Agile and commitment

Author: Han Schaminée

Contributors: None yet

Version 0.2

Introduction

There is a general misconception that teams who apply Agile software development methods no longer want to commit to customer deliveries. I believe, on the contrary, an Agile way of working enables you to meet customer commitments in a much better and effective way than traditional project management methods. The misconceptions may have come from the fact that with Agile, the team is transparent from day one onwards on the uncertainty. And many managers perceive transparency on uncertainty as a proof of lack of commitment; they are not interested in problems or risks; you just have to deliver (management by commitment). But this management style does not help the project. In this document, I will explain why I believe Agile development methods help to improve predictability and better meet customer commitments.

Executive summary

In many cases a supplier has to commit to a delivery deadline. Traditional project management recognizes projects often have to be executed within constraints like scope, costs and time and these constraints are strongly linked to each other. But there is intrinsic uncertainty in software development projects and these traditional methods do not deal effectively with that uncertainty. Think before you act can certainly reduce the uncertainty, but in many cases, it cannot take it away completely. We better apply a system that allows early indication corrective measures are required.

You can organise product development by the means or by the ends, I mean cluster all specification, design, implementation and test efforts for all features (Waterfall), or finish one feature after another (Agile). The Waterfall approach seems to do better justice to the 'think-before-you-act' principle, but it is felt the intrinsic uncertainty in software development creates a lot of waste. Uncertainty is best managed by lower work in progress which is the basis for the Agile way of working.

With an Agile way of working, we can

- better manage uncertainty in scope by just-in-time deciding on the scope
- better manage uncertainty in costs by providing an effective adaptive control system on progress
- better manage uncertainty in time by continuously delivering a releasable product

And last but not least it is proven that by minimizing WIP, quality will increase. This will increase customer satisfaction but also increases productivity by removing substantial unnecessary waste.

Deliver on commitment

In many cases a supplier has to commit to a delivery

In today's world, many companies depend on supplier deliveries that will be integrated in their customer product. That integration often requires a complex synchronisation of supplier deliveries and own activities. And activities here not only mean product development activities, it also includes many other activities like marketing, preparing a store, organising a production flow, etcetera. Where in the 20th century the manufacturing industry has developed more advanced methods (like JIT) for these synchronisation problems, in many other areas the synchronisation is still managed via fixed time schedules and committed dates.

And even when you as a supplier would very much welcome more advanced synchronisation methods, the customers often impose their way of working on the supplier and the supplier just has to live with the fact their performance is often judged against the fact whether or not the agreed product is delivered at the committed date. I know many examples, where later the product was just sitting on the desk of the customer for quite some time. Indeed, the manufacturing industry found out that synchronisation by planning often leads to waiting and waste. But it seems very difficult to change these commitment practices and companies better find a way to deal with it if they want to stay in business.

Project constraints like scope, time and costs are strongly linked

The iron triangle in project management claims that projects need to be performed and delivered under constraints like scope (features and quality), time and costs are strongly linked: One cannot be changed without impacting the other. A further refinement separates quality or performance from scope into a fourth constraint. Traditional project management is very much based on managing these constraints

There is intrinsic uncertainty in software development projects

Different than manufacturing, software development is characterised by the fact that most activities are new and done for the first time. It is not so easy to predict what it will take to bring an activity to completion. Moreover, it is also not so clear what the customer expects. Very often his expectations only become clear when the product is ready. Many projects have a huge amount of specification change requests during the project or rapidly after and many software companies improve the business case of their original offering by abusing the customer position when they better understand what they really want. It often creates unnecessary waste in the system and higher costs, often to be paid by the customer or end-user.

Rather than neglecting it, is seems better to accept there is intrinsic uncertainty in the project constraints when developing software, both in costs and specification. Calling them constraints, as done in project management literature, already indicates a denial of the uncertainty in these elements.

There can also be uncertainty in the expected time, but when the customer manages the time as a commitment, that uncertainty seems less relevant for this document. But as said before, there are many examples where the initial commitment date turned out to be less relevant.

Traditional project management does not deal effectively with uncertainty

Traditional project management often applies change procedures to deal with scope uncertainty, but it creates waste for activities already performed. Sometimes waste cannot be avoided as we only know at the very end of a feature is required or need modification. But to secure the end date, we better receive that feedback rather sooner than later to allow for corrective measures.

Traditional project management create buffers on the critical path to secure in-time delivery. But, as explained by Goldratt, buffers on the critical path do not help as delays only accumulate. GANTT charts are a typical example of introducing buffers on the critical path and are therefore to be considered as evil, although they may help a lot to visualise dependencies. In terms of system theory, synchronous systems introduce waiting and costs of waiting are not always outweighed by the costs of increased complexity of asynchronous systems.

We better install an adaptive system where initial project execution helps to improve later phases. We learn when we go on what the customer wants and what it takes to meet the delivery expectations. But this is not so easy when the later activities are of a totally different nature than the initial activities (like coding and test compared to requirements analysis). We better organise our project in a way where initial activities provide more information about later activities. Also, bringing forward higher risk items will help to improve predictability.

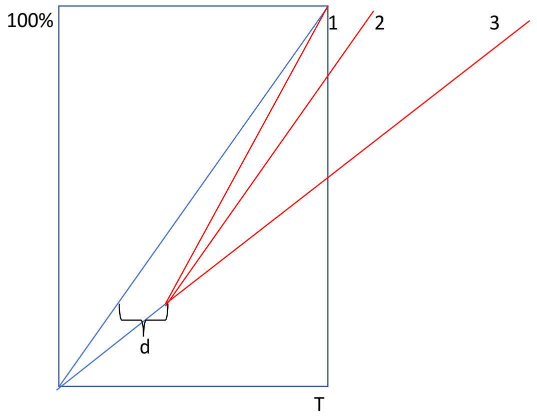
Think before you act can reduce uncertainty but not take it away

Adaptive control is a control method that can help when parameters show variation or uncertainty. Feedback loops allow the system to react to changes. Robust control assumes that the variations are small and will not seriously impact the output. As there is uncertainty in software development projects, feedback mechanisms that allow for adaptation of the execution seem relevant and more suited that design a robust control system, where we try to think of all issues in advance and define the control for that.

Having said that, I believe there is absolutely value in trying to reduce the uncertainty as much as possible. A good market analysis on how the product should look like is worth the effort. Spend sufficient time before the start of the project on design and architecture is a good idea and will reduce the need for corrective actions in the adaptive control system. And another misconception about Agile is that we don't need this thinking in advance anymore.

But thinking in advance will never take away the uncertainty and the need for continuous adaptation.

There is a need for early indication corrective measures are required



As explained in the figure, projects often suffer from insufficient learning from the first phases of the project. Suppose we have a plan to deliver 100% of the functionality at time T and that at a certain time the project faces a delay d. I recognize three types of project managers:

1. The optimists: I will be able to catch up, just by doing some hard work
2. The simple ones: the end date will be delayed by the same amount d.
3. The realistic ones: unless we change something in the project, there is no reason not to assume that a similar build-up of delay in the second half may be assumed, so we better either take corrective measures or report a continuation of the build-up of the delay.

In most projects, I have looked at, method 2 was applied, although method 1 is also quite popular, especially with managers who face the strong commitments, but don't see how to control.

Also, many tools support approach number 2.

I believe the only correct approach is number 3. It invites project managers to think about corrective measures in an early phase and allows for an adaptive control system. But it requires you are able to measure progress.

You can organise product development by the means or by the ends

Product development projects aim to deliver a number of features. Each of the features require specification, design, coding and testing. It feels like a matrix, where you can organise the project by clustering the various phases (waterfall) or by organising it feature by feature. The waterfall approach seems to do better justice to the 'think-before-you-act' principle.

Incremental product development provides better means to deal with the uncertainty

Uncertainty plays a role in many other industries, like in manufacturing and telecommunications. They have found that the best way to deal with uncertainty is limit the work in progress (WIP) and with that minimize waste. This is all excellently explained by Reinertsen. By clustering activities like in Waterfall you maximise WIP. By organising your work as an iteration of product increments, you minimise WIP.

Manage uncertainty in scope by just-in-time deciding on the scope

With an Agile way of working we decide on the specification of a feature at the beginning of the product increment that delivers that feature. This allows for last minute changes to the specification of the feature or the exchange of features by new ones. In this way, it avoids unnecessary waste of activities of features that will never be released. But it also provides the required flexibility to deal with changing market demands or changed market insights.

Manage uncertainty in costs by providing an effective adaptive control system

Within Agile, features are implemented one by one and until a state that is release ready. The number of features accomplished is reflected in a measure called velocity. As on average the work to implement features is comparable (more comparable that different phases like analysis, design, coding or testing), this velocity can be used to pretty reliably predict future progress of the project. Especially when features are ordered in a way where high risk items are done first. Although there may always be variation in velocity, when velocity substantially stays behind the plan, corrective measures like extra resources or scope reduction may be considered. At least, the organisation is able to take corrective measures, they no longer fly blind in a world of uncertainty.

Manage uncertainty in time by continuously delivering a releasable product

Although customers often impose hard time requirements on the projects, also their plans may change, changing the time requirements to either earlier or later. By continuously (or at regular intervals) releasing a product that meets quality standard, the customer has more freedom to decide what release to use in their product. In this way, incremental design help to provide the required flexibility to deal with this uncertainty.

By managing WIP, Agile delivers better quality

With the iterative ways of development as proposed by Agile, we continuously deliver a quality product. But also within an iteration, the Agile way of working is minimising work in progress. And as proven in many studies, there is a direct relation between WIP and defect density, the number of defects per 1000 lines of code. So, it also helps to improve quality, which is often a major reason for slip in late phases of the project. So, also in this way Agile helps to increase customer satisfaction but also increases productivity by removing substantial unnecessary waste.