NTT DATA

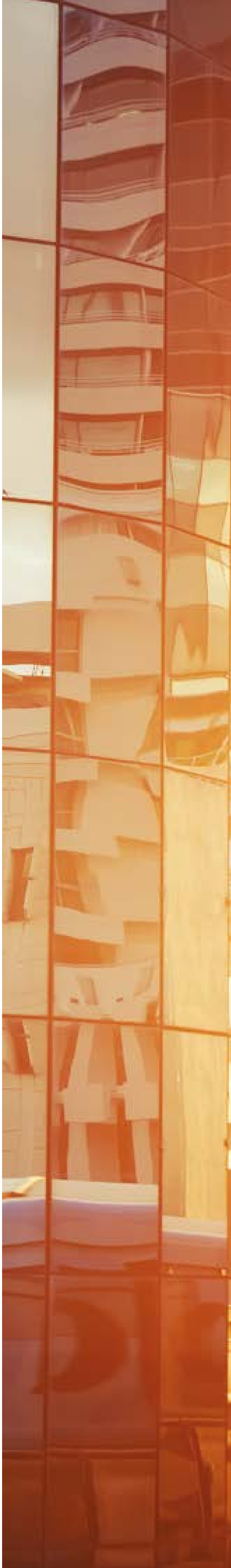
Scaled distributed agile delivery

How to power up agile across an organisation

uk.nttdata.com

素晴らしいガイド

GUIDING GREATNESS



Contents

Introduction

Disruption has become the norm. Technological innovation is delivering a new generation of more-capable, more-connected devices while the availability of previously expensive data creates a fertile environment for innovation.

Market entrants increasingly combine existing capabilities in new ways, disrupting established companies from Kodak and TomTom to Jawbone and Blockbuster. While many incumbents drag their heels, newcomers spot opportunities and adapt quickly to meet customer need.

Incumbents must cut costs and innovate rapidly to compete. The alternative is likely to be oblivion. It is expected that over half of S&P 500 companies will be replaced in the next 10 years[1], and the average lifespan of a company is already falling[2].

Agile delivery provides a means to mount the fightback, based as it is on principles of decentralised decision-making, empowerment, collaboration, feedback and constant improvement. Agile methodologies can deliver things faster and at higher quality than more traditional, waterfall methodologies[3].

Many companies have already adopted agile methodologies in some way. The challenge comes when organisations try to scale up their agile initiatives beyond a handful of teams. Further complexity arises when these teams, or parts or them, are distributed across more than one location where skills or cost advantages are being exploited.

This paper explores the benefits of adopting agile and how those benefits can be extended across the whole organisation by delivering agile at scale. The recommendations can be implemented stand-alone or alongside one of the many emerging enterprise scaling frameworks[4].

We also include some of the lessons learned from NTT DATA's experience of working with diverse clients implementing scaled agile.

The benefits of agile delivery

Adopting agile delivery is proven to deliver significant benefits compared to traditional sequential development[5]. While most of these benefits can be enjoyed to some extent even in small-scale implementations, the advantages typically grow as agile is deployed more widely.

Cost savings

While the primary focus of agile is on generating more high-value “bang for your buck” than it is on saving money, NTT DATA's experience confirms that cost savings can be a clear benefit in larger-scale agile implementations. The main savings accrue in cases where a distributed model allows a proportion of team members to be located in lower-cost regions. In one project, NTT DATA delivered cost savings of 24% through effective implementation of a nearshore, distributed scaled agile model.

Faster delivery

Many companies find that agile delivery results in faster development and shorter lead times[6]. For example, NTT DATA increased one company's velocity by 2% each sprint over the course of 20 two-week sprints. Average team velocity doubled in the first 18 months, despite the client flexing teams up and down to accomodate varying demand.

Predictability

Agile delivery requires careful management to achieve a steady output[7]. The longer the agile delivery mechanism is running, the more accurately organisations can estimate and deliver results at a sustainable pace. This approach increases predictability and enhances trust between stakeholders.

NTT DATA has achieved high predictability of between 80 and 100% of anticipated outcomes on average. If continuous deployment is accompanied by regular releases of new features, then value can be provided even earlier[8]. Many organisations now “release on demand”, resulting in value being realised as early as possible[9].

Ability to react to the market

A key driver for many companies is to get complex products to market quicker[10]. Minimising the time from idea to release is at the heart of lean practices, many of which are incorporated into agile thinking.

In contrast, sequential delivery can result in significantly longer lead times, which can mean the product is no longer relevant or even obselete when released.

Agile delivers high-value features in small increments, so market reaction can be tested earlier. Furthermore, the close alignment between business and delivery also makes it easier to prepare for frequent product launches.

Enhanced quality

Some 43% of organisations say they have adopted agile to improve software quality[11]. Scaled agile delivery improves quality by using small teams and short iterations, which lead to shorter feedback loops[12]. Frameworks such as LeSS (Large Scale Scrum), DAD (Disciplined Agile Delivery) and SAFe (The Scaled Agile Framework) also promote quality as a reason to adopt scaled agile[13].

Improved visibility and transparency

The whole business needs to work closely and collaborate with the agile teams in order to successfully drive value. The business must fill the backlog with requirements and sign them off at the sprint demos to maintain engagement throughout the process and prevent deviation from requirements.

Improved software maintenance and stability

Agile delivery focuses on product lifecycle, not project lifecycle. This aligns long-lasting teams to products promoting greater 'life-cycle' thinking. The same team will be doing the initial development and then go on to develop enhancements, fix bugs and ultimately decommission the product. This preserves knowledge throughout the lifespan of the product[7]. Furthermore, since releases are carried out in small increments, platforms are more stable, and fixes can be applied more often[14].

Improved team morale

Agility is as much about mind-set and culture as it is about a methodology or framework[15]. Agile's emphasis on autonomy, mastery and purpose improves job satisfaction and employee motivation[16], with teams finding a sense of ownership and meaning in what they are delivering[17].

Finally, NTT DATA recommends adopting an approach that balances the use of feature teams and component teams. Feature teams deliver capabilities that matter to an end user, for example a 'change address' feature. Component teams are established when the code being written can be re-used many times by other teams in order to build their features and requires specialist knowledge to create. Finding the right balance is key to successful scaling.

How to scale in a distributed way

Once the basics are in place for agile delivery, it can be scaled up beyond a few teams. Successful scaling follows the crawl-walk-run pattern. Start small, proving concepts and reusing insights gained along the way to define what the team will look like at scale. The following steps will keep the roll-out moving in the right direction.

01.
Managing requirements

Scaling, in a distributed way, requires careful consideration of the product owner function. Where team members are co-located this is not a major issue. However, in cases where teams are distributed, things become more complex.

In NTT DATA's experience ensuring all are clear on the distinction between product manager and product owner is critical. The product manager has several key jobs:

- Define the product strategy and roadmap
- Oversee the execution of strategy via delivery teams
- Own stakeholder relationships, both across the business and with suppliers.

The product owner role is more closely focused on the execution of the road map. They work with a small number of scrum teams to guide delivery. They also have several key jobs:

- Own the delivery of a discrete part of the product roadmap
- Represent the business need within the delivery team and transform these ideas into more refined concepts that can achieve their tar

In a distributed model, it is more than likely the scrum teams will be located nearshore or offshore. Since team access to a product owner is key, a proxy PO should be agreed and located alongside the team. In our experience, a team member with business analyst skills is often best placed to perform this role. Key tasks include:

- Elaborate the concepts to create detailed demand requirements for the development team
- Facilitate design sessions with the implementation team to define the solution.

These roles are responsible for taking each user story from an idea, through concept into the demand funnel, ultimately feeding into planning via structured stages.

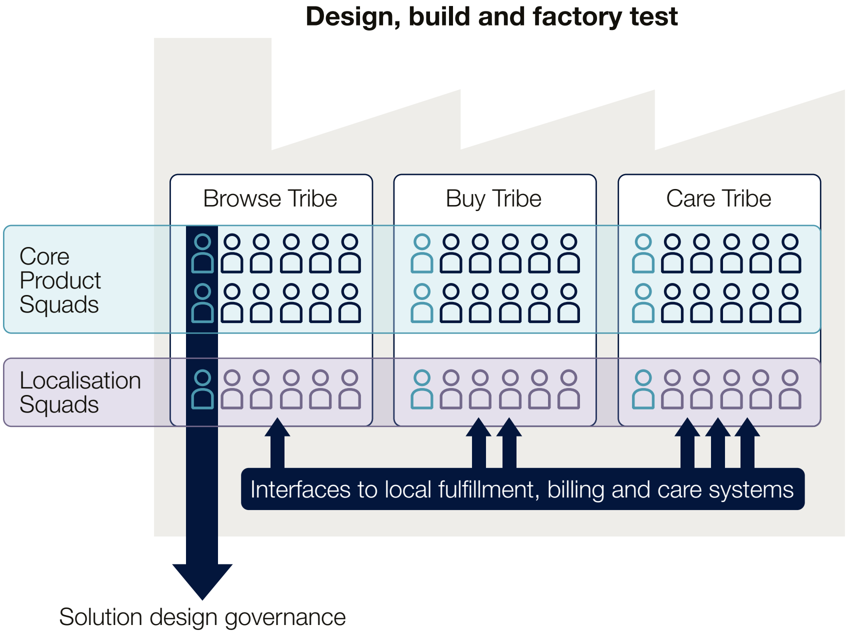
02.
Create an organising framework to house additional teams

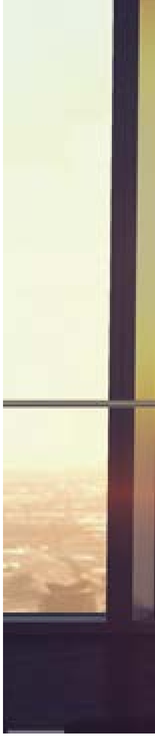
Frameworks such as SAFe suit very large-scale agile implementations and include their own language and organising concepts, which can be helpful. However, if the approach to distributed scaling is not using such a formalised approach, then an alternative model NTT DATA has found to be useful is an adaption of the work pioneered by Spotify.

In this approach, autonomous scrum teams are assigned to a specific part of the customer lifecycle. Where scaling requires multiple teams to work on the same aspect of the customer lifecycle these are grouped into 'tribes', with each covering an end-to-end function such as 'Browse', 'Buy' or 'Care'. Each tribe has multiple squads that work together to deliver the end product.

Know-how is shared across teams through the creation of chapters organised by functional discipline. For example, technical assurance and component re-use is driven by the 'design chapter', comprising a developer from each team responsible for sharing knowledge with other teams.

A third, and often key, dimension in more complex client environments is the split between core product and local market extensions and integrations. Here team knowledge of the local market domain is critical. Yet to be successful, these teams need to have a thorough understanding of the product area they are integrating. For this reason, we recommend having dedicated localisation teams aligned with each tribe.


Design, build and factory test
Browse Tribe
Buy Tribe
Care Tribe
Core Product Squads
Localisation Squads
Interfaces to local fulfillment, billing and care systems
Solution design governance



03.
Synchronise planning across domains and teams

As the delivery organisation gets bigger and more distributed it is critical to synchronise cross-domain planning. Having already established a cadence to turn unpredictable events into predictable ones, synchronisation creates an opportunity for cross-domain trade-offs to be agreed and establishes routine dependency management. It aligns everyone to a common goal.

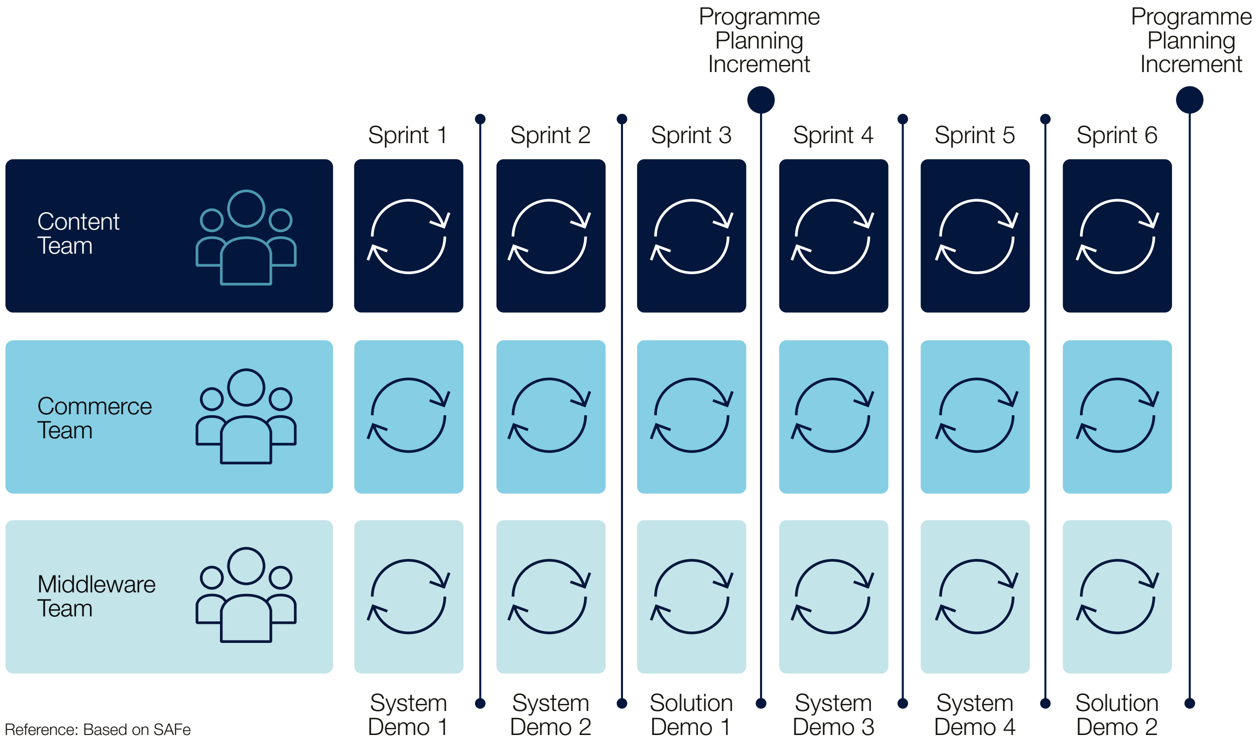
There are different ways to achieve this, ranging from the highly structured Programme Increment (PI) Planning approach advocated by SAFe to simpler joint sprint planning events. The key is to have participants actively involved and co-located, as far as is possible, for the planning sessions.

04.
Optimise the system for flow

The fourth success factor is to optimise the system to deliver a continuous and sustainable throughput of value. Whilst this is critical for any successful agile delivery, the consequences of poor flow are amplified at scale.

Optimising a complex system for flow requires fine tuning beyond the scope of this paper. However, as a priority we strongly recommend focusing on visualising and limiting work-in-progress. High utilisation will rapidly reduce the flow of value. The first step is to use a common tool across the scaled, distributed delivery team to record and visualise work in progress so that the issue can be actively managed. The resulting transparency means limits on work in progress can be implemented quickly to increase overall system flow.

Establish cadence and synchronise cross domain planning



Reference: Based on SAFe

05.
Plan team role distribution with care

Co-located teams, whilst favourable, are often impossible at scale, where cost and skill constraints play a significant role in the availability of the right team members close to home. Unless managed carefully, geographical team distribution has a major influence on productivity.

Our preferred model sees the business stakeholders, product owner, architects and UX located centrally, with scrum masters, developers and testers remote. This model has proven to be very successful but relies on establishing an additional new ‘proxy product owner’ (proxy PO) located with the remote team. To be successful, the product owner must regularly visit the remote team and build a solid rapport and shared vision with the proxy PO so that the proxy PO can genuinely act as the product owner for most day-to-day decisions.

We see some models where the scrum master role is located onshore rather than remotely. This feels familiar to those accustomed to having an onshore PM managing a remote delivery team. However, in our experience this is far from optimal since the majority of the work performed, and thus the impediments encountered, will relate directly to the work of the remote team. Locating the scrum master remotely leads to more effective team performance.

06.
Measure to manage

It’s impossible to manage what you cannot measure, so there are some minimum requirements for what to measure to ensure the organisation is working in the right way to deliver the product effectively. These include:

1. Measuring velocity to track the overall system output
2. Measure cycle time to check the speed at which work flows through each step of the system and thus identify bottlenecks
3. Measure the overall flow time to assess how rapidly value is being delivered, from the time approval is given to the time the feature is working in production
4. Measure the overall flow distribution to check that prioritisation decisions reflect the strategic business imperatives
5. Measure defect leakage to assess the extent to which quality is being built rather than tested in
6. Measure predictability to assess how reliable the team is at estimating the quantity of work they can complete.

NTT DATA recommends underpinning the standard Plan, Do, Check, Act cycle with an integrated toolchain so that important data is easy for everyone to access and key metrics can be easily derived and shared with all team members.

Meeting the challenges to successfully establish and scale distributed agile delivery

Experience shows that there is a consistent pattern in the sort of challenges raised during scaled agile implementations. There are also reliable ways to overcome those challenges.

Break down silos

First, domain silos must be broken down to keep hand-offs to a minimum. It is hard to make silos disappear completely. Instead, consolidate accountability and task a single organisation with end-to-end delivery so that one team is responsible for minimising any impact.

Cadence-based objective setting and planning can create common goals. DevOps also provides a useful way to erode barriers between development and operations. If true DevOps is not possible, bring Operations informally into any discussion of requirements.

Effective team management is critical

Next, recognise that team interaction needs careful management to promote collaboration. For example, maximise the overlapping time between distributed teams in different time zones, because collaboration is much harder with a five-hour time difference or more.

In addition, value onshore roles and maintain a sensible distribution of onshore and remote resources. Agile delivery using 10 to 15% onshore resource will likely fail, so work towards having 30% of the team onshore in long-term roles.

In addition, preserve team integrity as far as possible, since this is the lever used to drive accountability. Don’t break open teams to access individual resources. Instead use the backlog to prioritise work and hold teams to account.

Manage for outcomes, not activities

Agile delivery is about managing for outcomes, not activities. But it will take time to develop the new habits and embed the necessary cultural shift to prevent the natural fallback toward a default of activity management. For example, avoid the T&M (time and materials) management trap. If transitioning from an internally managed T&M programme, there will be a natural tendency for the client to want to direct activities rather than outputs.

Similarly, exploit the intrinsic motivation of knowledge workers. It is not possible to micro-manage knowledge workers who, by definition, know more than their bosses. Instead, articulate a vision, set objectives and implement outcome measures.

Similarly, recognise that projects will tend to eclipse product lifecycle thinking – at least in the early days, when organisational ‘muscle memory’ makes it likely that many parts of the organisation will continue to think in terms of projects with fixed scope and start-stop-start inefficiencies.

Finance needs to focus on results

Finance traditionally likes to approve fixed-scope not fixed-capacity or product-based funding. It's therefore crucial to actively encourage a shift in mind-set.

Most organisations build budgets around programmes and projects, rather than products. However, projects rarely consider the full lifecycle (maintainability, enhancements, decommissioning and future migration), so project-based costs are often unrealistic. In addition, customers do not interact with projects. Their perceptions are shaped by the experience of the products they consume.

Recognise that project stage-gate boundaries are not the only way to apportion operating costs. Use toolchain activity tracking to determine capital spend.

Build an integrated roadmap

In our experience road map planning is often not integrated. Demand for features is discussed in isolation from the supply of capacity. Often this is reactive. This inevitably leads to a 'bow wave' of requirements that demand unrealistically rapid ramp up of resource and results in lower productivity. Reactive planning can also result in under-utilisation during the resulting troughs, when short-term demand dries up. Instead introduce constant roadmap planning incorporating both supply- and demand-side management and set aside time for structured forwarded planning sessions.

Base the roadmap on 60% team capacity. Don't dedicate 100% effort to the roadmap because unanticipated work will inevitably arise. Also, avoid a large backlog of committed work because this becomes a queue. Remember Little's law: wait time is average queue length divided by processing time.

Avoid overlapping product management responsibilities, which can create conflicting demand in the same product area. Accountability for benefits realisation and budget management should determine content authority.

Prioritise platform stability over new features

Another common mistake is to prioritise new features over platform stability. Features are seductive but maintaining platform stability is more important.

Avoid prioritising new feature development to the exclusion of resolving technical debt. Failing to carry out this work is a false economy because not resolving architectural or code constraints today will significantly impact feature development time in the future. Instead, dedicate 20% of sprint capacity to resolving technical debt and monitor the sprint backlog to ensure product owners are making it a priority. In addition, dedicate 20% of sprint capacity to 'should have' optimisation work.

In other words, it's important to plan technical debt and optimisation into the roadmap. If using programme increments, dedicate some of the sprints to technical debt. Avoid measuring a product team's performance solely on features and benefits. Instead include platform stability in the targets for product management.

Leadership needs to drive a change in culture

Agile is all about changing hearts and minds. If it is not advocated from the top down it will not be sustainable. Leadership and management must understand the reasons for adopting scaled agile delivery, including the benefits and risks, as well as learn to understand what the metrics say about the health of the delivery system.

Leaders must shift from a traditional, hierarchical command-and-control style to a 'servant leadership' model. This puts the focus on the needs of the customer and the delivery teams. Leadership also plays a fundamental role in ensuring the delivery teams do not face impediments originating from external teams such as HR, procurement and finance.

Don't run before you can walk

Remember the crawl-walk-run approach to scaling up. The basics need to be adopted and done well before scaling so that the need to re-work does not reduce productivity or increase waste as the work progresses.

Agile must be understood by all teams and the whole organisation before it is scaled, or bad behaviours will take root. Metrics take time to capture and teams take time to be fully productive, so some aspects cannot be rushed.

Conclusion

The benefits of adopting agile are clear. It is understood that when implemented correctly agile delivery increases productivity, quality and predictability while reducing the time for products to get to market.

Challenges emerge when organisations try to scale these practices across distributed locations without recognising the need to take a different approach. This is particularly important in the absence of scaling frameworks such as SAFe. Here the temptation is to assume large, distributed agile is just more of the same. This is never the case.

To exploit the benefits associated with scaled, distributed delivery – lower unit costs and greater output – care must be taken. NTT DATA's experience shows that the following key steps should be taken in order to achieve effective distributed scaled agile delivery:

1. Do the basics well
2. Scale gradually
3. Enhance requirements management
4. Create an organising framework to house additional teams
5. Synchronise planning across domains and teams
6. Optimise systems for flow
7. Plan team role distribution with care
8. Measure to manage

NTT DATA can help clients establish efficient agile delivery teams and design how they operate within the wider organisation.

NTT DATA can also help clients leverge the cost advantages of distributed agile teams whilst maintaining high levels of engagement and efficient delivery.

For more information, or to speak to our Enterprise Agility Practice, please contact Henrietta Marsh-Smith or Jason Ford.

Henrietta Marsh-Smith
Practice Lead
Enterprise Agility

+44 7583 701 324

henrietta.marsh-smith@
nttdata.com

Jason Ford
Vice-President
Advisory Services

+44 7776 184 015

jason.ford@nttdata.com

References

1. Scott D. Anthony, S, Patrick Vinguerie, Evan I. Schwarts, and John Van Landeghem. 2018 Corporate Longevity Forecast: Creative Destruction is Accelerating. [Online] 2018. https://www.innosight.com/insight/creative-destruction/

2. Huron. Innosight: Strategy and Innovation. [Online] [Cited: 09 16, 2019.] https://www.innosight.com/insight/creative-destruction/

3. Move Away From Waterfall to Agile and Product-Centric Delivery Methods. Wilson, Hotle and. 2018, Gartner.

4. More detail on the range of agile scaling frameworks can be found in Gartner, September 219 'Market Guide for Enterprise Agile Frameworks'.

5. The Standish Group's Chaos Studies Report, 2017 examined 50,000 projects implemented between 2013-17 and found that the likelihood of a successful delivery using agile methods was 42% versus 26% for projects delivered using a waterfall approach.

6. Agile software development in large organisations. Lindvall, M., et al. 12, s.l. : IEEE (Computer), 2004, Vol. 37. 10.1109/MC.2004.231.

7. Kersten, Mik. Project to Product: How to survive and strive in the age of digital disruption with the flow framework. Portland, OR : IT Revolution, 2018.

8. Transforming a Six Month Release Cycle to Continuous Flow. Marschall, Matthias. Washington, DC, USA : IEEE, 2007. 10.1109/AGILE.2007.64.

9. Forsgren, Nicola & Humble, Jes. Accelerate: The science of Lean Software and DevOps: Building and Scaling High Performing Technology Organisations. Portland, Oregon : IT Revolution, 2018.

10. Towards theory building in agile manufacturing strategies - Case Studies of an agility taxonomy. Zhang, D. Z. s.l. : International Journal of Production Economics, 2011.

11. 13th Annual State of Agile Report. stateofagile.com. 2019

12. Operations Management. Slack, N., Chambers, S., Johnston, R. s.l. : Harlow: Pearson Education, 2010.

13. A review of Scaling Agile Methods in Large Software Development. Alqudah, M., Rasali, R. 6, s.l. : International Journal on Advanced Science Engineering Information Technology, 2016, Vol. 6. 2088-5334.

14. Architectural Tactics to support rapid and agile stability. F Buchmann, RL Nord, I Osakaya. s.l. : Cross Talk, 2012.

15. Agile Methods for Engineering. Choudhury, Islam. s.l. : Management for Scientists, 2019. 978-1-78769-204-6.

16. "The paradox of self-management: Individual and group autonomy in work groups. Langfred, C. W. s.l. : Journal of Organisational Behavior, 2000.

17. Team Autonomy in Large-Scale Agile. Moe, Nils Brede Dahl, Bjørn Stray, Viktoria Karlsen, Lina Sund Schjødt-Osmo, Stine. s.l. : Proceedings of the 52nd Hawaii International Conference on System Sciences, 2019.

NTT DATA UK
1 Royal Exchange
London
EC3V 3DG
020 7220 9200

NTT DATA is a leading consulting and IT services provider, combining global reach with local expertise in over 50 countries. Whether it's business transformation, enabled by digital, data and technologies, safeguarding against security breaches, improving operational efficiency or driving new revenue streams, our vision as the Trusted Global Innovator can help organisations navigate the ever-changing digital landscape and deliver outstanding results.

NTT DATA offers a portfolio of best-in-class consulting services and innovative enterprise solutions tailored to suit the entire life cycle of IT investment. Supported by our international Centres of Excellence, our team of local experts can deliver on a wide range of services from transformation to agile development and intelligent automation for industries across manufacturing / automotive, banking, insurance, telecommunications, media and public services.

For more information about NTT DATA please visit uk.nttdata.com