Why agile?

An Agilesphere white paper

Version: 1.0
September 14, 2014
Author: Jeremy Renwick, jeremy.renwick@agilesphere.co.uk

Summary

This paper frames the business case for agile across three key categories:

- providing practices and a culture to embrace changes in requirements
- focusing on early and incremental delivery increases return on investment
- reducing generic project / programme risks

This paper uses the term “projects” as shorthand to mean projects, programmes and portfolios as the business case points apply as much to programmes and portfolios as they do to projects. It also uses the word “timebox” to be equivalent to “sprint” or “iteration”.

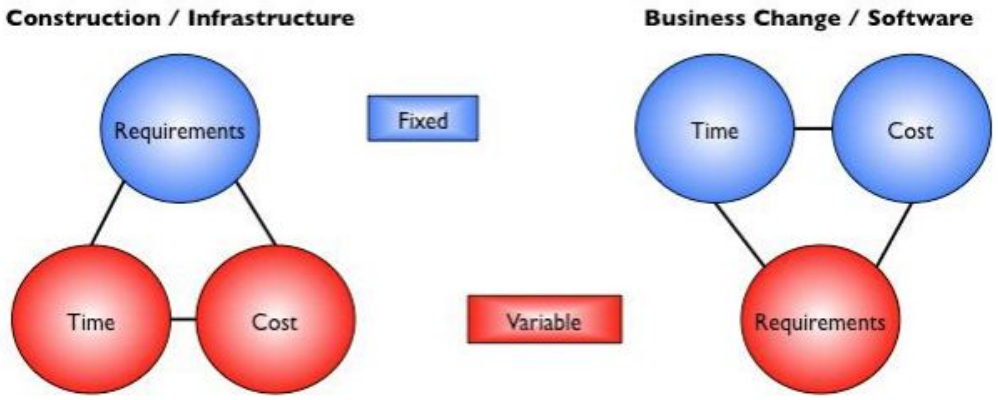
Embracing change

When managing projects, there are three primary controls: **time**, **cost** and **requirements**.

In most traditional project management literature, **quality** is added as a fourth control (but we take the view that quality is part of requirements) or **requirements** is replaced by **quality** if **requirements** are assumed to be fixed.

This assumption was valid when traditional project management practices, processes and tools were developed because they were created to manage construction projects where fixed requirements (blueprints) were already established practice.

In reality, requirements can always be varied - even in construction projects to some extent - and the reason software as a concept was developed was because it can be changed easily.



When considering business change or software development projects, the assumption that requirements are fixed is flawed for a number of reasons:

1. The business and its environment doesn't stand still during the 6 to 18 months of a typical project or programme.

2. The businesses' understanding of the requirements will evolve. Projects manage the uncertain and the new – not everything can be known at the beginning.

3. When requirements are fixed, this puts a cap on delivery so there's a strong incentive for people specifying them to make up requirements "just in case," which is wasteful.

4. If the business has a hard deadline to meet or a fixed budget to operate in, then it needs to be able to vary requirements to meet those deadlines and budgets.

From this, we can see that a project needs to embrace changes in requirements. Agile provides the culture, techniques and management to do this by:

● focusing on breaking the requirements into small pieces to see the value each piece delivers to the business

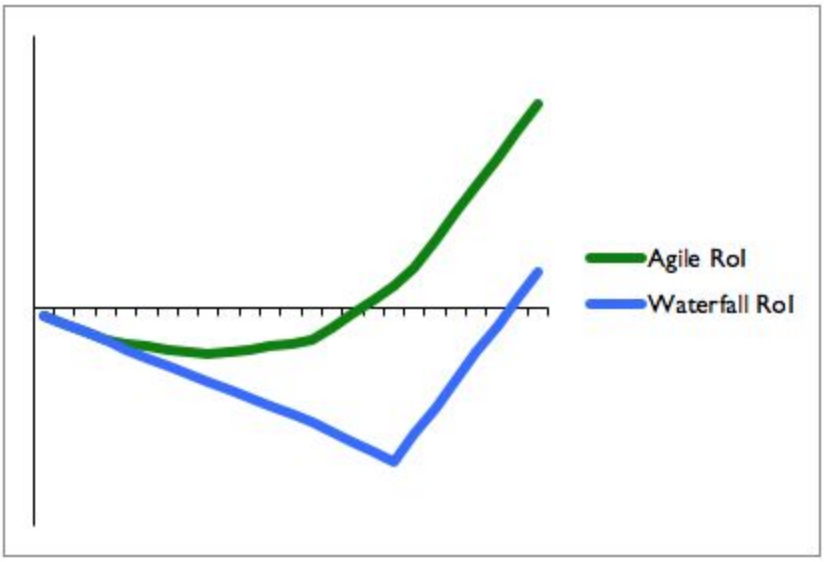
● prioritising those small pieces of value according to business need

● delivering some of these pieces ready for the business to use every 2-6 weeks

● structuring the delivery so that every 2-6 weeks there's an opportunity for the business to adjust requirements or re-prioritise

● explicitly embracing change into the culture of the delivery team

As traditional techniques assume requirements are fixed, they only manage change in a bureaucratic way – if at all. In practice, when faced with an imminent deadline, traditional projects cut and prioritise opportunities because they don't have the tools to be disciplined.



Increased Return On Investment (ROI)

No project, programme or portfolio delivers any business value until the changes have become "business as usual" (BAU).

In traditional projects, it's all about cost (investment) until everything is finished. This typically takes between 6 and 18 months.

Agile projects are structured to deliver change into BAU as soon as it's ready – where possible, every 2-6 weeks, so they start recouping the investment sooner, as illustrated by the graph.

However, there's one qualification that has to be acknowledged. It's a fundamental property of all change projects that no two projects are the same, which means that direct or detailed comparisons between traditionally-managed projects and agile projects can't be drawn.

But the basic logic is almost impossible to argue against. By delivering value into BAU regularly, the return on investment for agile must be higher for traditional techniques.

Another equally important aspect is that by delivering real value into BAU regularly, there's early transparency as to whether the project is on track or not. This clarity is not available in traditional projects.

And if an agile project is cancelled, at least the project has some return on the investment. In a traditional project, all the investment is lost, leading to failing projects continuing much longer than they should because the organisation doesn't want to write off the sunk cost.

Reduced project risk

In any project there are some generic risks. Agile techniques help delivery teams manage these risks better in traditional techniques.

Changing requirements risk

It is worth repeating that requirements change and this is both likely and necessary to keep the project focused on delivering real and current value to the business. Traditional projects do not handle requirements change well: agile projects have change built in.

Estimation risk

All estimates are guesses. In most cases they're based on education and experience, but they're still guesses. This should be expected because each project is unique and manages the unknown, the uncertain and the future.

Estimates are important in any project to help manage expectations and establish plans for delivery timescales and costs.

In an agile project, estimates are reviewed both formally and informally at regular intervals, usually at the timebox boundaries every 2-6 weeks. This might also be the case in traditionally-run projects, but the key difference is that in an agile project, because complete functionality has been delivered, actuals are available to compare against estimates – giving the team the opportunity to learn and improve estimation accuracy.

Testing risk

All project deliveries have to be tested. In traditional projects, testing is done after all (or at least most) of the requirements have been delivered. This results in a lengthy period, often many weeks or months of testing and retesting while defects are fixed.

In agile projects, each requirement is tested as it's delivered and testing is part of each 2-6 week timebox.

Agile projects are better tested than traditional projects because testing a small number of requirements each timebox:

● reduces the complexity as there are fewer dependencies. This also makes defect root cause diagnosis easier

● means that the focus of the delivery team in ensuring the delivery meets requirements is much higher

● significantly reduces the pressure on the delivery team to cut corners on testing, particularly, close to “go live” deadlines

● provides real feedback on how well testing is being done as early as possible in the project and well before any deadline so it can be improved

● catches any regression defects immediately, provided the tests are automated and included as part of the software build process

This last point is important. Agile exposes a regression testing overhead that usually can’t be distinguished from other parts of the testing phase of a traditional project. With software, this overhead is mitigated using modern development tools that automate these tests and run them each time the software is built.

In practice, software developed using agile engineering is regression tested at least once a day and this regression testing pack runs to many thousands of tests.

While regression testing the business process aspects of a change project may give the impression of extra work, this is a small, additional investment to make for improved quality and transparency.

Go-live / integration risk

Testing environments are never exactly the same as production environments by definition, so there are always “to-dos” that are unique when putting a change into BAU. Often, these aren’t completely identified until the team actually tries to put something live.

Traditional projects tend towards a “big bang” approach to going live. In an agile project, the team is looking to put change live as soon as it’s tested and signed off. Typically, this change is a small part of the end solution. By going live early, issues can be identified and resolved before higher pressure deadlines later in the project – real-world feedback reduces risk.

Key person risk

In nearly all projects there'll be key person risk – a small handful of people whose unavailability would severely compromise delivery. This is usually expressed as "what happens if they win the lottery / get run over by a bus", but the actual risk is more mundane and consequently higher probability.

The real risk is that key individuals decide to do something else and in most cases, this will be because the current project no longer motivates them – or worse.

Agile helps manage this risk because of the motivational impact of regular delivery of functionality and so being able to see the impact of what you're doing. Also, the culture of a good agile team is one that's collaborative, non-hierarchical and has reduced bureaucracy – traits which most people find attractive.