Agile: Myths and Truths

By Laurence Nicholson
Founder and Managing Director of N Cubed Consulting

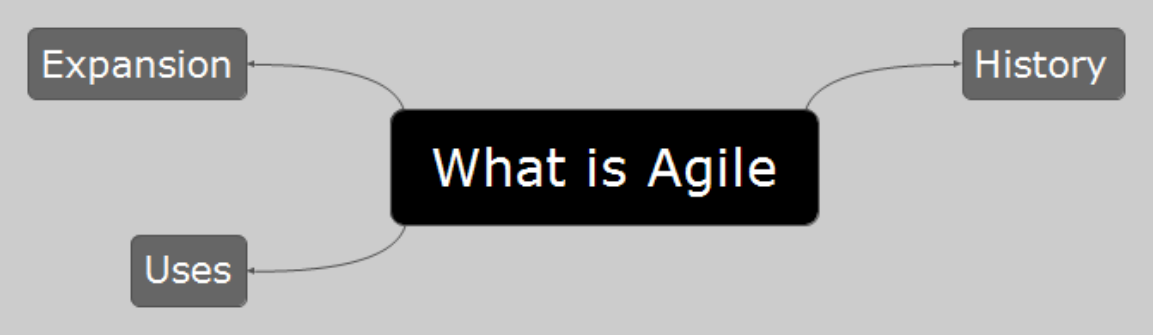
1. Agile: What is all the fuss about?

Today there are conferences, forums, professional coaches, expanded techniques, optimised agility, even specialist 'Agile' recruiters. So what is all the fuss about?

In this article I will take a look at the history of this type of software development technique, what the current incarnation is promising adopters and what are some of the truths from the front line. I will also highlight some of the elements 'Agile' struggles to deal with from a business perspective.

In my breakfast briefings I expand more on the things you will need to consider as an executive thinking about how 'Agile' would benefit your business and if you should adopt it. (Contact me to find out more about these events).

2. What is Agile



2.1. History

Incremental software development methods have been traced as far back as 1957at IBM's 'Service Bureau Corporation', which was superseded after a paper by E. A. Edmonds in 1974, introduced an adaptive software development process.

Concurrently and independently the same methods were developed and deployed by the New York Telephone Company's Systems Development Center under the direction of Mr Dan Gielan.

During the mid to late 1970s Mr. Gielan lectured extensively throughout the U.S. on this methodology, its practices, and its benefits.

So-called lightweight software development methods evolved in the mid-1990s as a reaction against heavyweight methods, which were characterized by their critics as a heavily regulated, regimented, micromanaged, waterfall model of development.

Many proponents of lightweight methods (and now agile methods) contend that they are a return to development practices from early in the history of software development.

Early implementations of lightweight methods include Scrum (1995), Crystal Clear, Extreme Programming (XP) (1996), Adaptive Software Development, Feature Driven Development, and Dynamic Systems Development Method (DSDM) (1995). These are now typically referred to as agile methodologies, after the Agile Manifesto published in 2001

2.2. Uses

Development methodologies such as 'Agile' are essentially tools for software developers to allow them to produce applications whilst keeping pace with change during the Software Development Life Cycle (SDLC) and thus avoid the pitfalls of waterfall methods where the requirements for the entire solution are specified in detail up front, only for the developers to find out after months, or years, of coding and testing that the business needs have moved on and requirements have changed.

The optimum use of 'Agile' is against small independent pieces of functionality, allowing the full process to be completed and signed off ('done') within the Sprint process. Larger, interrelated functions, especially ones which combine to form a single business process, create significant problems for the methodology and some of its underpinning principles.

These can be overcome but will lead to a 'Hybrid' form of 'Agile', and is a topic covered in my Breakfast Briefings.

2.3. Expansion outside the development boundaries

When project management became popular in the 1980's, the traditional approach of most project managers was what was known as a waterfall. A strict, rigid stepping from one stage to another in the process for gathering requirements, building the solution, testing the solution built and then putting this live.

Whilst this was a successful method for delivering benefit to the business, it soon came under scrutiny as businesses became more dynamic, reacting rapidly to changing needs. The limitations of the waterfall design were based ironically in its very strength; the strict control of the stages of a project.

What was needed was a way of 'trying out' solutions on the fly and making changes throughout the process. Prototyping became the way forward for many, but this was difficult for the traditional project methodology to manage.

Prototyping became a series of small waterfalls, which worked for a while but still had the inherent limitations associated with it.

Something new had to be designed, and the profession turned to the software development and engineering industries, and discovered 'Agile'. A cyclical set of activities called Sprints, during which an agreed set of requirements were designed, built and tested in a short time frame.

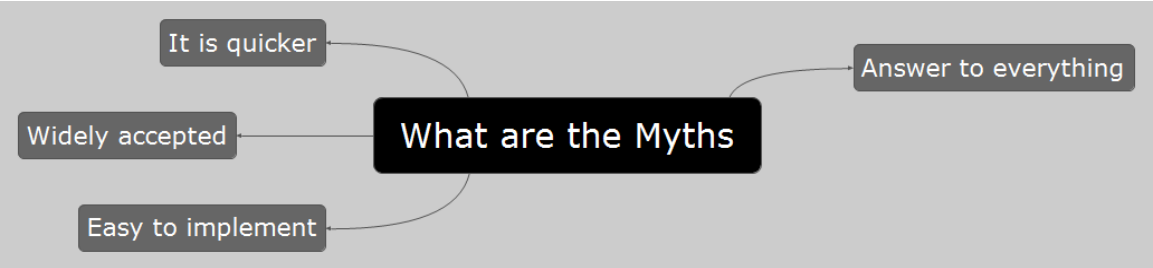
'Agile' methods break tasks into small increments with minimal dependencies and do not directly involve long-term planning. Iterations are short time frames (timeboxes) that typically last from one to four weeks. Each iteration involves a team working through a full software development cycle, including planning, requirements analysis, design, coding, unit testing, and acceptance testing when a working product is demonstrated to stakeholders. This minimizes overall risk and allows the project to adapt to changes quickly.

Stakeholders produce documentation as required. An iteration might not add enough functionality to warrant a market release, but the goal is to have an available internal release (with minimal bugs) at the end of each iteration. Multiple iterations might be required to release a product or new features to market.

All of this however can lead to a conflict with large scale end to end business processes, which will be discussed more a little later.

Interestingly, the idea of disciplines such as Project Management becoming 'Agile' is in the most part a jump on the bandwagon by the PM Industry because any 'old school' project manager will tell you they are already well versed in managing dynamic changes to requirements in a similar way to the 'Agile' development techniques, and the above only goes to substantiate that they are still fundamentally a series of waterfalls, albeit carried out in a more flexible and dynamic way.

3. The Myths



There are many 'Agile' evangelists who claim that going 'Agile' is the answer to all your problems. It isn't.

Now, I confess I am a big fan of incremental software development techniques, and 'Agile' is as good as any, but it can throw up as many problems as it professes to solve.

The claim from the governing body is that 'Agile' frameworks help companies accelerate time to market, increase productivity, and respond to changes in priorities.

Let's take a look at these along with the beliefs that it is easy to implement and is widely accepted.

3.1. It is quicker to market

This is based on the fact that each Sprint should produce a releasable product, although not always releasable externally. It means a product can be brought to market in stages, after each external release is made available, giving the appearance of a speedier market entry.

Assuming the products are individual and not dependent on other functions yet to be developed, the process of *definition through to delivery* under any project management technique would be similar, so not necessarily quicker.

Once the products become interrelated and dependent on other deliverables (stories) and combine to create a single large scale business process, the challenges begin to appear, including when external releases can realistically be made.

3.2. Increased productivity and more reactive to change

The benefit of 'Agile' (or Scrum) is its ability to adapt quickly to changes, thereby allowing the rapid reactionary development of needed functionality in iterative cycles, encouraging re-use and shared knowledge which makes the team more productive. Not in speeding up the development process.

Again, this is typically only true under optimum circumstances of small independent pieces of functionality and a mature and well managed 'Agile' team. This suffers once larger interrelated and dependent functions combine in order to satisfy wide ranging single business processes.

3.3. Easy to implement

Technically, 'Agile' is actually a simple process, and can be adopted easily within the development teams after the required training from either an 'Agile' coach or trained professional.

The problem is that unless the whole business is behaving in an 'Agile' way and feeding the process, you are not 'Agile', but a hybrid, which is actually what almost every implementation is.

Extending this approach across the traditional business disciples is far more challenging and unless close attention is paid to the psychology of the roles and individuals moving from a traditional to an 'Agile' environment, it more often than not, struggles to gain acceptance or achieve the claimed benefits.

3.4. Widely accepted

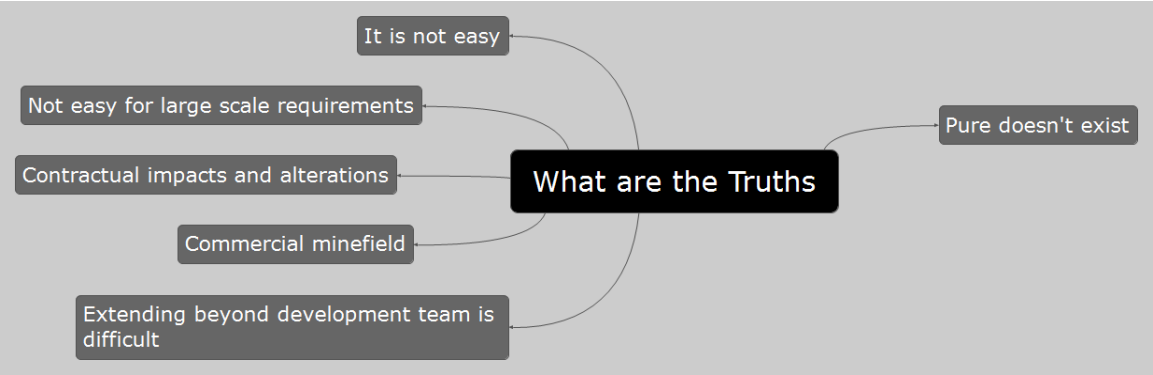
This claim is a little more subjective in nature and depends on your reference point. As a software development methodology, it is very widely accepted within the software industry, but as pointed out in the earlier section about the history of incremental software

development, it would be, because it is nothing new and possibly even a return to new incarnations of previous techniques.

The claim becomes a bit more clouded when you consider its adoption by non-technical and more corporate disciplines within business, especially within European businesses.

In some very large, European businesses, I have experienced my greatest challenges to acceptance of the concept, for a number of reasons, discussed in the next sections.

4. The Truths



4.1. Pure doesn't exist

Having spent over 30 years involved in projects and programmes often involving software development, including running a number of application development divisions for some household name companies, and being an avid fan of incremental techniques including 'Agile', I have yet to come across any organisation that has redefined itself into a 'pure' 'Agile' one.

More often than not, it is due to the immense challenges in the corporate and contractual disciplines that result in a hybrid approach, which can work very effectively if done correctly.

4.2. Extending beyond development team is difficult

When considering expanding 'Agile' outside of the technical environment, you have to bear in mind this is a technique created in the engineering world including software engineering or development as we now call it.

Extending it to corporate disciplines is difficult to say the least, and as mentioned, project managers by definition have to know how to react as quickly as possible to changes in the project requirements, and are already using techniques to do just that, albeit in a more structured and formal way.

4.3. Contractual impacts and alterations

One element of the challenge often not considered when transforming a business to be more 'Agile' is the structure and content of service contracts with your customers, current and future.

Most contracts are heavily based on the concept of defined deliverables, deliverable based payments, contract estimates based on requirements and standard waterfall based terms.

Creating contracts based on an 'Agile' approach where requirements are far less defined and definitely not static, becomes a challenge for the legal and procurement teams and needs specialist input from someone knowledgeable in 'Agile'.

4.4. Commercial minefield

When working for a very large European engineering company, one of the greatest challenges faced in making the software delivery and supporting services programme 'Agile' in approach was the question of re-work due to bugs in the deliverable and the commercial implications to both parties.

When confronted with a question of *"why would I pay for your errors in deliverables?"* after a manager had explained the process of creating a new User Story for the re-work and adding it into the product backlog for possible inclusion in the next sprint, I spent a long and often painful few months creating an acceptable process for both sides, whilst maintaining the client-supplier relationship.

As always, the result was a compromise and both parties had to accept some commercial impact, but the resulting hybrid approach was well received.

4.5. Not easy for large scale requirements

Consider the situation where a business process spans a number of business disciplines and a set of functionality in a large ERP is required to allow this single business process to complete.

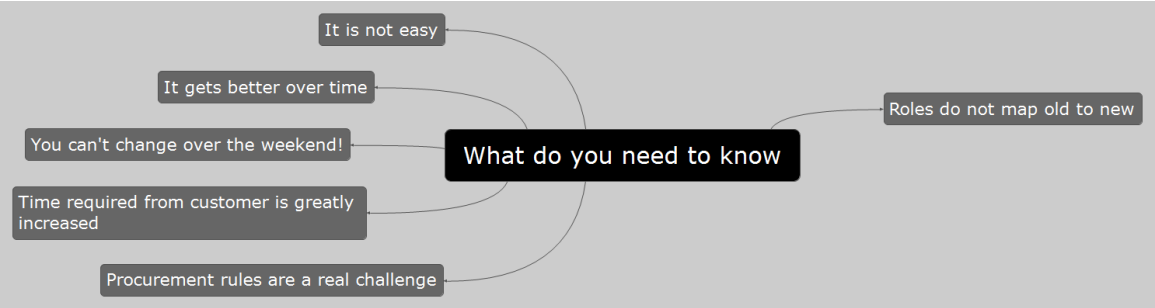
Each story (requirement) can still be managed in the 'Agile' process, however the business cannot sign off on the delivered release at the end of each Sprint, because they are unable to complete the user acceptance testing on the whole business process. They will not then be able to close out each Sprint as 'done' until the whole set of functionality is delivered and tested working together, end to end, which could be at the end of a large number of Sprints, maybe as a Release, or even at the end of the whole contract.

This calls for a creative way to work together in using the best parts of 'Agile' and maintaining some more traditional control on testing and sign off, which affects commercial staged payment contracts.

4.6. It is not easy

When you consider all the non-technical challenges involved in transforming your business into an 'Agile' one, the process stops being so **easy** and becomes a complex mix of methodologies, psychologies, disciplines and negotiations, that needs a very experienced director to navigate the pitfalls and not just an 'Agile' specialist who does not have a broad business acumen covering IT, procurement, legal, HR, commercial and executive relationship management.

5. Some things you need to know



5.1. Roles do not map old to new

One company I worked with could not understand why their switch to 'Agile' had not gone so well, even at the development team level.

For them, one of the fundamental issues was that they had assumed their project managers would, after training, become the ScrumMasters. They had not put any thought into the psychology of the two roles and the skills required by each. The fact that the two roles require very different characteristics and psychologies meant they ended up with people in the wrong roles, not only doing a less than acceptable job, but not enjoying their work either, causing many to leave.

You must consider your people and their match to the roles very carefully.

5.2. Procurement rules are a real challenge

Procurement has a remit to challenge costs, achieve savings and to control everything tightly within contract terms. You can imagine the reaction when faced with an 'Agile' contract which has no formally stated deliverable functionality, a changing landscape of resources and requirements and no tangible deliverables to associate payments to.

'Agile' requires a 'leap of faith' across the business, no more so than with traditional procurement teams who view this as a relinquishing of controls associated with often very large contractual sums of money. It is as much a journey for traditional business processes as for the development teams.

5.3. Time required from customer is greatly increased

In order for 'Agile' to work well and be as dynamic and reactive as it needs to be, the profile of involvement from the business owner, the business' counterpart to the 'Agile' Product Manager, becomes far less front and back loaded and more levelled throughout the lifecycle, which means this becomes much more of a day job than before. When the desired result is a complex set of interrelated functionality, often the businesses involvement is full time. Be prepared.

5.4. You can't change over the weekend!

This is a journey, as any true transformation is, and you cannot train people, even assuming you have done the due diligence described in section 5.1 above, one week and have then come in on the Monday as 'Agile'.

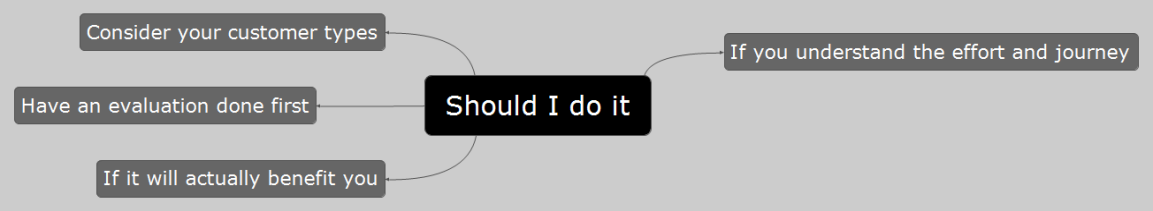
Engage a competent transformation director who knows and more importantly has experience of 'Agile' to plan it properly and create, direct and control the roadmap.

5.5. It gets better over time

Like most things, if you have done the planning, preparation, training, implementation and adoption right, the effectiveness of 'Agile', even in hybrid form, increases as everyone get more proficient, so stick with it.

Try to maintain your 'Agile' teams because their efficiency increases as they mature as a team and get to know each other's strengths and weaknesses, making their velocity (amount of work completed per Sprint) increase.

6. Should I do it



The answer to the question of whether you should consider taking your business into an 'Agile' environment is dependent on a number of considerations.

As a long term executive and advocate of such techniques as 'Agile', I would say yes, do it, but only under careful consideration.

6.1. If you understand the effort and journey

Make sure you do your due diligence and fully understand the effort of what is involved, what your specific journey will look like and how long it will take.

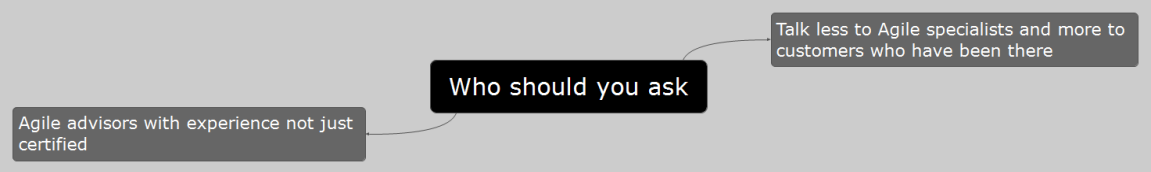
6.2. If it will actually benefit you

Have a knowledgeable and experienced consultant review the benefits for your company specifically to determine the ROI and timeline. As effective as 'Agile' is when done correctly, it is not for everyone and may not provide sufficient, if any, long term benefit.

6.3. Consider your customer types

Make sure the evaluation considers your customer profiles. If your customer base is predominantly large, traditional, bureaucratic, European based organisations the depth and scale of the challenges will be considerable for your commercial and legal teams.

7. Who should you ask



7.1. Talk less to 'Agile' specialists and more to customers who have been there

'Agile' specialist are a great source of information, however they can be too enthusiastic and not possess a broad enough business acumen to carry out a thorough evaluation of what it will mean for your business as a whole and the impacts across your customer and supplier base.

7.2. 'Agile' advisors with experience not just certified

Make sure any 'Agile' advisors are also experienced and not just certified. It is only through experience that they appreciate the finer nuances and limitations of 'Agile' and the ways in which it may need to be adjusted to suit your circumstances.

About the Author

Laurence Nicholson

Founder and Managing Director, N Cubed Consulting
United Kingdom

Laurence Nicholson is an Interim Executive focusing on Operations, Complex Programmes and Transformation Direction with an extremely broad and flexible professional background, extending across finance, procurement, pricing, operations, software development & techniques, consulting and PMO evaluation & governance. He is also a Certified Agile ScrumMaster .

He is experienced in operating in numerous industries within the private sector as well as in the public sector for local and central government including NHS. He has operated at CxO level, led consulting teams as department head, provided executive coaching to senior leaders in large global organisations, and contributes to global on-line Project Management resources, published in numerous languages.

His core skill is solving business' strategic and/or operational crises, turning around and transforming organisations' prospects, mainly through understanding their issues quickly from a 'business, technology and people' aspect, re-aligning diverging people, helping with potential cultural integration, and improving operational processes.

Above all, he has the insight to quickly diagnose challenges, the toughness to deal with them and the personality to build external and internal relationships that create sustainable solutions.

Mr. Nicholson can be contacted through the N Cubed Consulting website at www.ncubedconsultingltd.co.uk contact us page, or at laurence@ncubedconsultingltd.co.uk