RISK TO REASON

Managing Project Risk with Agile

By Nathan Donaldson

Risk to Reason

Managing Project Risk with Agile

Nathan Donaldson

This book is for sale at
http://leanpub.com/managing-risk-with-agile

This version was published on 2014-04-07

Contents

Managing software development risk with Agile **1**
Agile: pragmatic tools for managing risk in projects 1
The 4 risks common to most software projects . 2
Managing risk in software development projects using Agile 6

About the author 19
Nathan Donaldson 19

Managing software development risk with Agile

Trying to predict the future is like trying to drive down a country road at night with no lights while looking out the back window.

– Peter Drucker

Agile: pragmatic tools for managing risk in projects

With Agile now the de facto approach for software development, it is essential to understand how we can use Agile approaches and tools to manage and mitigate risk.

Worldwide it has become clear that software projects have high rates of failure and even when ostensibly successful often don't deliver the value that was promised at project inception.

When we talk to organisations about the benefits of Agile software development, all too often we get the response that the current priority - Big Project X - is too important to fail and that Agile is too risky.

In this white-paper we will look at:

- the Agile Risk Management Model
- the four different kinds of risk faced by software development projects
- how we can use the Agile Risk Management Model to manage risk.

The 4 risks common to most software projects

There are many risks in software development projects, some of them generic and others specific to a particular project and situation. We will look at four of the most common and obvious risks that apply to almost all software projects.

The first risk is that we will fail to deliver the quality that our market needs.

Second, we will not deliver within the resources available. The third risk is that we will fail to deliver within the timeframe needed.

And the fourth, often most underestimated risk, is that we will not deliver a product the market needs or wants.

Poor quality

Time and time again we hear about large project failures in the media. In New Zealand the latest and greatest is the Ministry of Education's payroll system. Delivered 12 million dollars over budget (greater than 100%) and at a universally agreed level of poor quality, Novapay's problems and lack of success have remained in the media for more than 12 months. Even today, one year after launching, the media still reports that more than 300 defects remain to be corrected and that school payrolls are not being processed correctly.

Poor quality is a root cause of many failed projects. It destroys team productivity as more and more time is eaten up with finding and fixing defects. The teams that work on these systems often have one defining trait - they are terrified of touching the code base. They know that it is fragile but have no idea of where or how badly.

Too expensive

The second risk is that the project will consume all available resource and still not be in a complete state. Take this commentary on the Commission of Inquiry's report into Queensland's health care payroll system as an example.

> ... the commission report's finding that the fiasco-which saw an effort to replace Queensland Health's legacy payroll system at an expected cost of A$6.19 million (fixed price)

> explode into one that will cost around A$1.2 billion to develop and operate properly when all is said and done-“must take place in the front rank of failures in public administration in this country. It may be the worst.”[1]

Delayed delivery

The third risk is that the project may not be delivered in time. Many projects are extremely time sensitive, and missing deadlines means lost revenue, lost market share and sometimes penalties.

Poor quality is a common cause of missed delivery deadlines. This problem is exacerbated by late integration, meaning that issues of quality are not discovered until very late in the project.

Perfectly executed failure

The fourth risk is that businesses may hit their deadlines, make their budgets and even deliver the necessary quality only to find that they have delivered a product nobody wants. Eric Ries describes it as “achieving failure”.

> “We spend a lot of time planning. We even make contingency plans for what to do if the

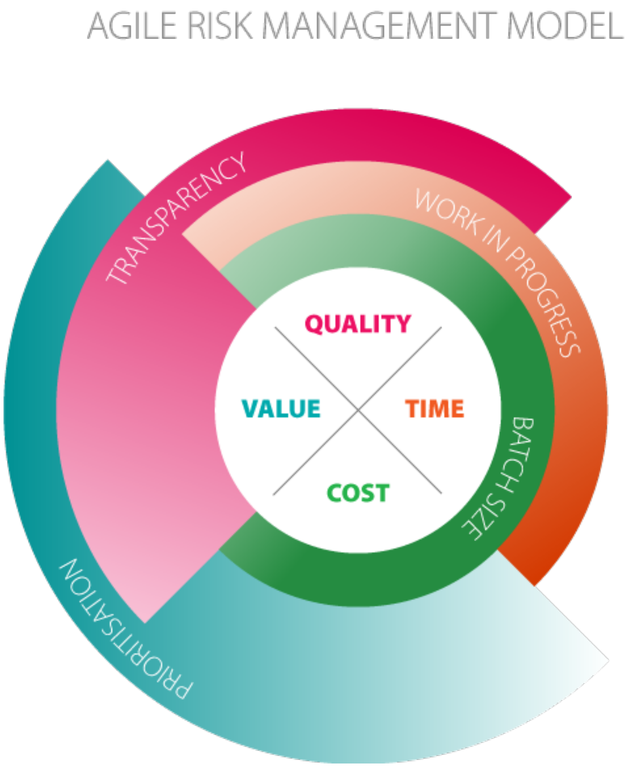
[1]http://spectrum.ieee.org/riskfactor/computing/it/queensland-government-bans-ibm-from-it-contracts

> main plan goes wrong. But what if the plan goes right, and we still fail?"[2]

To reduce the likelihood of these risks impacting our projects, we need to stop and ask ourselves what we need to do differently in our management approach to ensure we deliver the right value to our organisation and our customers.

[2] Eric Ries:http://www.startuplessonslearned.com/2009/01/achieving-failure.html

Managing risk in software development projects using Agile



Agile software development has a number of ways to reduce the four risks common to most software projects. In this white paper we will focus on transparency, batch-size, limiting work in progress (WIP) and prioritisation.

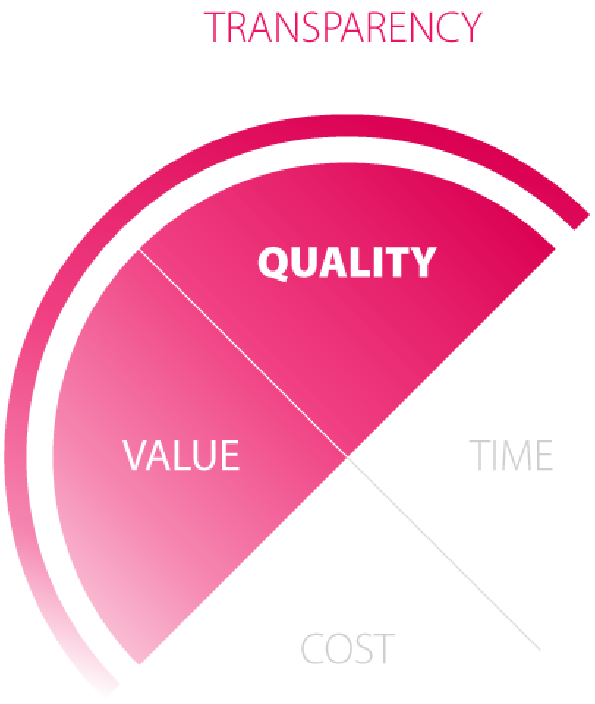
We propose an Agile Risk Management Model for managing and mitigating risk. In this model you will see that different Agile approaches mitigate different risks, with each approach primarily affecting one of the areas identified as well as influencing the other areas.

Alongside the model we describe various tools for managing the common risks. Taken together, these tools greatly increase our chances of successfully delivering projects for our organisations.

The tools enable us to prioritise the continual, incremental delivery of working software. By prioritising the delivery of working software we are able to constantly inspect the quality, functionality and desirability of the software.

Like tasting food while you are cooking, we taste at every stage of the project and adjust the ingredients and seasoning as needed to ensure a flavourful and well-balanced dish.

Increasing Transparency

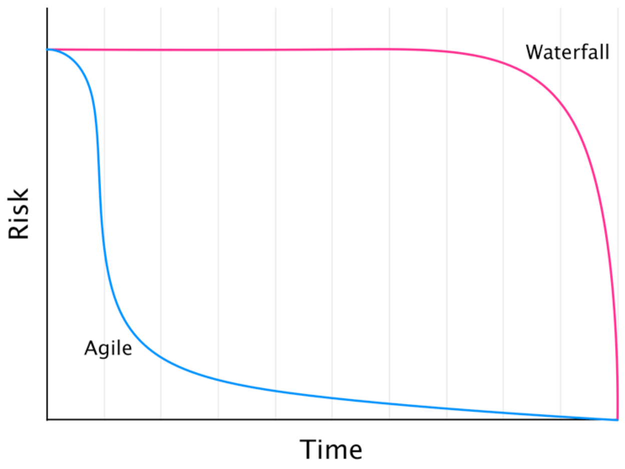


Increasing transparency reduces the risk of producing work of poor quality and helps to ensure that the project is delivering value.

Transparency is at the heart of reducing risk. If we do not have a full and accurate view of the status of the project, the business priorities, the resource needed to deliver and the quality of the output, we cannot manage risk in the project.

This is a key problem in many software projects, as it's not until the final stages of a project, the release, when we really get to examine the quality of the product. We rely on wishful thinking to carry us forward.

Consider the graph below. We can see that risk remains high in a Waterfall project right until the majority of the functionality is delivered. At the end we are able to get a real view of the risk, which may be high or low depending on how well our requirements matched market need and whether our work has been of consistently high quality.



Contrast this with the Agile project shown on the graph. Here we see that initially risk was high - identical to the

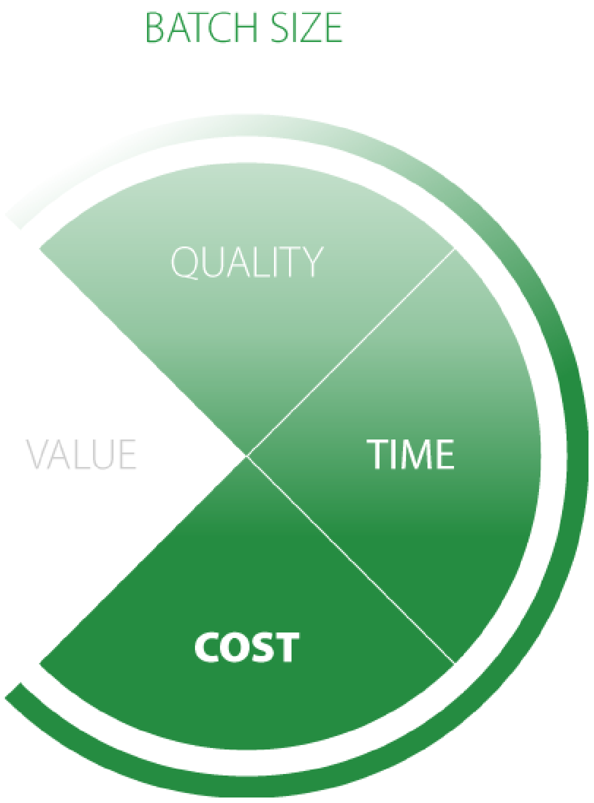
Waterfall project. Using Agile the risk drops quickly and significantly as we start to deliver working, tested and deployable software. We can determine the quality of our work and take our software to a test market to determine the value our features provide to our users.

During an Agile project we can shift the focus at any stage. Should we discover that our users need feature X instead of feature Y we can re-plan and re-prioritise without any penalty to our overall timeline.

Agile works to make projects more transparent in a number of different ways. An example is the iteration time box in a Scrum project. It makes transparent the work the team is doing and the amount of work that can be completed by the resource applied. It also enables us to inspect working software.

Other examples of increased transparency include: - regular "retrospectives" - meetings every iteration where the team can raise blocks, impediments and inspect the way they are working together and with the organisation - Kanban boards, Scrum boards and other "information radiators" - they make visible the work of the team and the progress being made, and are a rich source of information about the team's processes for the team and the organisation.

Batch size



Reducing batch size reduces the risk of cost and time overruns while also increasing quality.

Core to reducing the variability in our projects is reducing the size of the batches we work in.

Agile projects constantly work to reduce the batch size in every part of the project. Whether in the daily work of the team or in the delivery of the product or project, reducing batch size remains paramount.

Software development projects have traditionally seen the project as one large batch that must be delivered in a big bang. Even iterative approaches see the iterative deliverables as part of a larger deliverable batch.

Reducing batch size is important for managing cost on projects for three reasons:

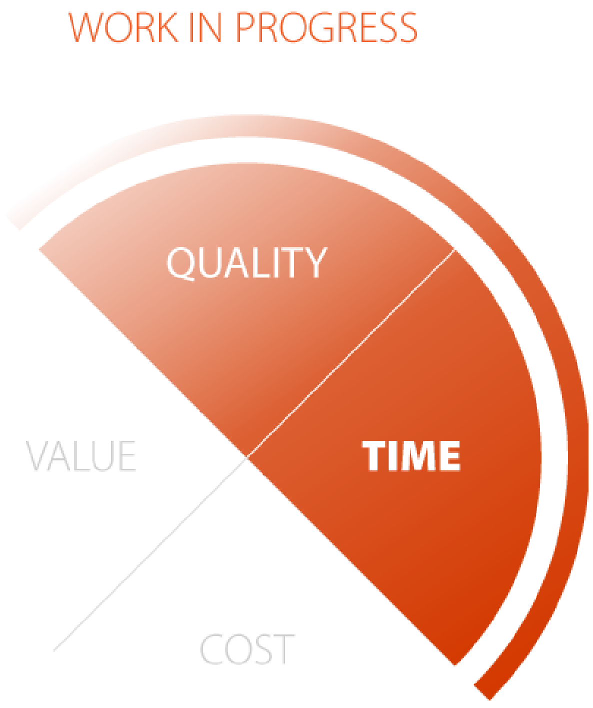
1. it enables us to be more productive
2. it reduces the variability in the work
3. it enables us to inspect the output of our work at greater frequency.

Reducing the size of the batches we work in reduces the potential amount of rework needed to remediate errors. We can catch problems with our output, processes and tools at an early stage, limiting the extent and impact of an error.

Tools for managing batch size in Agile include:

- splitting stories into the smallest increment of value
- working on vertical slices of a system
- continuous integration
- increasing the frequency of deployment

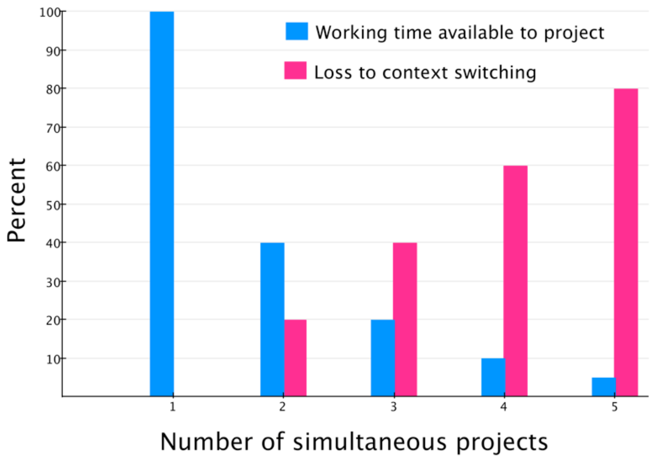
Limiting Work In Progress (WIP)



Decreasing WIP reduces the risk of missing deadlines and also increases the quality.

Neglecting to limit the work in progress for our teams and organisations quickly leads to thrashing, as we constantly switch from one task to another.

We can see in the graph below (based on Gerald M. Weinberg's - *Quality Software Management: Systems Thinking*) that as we have more tasks in progress at any one time productivity starts to fall to zero as we spend more time context switching.

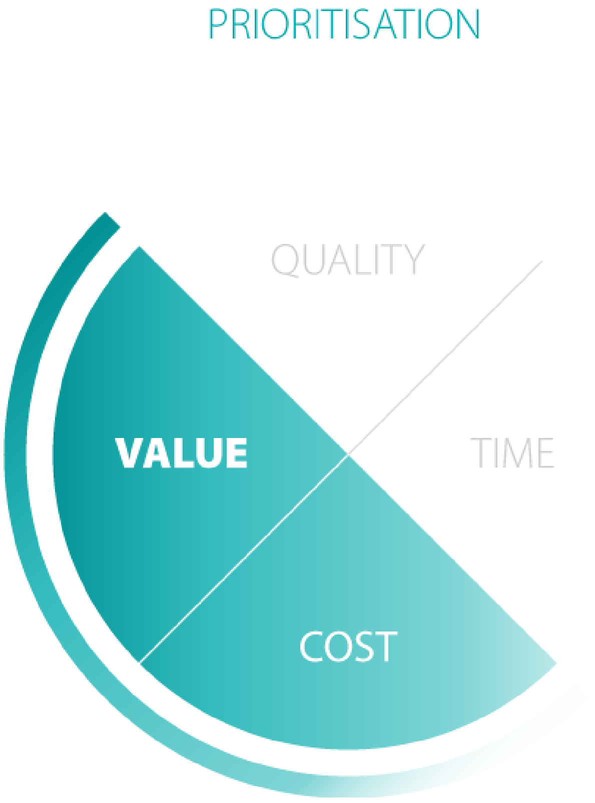


With many of the Agile methodologies there is a focus on managing and reducing work in progress. Scrum does this using time-boxes. The amount of work in progress is limited by the size of the time-box (2 weeks for the majority of teams). The team takes on no more work than can be completed in that time. High-performing teams actively work together to reduce their work in progress by ensuring that each story (or increment of work) is fully completed, tested and ready to deploy before moving on to the next story.

In addition to time-boxes there are a number of other tools for limiting work in progress, including:

- explicit WIP limits in Kanban stages
- continuous integration
- test and behaviour driven development
- protecting the team from outside requests.

Prioritisation



Rigorously prioritising work reduces the risk of delivering software of low value while ensuring that costs are controlled by only working on what is important to our organisation and customers.

When working with Agile methodologies such as Scrum and Kanban we focus on prioritisation. Focusing on the highest value work is a key method of driving risk out of the system. It avoids time and energy being diverted into low value work and enables us to deliver value to our customers much earlier than in traditional Waterfall projects.

An example of prioritisation is the mobile app Reel Choice that Boost New Media produced for NZ On Screen. When writing the user stories we made sure that all stories contained some value to the end user, completed all stories to a releasable state and got feedback from customers as we worked.

Our backlog for the application looked like this:

- display one video full screen
- show multiple videos and let the user choose which to view
- when the screen is in portrait view, show additional information about the video
- enable certain videos to be “featured” and displayed larger in the multiple video listing.

We worked through the stories in priority order and at every stage we had a potentially shippable product that offered value to the end user and the client. At any stage the client could have stopped the project and had a working, valuable product.

This is in stark contrast to alternative project management methods where the plan determined at the beginning of the project must be followed to completion to see any value delivered.

Using Agile delivery project work is strongly prioritised as the assumption is that all functionality will and must be delivered in small batches. In other types of iterative development processes work may be delivered iteratively but without the focus on prioritisation based on value to the customer that is central to Agile.

In Agile projects re-prioritisation happens continuously in the backlog of work as more information is discovered about the project. In contrast with staged project delivery methods (and more traditional iterative delivery methods), priorities are set once at the start of the project, if at all, and are often based on imagined future needs and dependencies.

Working with the model

The Agile Risk Management Model describes the relationship between Agile approaches and the risks common to software development projects.

For organisations, it demonstrates how Agile actively mitigates risk in ways that traditional methodologies do not. For teams, it clarifies how different but interrelated Agile approaches and tools address the risks they face on projects everyday.

About the author

Nathan Donaldson

CEO Boost Agile

I am a business coach, specialising in business transformation. I ensure that organisations can successfully transform and take advantage of industry best practice.

Organisations are dealing with high staff turnover, poor product quality and extended delays in delivery. Ultimately this results in a lack of competitiveness in the marketplace.

It doesn't have to be this way. Transforming your business into a Lean/Agile organisation is like replacing your old, inefficient Model T Ford with a new, modern, efficient car. My coaching equips your leaders and teams with the tools they need to deliver the most value to your organisation.

Driven by a passion for change

I'm dedicated to finding new ways to grow individuals, teams and organisations.

When I started my career I worked 12 hour days, day in day out, for months on end. I can honestly say this was the least productive time of my working life. Since then I have strived to find a good work/life balance and in the process maximize my productivity.

A history of success

I own and manage Boost New Media, a thriving software development company and Boost Agile, Asia/Pacific's leading Agile/Lean consultancy. I've worked with many intelligent and passionate people and have enjoyed coaching them and their teams through Lean/Agile transformations.

I provide coaching, mentoring and training to leaders and teams that enables them to effect lasting and measurable change.