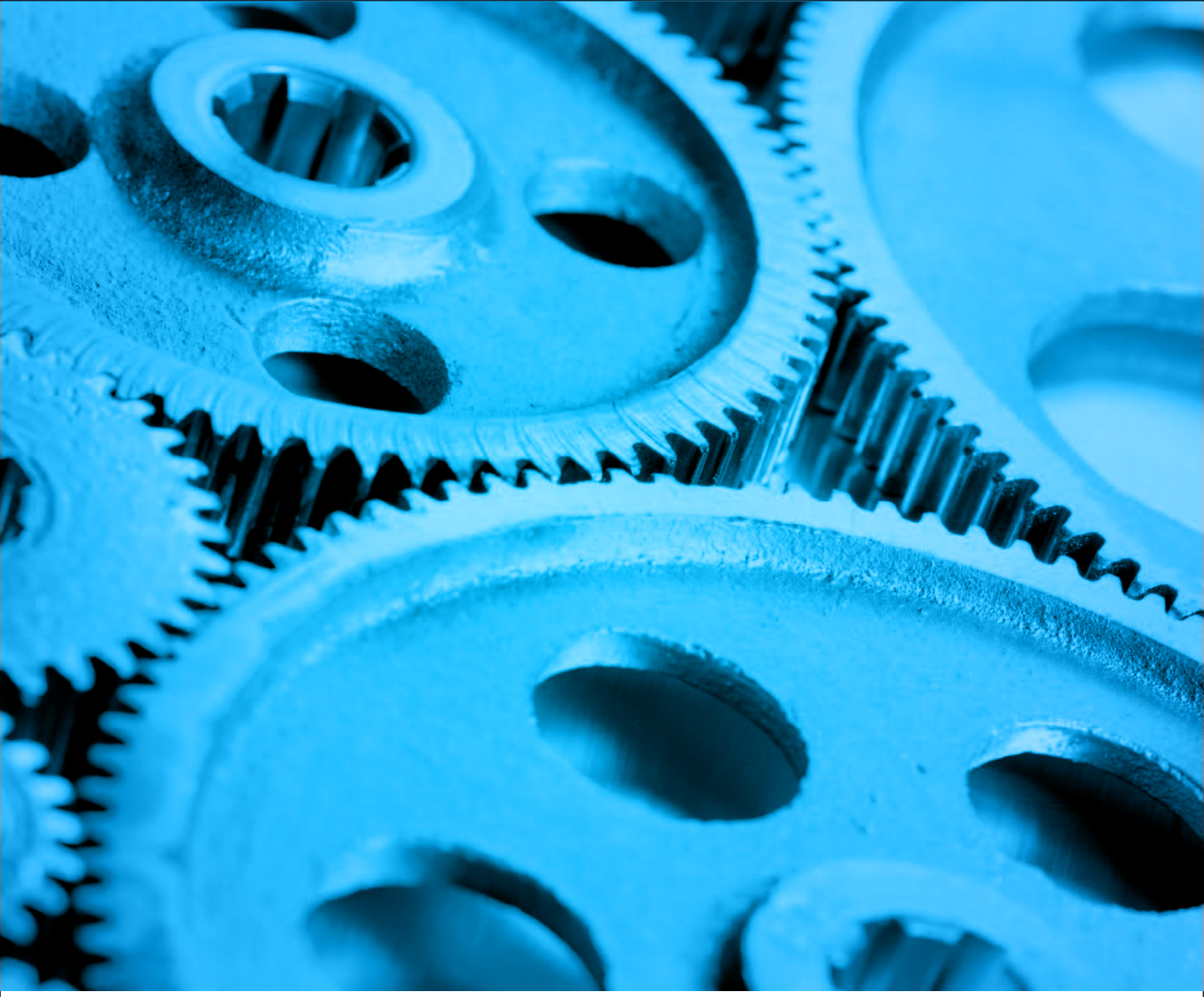


Using Agile Methods for creating better products faster

Using Agile Methods for creating better products faster

In the recent past, a new class of software development methodologies has found favor - Agile methods.

We can apply the term "Agile" to a group of specific approaches that include Extreme Programming, Scrum, DSDM and Adaptive Software Development. While these methods may vary at a detailed level, they embody a common philosophy - adaptive, iterative, methodologies built around the themes of "just enough" and constant communication. Agile methods ensure faster and better outcomes especially in scenarios of varying requirements and fast time to market.

The origin

Developers began experimenting with Agile methods through the '90s owing to a growing dissatisfaction with the conventional waterfall method. The waterfall method had its uses, but the sequential, structured method tried improving the predictability by discouraging changes during the design and development process. In addition, traditional software development methods stress heavily on documentation as a means of freezing and elaborating requirements, and this in turn added to the time lines.

This made it unsuitable for scenarios where the user was unsure of the requirements at the start. Thus, traditional methods were found to be unsuitable for enterprise software development as dynamic requirements and short turnaround times were the order of the day. The Agile Manifesto sought to get around the various pitfalls of traditional methods by stressing on:

- Individuals and communication over processes and tools
- Working software over comprehensive documentation
- Customer collaboration over contract negotiation
- Responding to change over following a plan

Software product development and Agile methods

Consider the case of building an enterprise software product. Imagine there is a bunch of developers with a "great" product idea they have validated by talking to select customers and prospects. Through these interactions they have

- A good perspective on the market worthiness of the product idea
- They also have a broad idea of the scope of the product
- They have a fair idea of must have and nice to have features

They now come up with a "time-boxed" software product development plan that ensures there are releases of working software to the customers at frequent intervals of time. Now, what exactly is time boxing? A time box has a duration of a week to four weeks typically with a deliverable or outcome at the end. The important point to note is the deadline is fixed up-front - a release may comprise more than one time box, depending on the complexity of the development effort.

The development team "releases": the software after each time box to the set of customers/ users, who then use it and give feedback. In fact, even while developing the product, they constantly interact with the customer community to decide what is necessary and when.

At the end of the exercise, this process leads to a more market driven and relevant product for two reasons- collaborating with users and often giving users working versions that allows them to "touch and feel" the product.

For a moment, superimpose the waterfall method on this scenario and the answer is evident- it is infeasible to mesh this real life scenario with the structured and change resistant waterfall method.

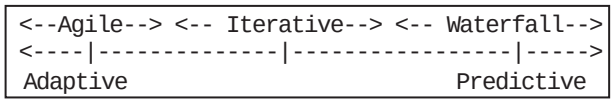
On the other hand, Agile methods allow you to do precisely the above- follow an iterative, collaborative approach that involves users and assumes that software development needs to consider change rather than discourage it.

Clearly, such a method, as explained above, leads to a superior product, as well as one that can hit the market quicker.

What Agile is and what it is not

In software jargon, we term Agile variously as a movement or philosophy and its interpretation has also been subjective. In this section, we offer our interpretation, as practitioners of this method -

Agile methods are adaptive rather than predictive



Source: Wikipedia

This is the most fundamental difference or perhaps the key differentiator of Agile methodologies. Traditional methodologies do most of the detailed planning up-front; assuming a well thought through and documented design would ensure a perfect product. This approach is similar to the way one would go about a construction. The architect discusses the requirements with the client and comes out with a drawing and detailed design, and once that is signed off the construction begins. Given that it is a brick and mortar exercise, it is hard to consider changes once the building gets underway.

On the contrary, the premise of Agile methods is that users cannot decide all changes early in the life cycle. It is likely that once the user sees a version of the product, they get ideas that may warrant changes to the original design and scope.

Agile outcomes can be predicted

One may infer that Agile methods lead to a chaotic and unstructured way of software development- while that may seem like an intuitive inference, in reality, a well implemented Agile methodology leads to more predictability and this is how-
One of the banes of traditional methods is that it does not allow for change and delivers the product in an almost fully finished form. In a complex development effort, the time elapsed between freezing a design and delivering a product may be significant. Meanwhile, the business environment may have changed, new challenges may have cropped up, and users may have identified new needs. Now, when the product finally does arrive, they find, contrary to their expectations, the product has not turned out the way they now envisage.

This either leads to their "canning" the initiative or asking for large rework- certainly not the predicted outcome of the exercise.

Switch back to Agile now- how would the development team go about the process? They would elicit high-level and high priority needs of the user and market and estimate a date by which the user needs these features, to be most relevant and useful.

They would then do another exercise of prioritization of this list too and then commit to a delivery date.

They would then decide on a suitable architecture and begin the development- along the way, they would involve the users to confirm the detailed assumptions and design. Once the developers build the prioritized set of features, the software is shared with the users for feedback. In addition, users have flexibility to change their mind on features the development team is yet to incorporate.

This process may be followed a few t ' if they begin the final product shapes up. It is not hard to infer that this method leads to a more predictable outcome, even if it does not allow for as predictable a process. And therein lies the difference- while traditional methods highlight the process so much that it reigns supreme, Agile methods make the process an enabler rather than an imposition. They mandate the process should allow for changes and variations to result in a more predictable outcome- the needed software on time.

Agile processes highlight real time communication

By their nature, Agile processes are collaborative- they work well only when there is high degree of involvement between users and developers. The process assumes time boxes that result in frequent intermediary deliverables that users review and provide feedback on. The advantage of this approach is that all users are almost fully informed of what they will get at the end of the exercise, and there are no surprises. At the same time, they also need to commit time and effort to join in the reviews and give relevant feedback.

Agile processes speed up time

Given that Agile methods allow requirements to change all through the process, it is natural to assume that Agile methods lead to longer life cycles. However, statistics show that Agile methods result in quickened product development. As users are involved through the life cycle, the chances of a late discovery of the outcome not matching expectations are low. Also, it gives a chance for users to review and change the overall scope at every intermediary step. Besides, users buy in to the product and feel that they "co-own" the product - which is important for a successful outcome.

Second, given the method of prioritizing needs and agile's focus on "working code", users get a feel of the software far earlier than they would with a traditional sequential method. In fact, what may even happen at times is that users prune the requirements and shave off features that look superfluous and unnecessary as they go along.

Agile methods are cost-effective

We are always led to believe that good planning and execution always leads to more efficient outcomes. True in many real life scenarios, but not with software. The challenge with software development is that users may come up with requests all the time. In a process that does not cater to this behavior, the cost of effecting changes becomes high.

As discussed earlier, traditional methods sometimes lead to large-scale rework, costs, as well as time escalation leading to the common refrain that X % of software projects do not adhere to their original budgets.

In contrast, Agile methods underscore frequent deliverables, and give an opportunity not just for requirement changes, but even for paring down requirements based on real need, rather than "expected need".

VAssure

VAssure develops software products for small to medium sized software companies and uses agile yet distributed teams around the globe. To ensure success, VAssure has created a repeatable methodology called "Velocity" based on Agile principles with a platform of open source and commercial tools. We have adapted and integrated Velocity into a remotely deployable and manageable platform on which we can develop quality software products quickly.

Proprietary aspects of "Velocity"

Collaborative specification:

The qualities, structure and delivery mechanism of all new features and improvements to the product are specified in a way that increases the effectiveness of chiefly asynchronous communication. Global engineering teams do not have the luxury of scheduling impromptu meetings to seek clarifications. VAssure has defined a method and configured a web based collaboration system specifically designed to improve the effectiveness of asynchronous thought and knowledge sharing.

Database-driven work method:

Software products developed using an Agile process need continuous prioritization of all the requirements, tasks and defect removal activities in each time-boxed release.

VAssure has defined an estimation, prioritization and scheduling process to manage this in a distributed and cost-effective manner.

In addition, VAssure has adapted a web based issue tracker for global teams to manage and oversee the continuous changes that are part of the Agile method.

IP protection:

VAssure has built a globally deployable platform that contains all the collaboration, repository, build and tracking databases that contain the intellectual property of our customers. As our partners are sensitive to protecting these assets, VAssure has set up a secure network, operating system and managed audit service to manage these assets.

In conclusion:

Agile methods have repeatedly demonstrated getting quality products to market faster. Agile's intrinsic nature of considering and planning for varying requirements with real time and constant communication between users and the development team make this possible. It thus offers a sound alternative to developers desiring to develop market-ready and superior products in quick time.