocation, and many others that are specific to the organization size or any project particularities (such as working with toxic chemicals needing dedicated personnel and constant monitoring). Many of these will have been broken down as a backlog, and a planning session at the beginning of the sprint will determine which items of the backlog will get done by the end of the sprint. Consensus drives the Agile Framework, so the team must gather to decide how exactly to achieve the goals they set out to do for the sprint, and who does what, in order to have everyone on board with what needs to happen.

Once everyone is on board, the team moves into the execution phase. Though the work in and of itself is different every time and comes with its own challenges, a daily meeting, called a scrum, must always occur. The scrum serves to have everyone in the scrum team report

[1] See https://agilemanifesto.org/

on progress from the previous day, decide who will tackle which item will that day, and tackle any disruptions to business as usual.

At the end of the sprint period, the team will gather once more for a Review and Reflect stage. The Review involves presenting progress to the end-customer for approval. The team also needs to Reflect, and figure out how to do better in next sprints based on the one that just ended. The cycle then repeats from definitions, until the project's definition of done gets fulfilled.

Looking at the whole Agile process in general, there are few security concerns; they mostly revolve around keeping the sensitive data within team conversations and the Review safe. Should the employees work remotely, additional concerns regarding unauthorized access to the company system arise. Most information security complications arise from Agile's prerequisites to functioning properly and from scaling, discussed below.

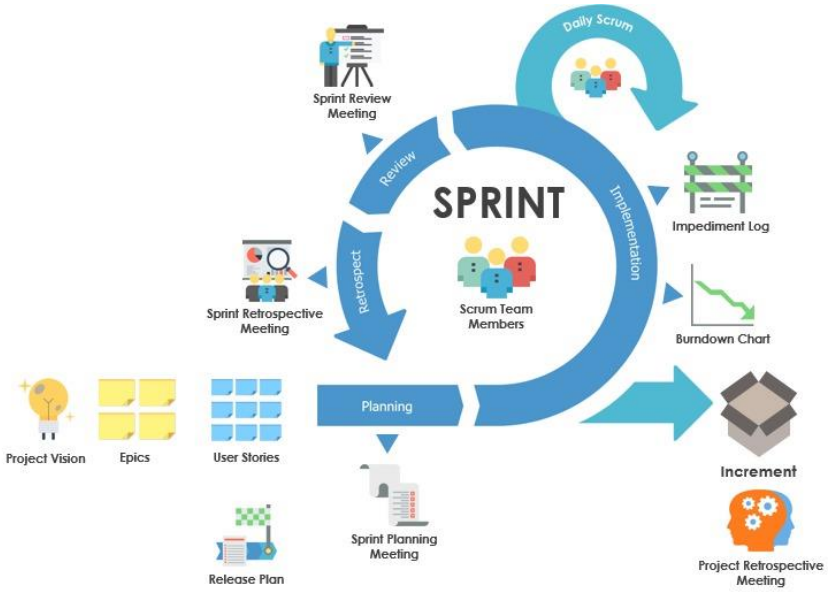


**Figure 29[2]. General Structure of a Sprint.**

## Agile Looks Simple until You have to Scale It

The phrase "two is a couple, three is a crowd" perfectly illustrates how quickly scaling Agile can turn into a manager's worst nightmare. Most organizations employ multiple teams,

[2] Source: https://www.scrum.org/resources/what-is-scrum

which adds a layer of complexity. While the project will mostly run as a series of parallel sprints, with all teams meeting for the Definition and Review and Reflect stages, managing the whole thing is a whole mess in and of itself.

There are multiple frameworks for Agile at scale, such as SAFE and LeSS that can help manage all concurrent sprints to ensure everything runs smoothly. The biggest issues in scaling Agile are communication across sprint teams and dependencies. To ensure effective communication, everyone should be able to maintain interactions with everyone else involved in the project, which is why Agile project should have no more than Dunbar's Number, or ~250 people. Dunbar's Number is the number of connections any person can maintain at any given time.

Dependencies occur when one sprint team needs to complete a task in order for another to begin theirs. All Agile projects should eliminate every possible dependency from their project. However, sometimes, they need to exist; in those situations different sprint teams need to coordinate their actions, which slows down progress.

From an information security perspective, scaling a project creates more pathways for attackers to breach your organization's cybersecurity - with more people to target and more information being passed around. Further, there may be differences in approaches for bits and pieces of the IT solution that are being developed in different sprints (especially parallel sprints). Thus, scaling Agile also requires scaling security with it, which can be a burden if the team is too large.

That being said, the ideal scenario would employ minimizing the number of teams working on a given project; that way, scaling becomes easier to handle. To keep the other teams from idling, your organization can run multiple independent projects at the same time. Granted, removing too many employees slows down the project, which is why you need to find the ideal balance.

## Nothing in Life is Free. So, What am I Trading for Speed?

The main thing you will be trading for speed is scope - features that may not be critical to the project's success, but which your organization may find useful. For instance, a car cannot function without an engine or breaks, but it can work without an in-built entertainment system. Agile is an end-justifies-the-means (within moral and legal limits) kind of framework, where the only thing that matters is if the outcome of the project is able to address the issues at hand and have the features as written in the project definition of done.

Agile also needs consensus and trust to work. This, in turn, requires adjusting the organization culture to embrace speed, trust, and communication. This kind of investment is not unlike electrifying a heavily-used rail corridor to reduce the rail company's carbon footprint

as part of their commitment to be environmentally-friendly. Therefore, Agile cannot function without having a strong HR Department; a culture of Diversity, Equity, and Inclusion; and a commitment to fostering communication and allowing dissent.

One thing you should never trade for speed is security. Every second that passes represents a chance for an attacker to find a vulnerability or a loophole that compromises your systems; once that happens, everything your company is legally obligated to protect becomes vulnerable, and the sprint grinds to a halt until the incident gets closed. So, either way, losing a little speed by implementing information security protocols is worth it, compared to losing days and millions in fines and lawsuits if you don't implement them. Also, the cost of implementing the information security protocols scales with the sensitivity of a project that gets compromised during a cybersecurity attack; therefore, if you tailor the information security protocols to the project's sensitivity and specifics, you should have little trouble getting buy-in from both upper management and the scrum teams.

## Trust Your People Enough to Tell You when the Ship has a Hole, so You Don't Sink with It

The big boon for information security that comes from Agile is its reliance on free and open communication: that way, people will be able to voice their concerns about something and usually help mitigate a vulnerability before it even becomes a problem.

That being said, there's no room for "ego" in "Agile." Your teams need to be able to openly criticize anything wrong with the project, otherwise you could be leaving vulnerabilities undiscovered; this is why consensus must drive a project: everyone brings something to the table through their unique perspective, meaning that any team member might miss something that another might find obvious.

## Begin with the End in Mind: Getting Something Done

The whole purpose of using Agile is solving a problem: that's the entire reason you have a project to begin with. Any problem has a multitude of solutions, some better than others, but your project needs to have at least one by the end - which can then be improved as part of a new project. This leaves room to play with scope, as some things may be done too soon, while others may get delayed for any number of reasons - the most time-consuming of which is when integrating the various modules that each sprint team produces doesn't go according to plan.

That being said, even though you can vary scope to your organization's desire to stay on schedule, there are some things you should never compromise on - chief of which being information security. One way in which you can make sure your organization doesn't turn

information security into something beyond the project's scope is to introduce it in the definition of done.

## Just Writing "Keep the Project Secure from Cyber Threats" in the Definition of Done Sounds Unusually Simple. What's the Catch?

Keeping a project secure is a layered approach. You need to have every step of your sprint secure, then the sprint as a whole secure, and finally the project as a whole secure. Securing Agile is similar in many ways to a window: the windowpane itself represents the individual steps of every sprint - one hole and you need to replace it -; the window locks represent the sprints as a whole - if one is compromised, the window cannot be closed -; and the window hinges represent the project as a whole - if they fail, your window is useless.

Setting-up from the beginning, based on the architecture of the IT solution and the IT landscape of the organization, certain baseline requirements for specific components can speed the analysis and implementation phases for each sprint. Further, it helps in establishing a coherence in approach and in the level of security throughout the newly developed IT solutions.

In addition, for each sprint and each component of the IT solution, additional analysis from a security perspective (e.g. secure software design, network and integration security) should be had in mind. In practice, this can be achieved through review by security staff of the stories for each sprint and, ideally, participation in the sprint meetings. Of course, certain aspects can be automatized, such as the code review part. Similar analysis should be performed after partial/full integration of all components of the IT solution and when the preparation for deployment in production takes place. In case of shortage of resources, certain milestones can be set for further apart for security review to take place.

## This Information Security Infrastructure is Massive. How can I keep it Going at the Same Speed as Everyone Else?

As mentioned before, Agile begins to slow down at scale. So, one way to manage the information security infrastructure is to tailor it to every project within the first sprint. At the same time, you would need a company-wide information security team monitoring every project, to maintain the larger picture. It's like the window and hinges analogy from before: you need a solid window that locks to keep you safe, as well as sturdy hinges so it doesn't just fall off.

Agile allows you to model every project as an independent mini-organization. Therefore, you can simply use the information security infrastructure already presented in previous articles

and apply it to a project as if it were an organization - and remove anything that doesn't apply. The key difference here is that you need to integrate their work in the scrum team, which also runs the risk of creating dependencies. So, again, you need to reach consensus with management, the scrum team, and the safety team before moving forward in any way.

## I need Speed to Protect my Data. Can I use Agile to do that?

Not exactly. Though Agile does provide incredible speed, protecting data is more expert-based, and the tasks are unpredictable. The Lean Framework fits the situation much better. Lean is basically a ticketing system that many companies are already using: write down every task on a ticket, prioritize the tickets received, and then execute the tickets from most to least important, minimizing work in progress.

## So, Agile Can't Make Me Breakfast?

Sadly, when it comes to shiny new methodologies or technologies, people rush to implement them without a thought - which is incredibly problematic. The same goes for Agile: it's only effective in the settings it was designed to operate - which are mostly projects that require some degree of innovation, that aim to resolve concrete, specific issues.

There are other methodologies out there that can be better suited for your situation. For example, restaurants and tech support frequently use Lean to manage their operations, simply because of how unpredictable the timing of their issues is, and because they usually operate with smaller-scale projects.

Traditional methodologies work incredibly well for replicating already-existing results. This usually works in construction and mass-production, which made up most of any nation's economic output until Agile came around. Traditional methodologies also work with anything that has incredibly stringent requirements - which is usually the case in things like sculpting statues of historical figures.

So, before you dig into Agile, make sure to ask yourself: is this the kind of project that best fits the Agile framework? If the answer is yes, then, by all means, go ahead! If you're not sure, hopefully, this section can shed some light as to when to use which methodology.

## Conclusion

To sum up, the Agile framework breaks down your organization's work into its atoms while also maintaining a larger picture of a project's end goal, which is also what your information security team needs to do to keep it secure. With this framework, treating every

project as its own independent mini-company will help you reuse already existing information to your advantage - in true Agile style. Finally, you can simplify the work your information security and scrum teams do by minimizing dependencies.