Deloitte.

Scaling agile at financial institutions
Lessons from the trenches
<table>
<thead>
<tr>
<th>1</th>
<th>Agile approaches to meeting new challenges</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>Keys to scaling agile</td>
</tr>
<tr>
<td>3</td>
<td>Governance and execution</td>
</tr>
<tr>
<td>9</td>
<td>Governance strategy</td>
</tr>
<tr>
<td>9</td>
<td>Execution strategy</td>
</tr>
<tr>
<td>12</td>
<td>Adapting agile</td>
</tr>
</tbody>
</table>
Many large financial services companies face mounting pressure from new, nontraditional competitors. The most disruptive of these newcomers develop and launch multiple innovations, identify the winners, and then grow at speeds generally hard to match by major institutions.

Relative to waterfall development approaches, agile has demonstrated its merits and can accelerate speed-to-market, enhance quality, and increase flexibility while reducing costs and complexity. Various agile methods employ various approaches to implementation, but share the following characteristics:

• Delivering tested software in short, specifically timed iterations
• Accepting and even encouraging changes in requirements between iterations
• Mandating client involvement during each iteration

However, agile methods that work for start-ups, digital design agencies, and mobile app developers do not directly scale to the purposes of large organizations developing complex products. Those methods require significant changes to meet the needs of large institutions with their legacy systems, external partners, massive footprints, and extensive compliance requirements.

Lessons learned
Deloitte has assisted a number of major financial institutions in deploying agile at scale and has identified a number of leading practices from the successes of its clients. These projects, among the first of their kind, aim to bring innovative products to market at speed, simultaneously, across multiple locations. Challenges have included regulatory demands, external-system interfaces, documentation needs, and project size, with the latter involving as many as six dozen teams and 500 individuals.

This paper, prepared for chief information officers, key IT program leaders, and other transformational stakeholders, begins with observations on scaling agile and then moves to the lessons learned in these projects. Each section closes with a brief “Case in point” illustrating how project teams addressed the issue.

This paper assumes basic familiarity with agile development concepts and terminology. For further information, also see the Deloitte paper Don’t fear change, embrace it: Advancing the case for agile methods in systems integration.¹

¹ Don’t fear change, embrace it: Advancing the case for agile methods in systems integration, Deloitte Development LLC., 2011
Keys to scaling agile

Traditional agile is designed for small teams working on well-defined problems over short periods. Scaling agile to projects with dozens of teams working on complex products over multiyear horizons clearly presents challenges. Therefore, in order to scale agile to the needs of a large organization, the following are critical:

1. **Use multiyear road maps:** Initiatives in large organizations often involve multiple projects with long-time horizons. Agile implementations for such initiatives should conform to a long-term road map while maintaining basic principles of agile.

2. **Strengthen governance structures:** Large-scale implementations involve multiple scrum teams from disparate functions under different leaders, often in dispersed locations. In scaling agile, additional roles should be established to facilitate communication and resolve conflicts.

3. **Address cross-team dependencies:** Often a team requires completed components from another team before proceeding further. Such cross-team dependencies create the need for additional coordination when planning upcoming iterations.

4. **Consolidate reporting and tracking:** In traditional agile, progress can be tracked via burn-down charts or other reports from agile tools, such as Rally or VersionOne. In large-scale agile, team burn-down rates should be combined and fitted into the project-level plans to report overall progress.

5. **Increase end-to-end quality assurance:** Traditional agile relies on functional and unit testing within sprints as well as other quality control processes. In agile at scale, limits on how frequently code can be integrated create a need for dedicated end-to-end (E2E) testing before releasing code to production.

Elements of agile can be adapted to larger financial services organizations, but the organization should also adapt to agile. For example, organizations accustomed to the more sequential waterfall method should prepare to adjust to the highly interactive and iterative approaches of agile.

Agile can be integrated into existing development processes, and planning, control, and documentation issues can—as this paper demonstrates—be addressed.
Governance and execution

Successful scaling of agile depends mainly on effective governance strategy and sound execution tactics.

Effective governance calls for alignment of executive sponsorship across functions, broad agreement on the business case and desired business value, and acknowledgement of the potential benefits and the limitations of agile. A commitment to data-driven project management fosters the right cadence and atmosphere for the project. That commitment supports governance that relies on facts rather than opinions, and promotes honesty, openness, and collegiality across the various teams.

Governance strategy for agile at scale encompasses the following activities:

- Agile program structuring
- Programmatic backlog management
- Systematic reporting and dashboards
- Integrated quality management
- Organizational coexistence

Also, teams need realistic expectations regarding deliverables, testing, and documentation. Tolerance for the iterative nature of agile does not come naturally. However, training, mentoring, coaching, pilot projects, and hybrid or gradual adoption can help organizations develop that tolerance.

Sound execution tactics include the following activities:

- Enhanced scrum roles
- Business process alignment
- Development and release management

The remainder of this document provides an overview of each of these governance and execution activities.

**Governance strategy: Structuring the agile program**

Cross-team delivery governance constitutes the heart of the agile scaling process. Therefore, sound program structure and clear roles and responsibilities are important.

A scaled agile program differs from a traditional agile structure in several ways (see Figure 1).

**Figure 1: Agile at scale versus traditional agile program structure**

![Diagram showing the comparison of agile at scale versus traditional agile program structure](image-url)
Essentially, agile at scale calls for adding some new roles and enhancing traditional roles to promote communication and coordination. For example, a sponsor and a project management office (PMO) head up the overall project, with input and coordination with the steering team comprising the business leader, product leader, and technology leader. Each initiative also requires a workstream lead (WL), an agile delivery manager (ADM), and a chief product owner (CPO), as well as ongoing involvement of business stakeholders.

Figure 2 further details characteristics of roles in agile at scale.

**Figure 2: New and modified roles**

Case in point #1: New and modified roles for agile at scale

While implementing scaled agile within its development teams, a leading credit card issuer modified some traditional agile roles (e.g., scrum master, PO) and established a few new ones like CPO, WL, and ADM to improve efficiencies in the program.

- The CPO worked with the business stakeholders to create the epics.
- The PO detailed the epics, which were reviewed by the tech lead and architect, and broken down by the development team into tasks to be completed in one sprint cycle.
- The WL updated the PMO on the progress and status of the initiative.
- The ADM maintained alignment between the roadmap and development work, coordinated reporting and analysis, and facilitated use of agile tools and end-to-end testing.
- The PMO addressed controlling to budget and schedule and coordinated the entire project and its partners and stakeholders.
Governance strategy: Programmatic backlog management

Roadmap planning sets the stage not only for project-level planning and epic story prioritization, but also for sprint-level backlog planning and periodic reviews (see Figure 3). As such, roadmap planning is the first step in programmatic backlog management.

Figure 3: Overview of agile planning process

Creating and managing the roadmap

Business leadership typically outlines the key priorities and milestones in a multiyear product roadmap. Then the team-leads for solutions architecture, platform, IT, and business units translate the roadmap into bodies of work to be delivered quarterly. The following steps assist in planning a roadmap and translating it into agile delivery items:

1. Initiative prioritization: This step considers business criticality, revenue impact, delivery deadlines, and compliance and legal mandates, and other factors, to develop tiers of priority for each initiative. Senior leadership then develops a composite score and ranks each initiative.

2. Initiative-level planning sessions: In working sessions, each initiative is divided into level 1 and 2 (L1, L2) epic stories. In planning sessions, higher-level sizing on each L2 epic gauges the effort needed from various scrum teams to complete the epic. These sessions are attended by POs, AOMs, architects, WLs, scrum masters, and PMO personnel.

3. Resource capacity analysis and allocation: Based on the resourcing estimates for each initiative, program management performs a capacity analysis to gauge the requirements of each development group and the estimated utilization level of all the scrum teams for upcoming quarters. If the requirements exceed the capacity for any scrum team, then resource conflict can be resolved by CPOs using the scores developed in step 1 above.

4. Sprint definitions: Based on the inputs from capacity analysis and initiative-level planning, L2 epics are broken into executable stories and assigned to sprints. This generates the milestones for completion of these epics, in terms of sprint number and delivery dates.

5. Periodic roadmap reviews: At predefined intervals, such as quarterly or biannually, review sessions are conducted to assess progress and adjust forthcoming execution in response to changes in priorities or lags in execution.

Case in point #2: Creating and managing the roadmap

A global financial services company successfully transformed its 1,500-member delivery team to agile. At an off-site meeting, the business unit leaders from across the organization evaluated a range of products and initiatives for business value and expected benefits. They also defined success metrics, outlined program scope, identified risks, developed key resource and funding assumptions, and defined a high-level implementation roadmap.

This meeting achieved consensus among the business leaders on business and product priorities and resource commitments—and on the expected course of development implied by the use of agile. The expected business value drove prioritization and resourcing of the initiatives.
Governance strategy: Systematic reporting and dashboards
In a large-scale environment, agile dashboard metrics enable program management to track progress and performance across multiple scrums and obtain a consolidated view of the program. This can provide the data on which progress is measured and resources are allocated (see Figure 4).

Figure 4: Agile dashboard metrics

Case in point #3: Tracking progress and performance
A major payment services company successfully executed a multi year agile transformation project involving more than 100 scrum teams. The project held release planning meetings every two months to discuss the priorities for the next three or four sprints. This allowed the scrum teams to work with the big picture in mind while also driving their user-story prioritization.

In this activity, the teams worked to set reasonable expectations regarding which features would be developed in which period and how interdependencies would affect the development and timing of features.

Agile progress metrics can provide data on work accomplished to date and on the comparative performance of scrum teams. Agile predictability metrics can provide data on the work committed to and on upcoming required resources. The key dashboard metrics are as follows:

- **Sprint velocity** compiles the number of story points completed in one sprint; this provides an historic or cross-scrum view of performance.
- **Burn-down charts** depict the number of remaining story points or hours over a period; these provide a view of progress at the sprint, release, or epic level.
- **Say/do ratio** compares the number of story points committed to (Say) with the number completed (do); this indicates the level of resources to allocate to various teams and tasks.

These metrics assist the leaders and teams in sizing and managing the backlog.

Backlog sizing
Architects assist in estimating L2 epics, so resources can be allocated and delivery timelines can be set. POs work with the business stakeholders to groom the epics and add details on functionality and acceptance criteria. POs then work with the tech leads to break down the epics into stories that are then sized in story points by the tech leads. This provides an estimate of the number of stories to consider in sprint planning. The agile sizing formula can be scaled to large-scale program needs.

Managing backlog
It is essential to standardize sizing and user story structure across teams. Backlog and story prioritization is driven by the roadmap and business-case inputs, and continually evolves. The key difference in agile at scale is defining common standards for story structure and sizing across all teams.
Case in point #4: Integrated quality management

A major financial institution established agile methodology across multiple teams to adopt a new payment processing system. As part of this effort, it set up an E2E testing team that tested the entire life cycle of the product—from merchant interface, transaction routing, payment authorization, near real-time payment capture, buyer notification, and post processing—to ensure that components developed by different teams functioned seamlessly when released.

Once the code was developed and unit-tested by individual scrum teams, the various code components were integrated in a dedicated environment where the E2E team was tasked with performing integration, function, and performance testing. They achieved this through automation scripts that allowed them to send multiple transactions through the merged code and then follow it up with manual validations.
Managing compliance

Financial organizations in particular should build in regulatory compliance iteratively in agile rather than adding it in a bolted-on manner or trying to address it after development. Building in compliance can reduce costs and increase the effectiveness of control and reporting.

Legal and compliance experts should be part of grooming sessions and end-of-sprint demos with scrum teams. This helps to ensure that the solution complies with all applicable regulations, including those related to data protection, geographical location of IT infrastructure, antifraud/antimoney laundering, or consumer dispute resolution, among others.

Meeting compliance standards may require establishing role-based access, creating secure zones for certain data, or avoiding comingling of data, as well as developing reporting for compliance.

Case in point #5: Managing traditional processes and non-agile partners

When a financial institution sought to globalize its mobile payments platform through a large-scale agile implementation, the agile teams involved compliance and legal experts from the different geographies in planning sessions, developing user stories, and setting priorities. Controls and reporting capabilities were built in iteratively during development.

This also allowed the teams to develop exception reporting and key performance indicators (KPIs) to ensure compliance with applicable regulations, proactive remediation of potential regulatory issues, and rapid response to actual issues.

Governance strategy: Organizational coexistence

In any organization there will be groups that do not use or understand agile, and agile tools will likely never be perfect. In addition, customers may want firm commitments, business and technology priorities may be unaligned, and legacy systems can undermine continuous integration.

Even so, agile has the potential to offer speed, savings, and quality. Realizing those potential benefits demands that organizations with traditional processes, legacy systems, and non-agile partners create ways of collaborating from the outset (see Figure 6).

Figure 6: Agile development and traditional processes can coexist

Leading practices for coexistence between traditional processes and agile include:

- Understanding that certain areas do not operate as agile; these include legacy systems, organizational structures (e.g., compliance, finance), and most vendors and partners
- Factoring in interdependencies between non-agile business processes and feature/product delivery timelines from the outset
- Embedding compliance and legal requirements into product design and development (see sidebar)
- Involving all relevant areas of the business so as to promote collaboration, feedback, and risk mitigation
- Providing transparency to all stakeholders
Execution tactics: Enhanced scrum roles

To support the increased coordination needed in agile execution, the PMO and scrum master roles and responsibilities are enhanced in specific ways, as indicated in Figure 7.

Figure 7: Coordinating agile development

The PMO establishes the long-term program vision and:
- Coordinates with the WL to implement the long-term roadmap and strategy
- Works with the ADM to publish key milestones and E2E program status
- Addresses risk and issue management

The PMO can also be a change agent that fits agile projects into the larger portfolio, promulgates practices that work, and trims those that do not add value.

The scrum master facilitates development and the release planning and:
- Removes impediments for the team
- Reports scrum teams’ progress, status, issues, and risks
- Coordinates with POs and CPOs across SoS to align the project schedule

In SoS the scrum master generates discussion of interdependencies and impediments across work-streams. SoS identifies changes in upcoming sprints that could lead to an impact analysis of changes and assist in prioritizing changes (see next section for SoS).

Case in point #6: Enhanced scrums roles

Enhanced scrum roles of PMO and the scrum masters were critical success factors for a global payments facilitator using scaled agile. In addition to establishing the long-term roadmap, the PMO created and maintained the project plan across the program and provided visibility into initiative execution for the leadership via weekly status reports and scorecards. The PMO also facilitated weekly ADM and WL meetings to identify and mitigate cross-team risks.

By taking over most of the tracking and reporting tasks, the PMO enabled the scrum-teams to focus on core product development activities.

The scrum master participated in the biweekly SoS, and coordinated across teams to manage tasks and resources and keep deliverables on schedule. The scrum master was also responsible for ensuring adoption of common standards and agile practices across scrum teams. This provided a common baseline for developing metrics, measuring progress across teams, and ensuring an apples-to-apples comparison of KPIs.
Execution tactics: Business process alignment
Aligning development to business processes can potentially generate benefits during execution and assists organization of epics and user stories across teams, as well as functional and E2E testing of code.

General considerations in epic and story organization and in testing are depicted in Figure 8.

Figure 8: Business process alignment

<table>
<thead>
<tr>
<th>Epic and story organization</th>
<th>Functional and E2E testing</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Development of comprehensive high-level epics that cover all features</td>
<td>• Set up functional testing using simulators and dummy data to replicate real-world environment</td>
</tr>
<tr>
<td>• Identification of stories on gives and gets between domains/scrums</td>
<td>• Identification of end-to-end flows that can be tested at different points along the roadmap</td>
</tr>
<tr>
<td>• Prioritization of user stories across sprints assist synchronized feature delivery</td>
<td>• Planning and resourcing for end-to-end testing sprints</td>
</tr>
</tbody>
</table>

Large-scale agile also involves additional process flow creation and scrum of scrums.

- **Process flow creation** entails developing E2E process flow diagrams to illustrate the sequence of activities (also see next section). This helps in defining dependencies among scrum teams, which eventually flow into user stories. This also helps in creating E2E test cases.

- **Scrum of scrums** is a multiteam meeting in which one member of each team works together to resolve dependencies and impediments. While the representative can be any team member, it is usually the scrum master. SoS can be held daily or two or three times a week, depending on how interrelated the features are among the teams.

An SoS meeting typically runs longer than a team’s daily scrum and can be broken into two parts. In the first part, each team answers the same questions at the team level. In the second part, the teams works to resolve impediments or proactively mitigate risks. The second part should be attended by architects and other senior individuals who can help resolve impediments.

Case in point #7: Business process alignment
A payments services company initiative involved 30 scrum teams and rigorously employed process flow diagrams and SoS to ensure business process alignment across scrum teams.

The process flow diagrams were developed in tandem with backlog creation, which enabled scrums to understand their role in the overall product environment, and to identify upstream and downstream dependencies with other scrums. Once these diagrams were developed and signed off by the different domains/scrums, they were employed in developing use cases and identifying validation points for the E2E functional testing.
Execution tactics: Management of development and releases

For large releases in scaled agile implementations, supplementing scrum-level testing with dedicated E2E testing cycles will increase efficiency and quality. Figure 9 illustrates at a high level one way in which scrum testing and E2E testing can be structured.

Figure 9: Organizing scrum and E2E testing

In addition to upfront planning iterations, agile at scale demands dedicated E2E testing cycles at the end of product development, although it goes against the basic agile tenet of building fully tested features in every sprint. The resulting buy-in from stakeholders generates support and assists the E2E team in meeting its objectives.

At the end of sprints, functional teams check the code, which is eventually integrated and pushed to E2E testing (after integration testing). This can provide two potential benefits: first, the E2E team has greater visibility into the complete system in a production-like environment without simulators and, second, the triage process for E2E defect management brings all stakeholders to a single platform and helps in assigning defects to the appropriate owner.

Case in point #8: Establishing an end to end testing program

When a global financial services organization migrated its 1,500-member delivery team to agile, it established clear scrum-level and E2E testing guidelines to ensure high-quality products. Dedicated sprints for E2E testing were established in the program roadmap and clear entry and exit criteria were set up for the E2E phase.

Once the scrum teams completed their development work, they were also responsible for completing component level unit testing, as well as neighborhood testing before the code was pushed to the E2E phase. At the same time, the E2E team worked with the product owners from each initiative to map the process flows using plain business language and diagrams to illustrate how the product should function after going live. This process-flow documentation enabled the E2E team to create test cases and to accelerate the testing process.
Practical considerations
Addressing the following practical considerations can assist an organization seeking to scale agile to larger projects:

• **Project selection:** The potential benefits of agile will likely be most apparent on projects with volatile requirements, aggressive timelines, and high end-user interaction. Business applications, self-service portals, and ecommerce and similar Web-based applications are good candidates. In addition, a hybrid approach may be useful for projects outside this profile or in organizations that want to take a more gradual approach to adopting agile.

• **Business involvement:** Agile methods require increased involvement from affected businesses and, in some cases, dedicated business resources. Therefore, it is more suitable to develop a mutual understanding of the frequency and type of involvement that will be needed from the business to specify needs, review progress, answer questions, and provide feedback.

• **Regulatory requirements:** Regulatory and compliance requirements typically call for rigorous exception and reconciliation reporting, which can be developed with agile methods. This can be done by embedding fulfillment of regulatory requirements into the work of each iteration and then auditing the test reports. In other words, regulatory requirements and reporting needs should simply be treated as features by the development team and addressed accordingly.

• **Human resources:** An agile team requires collaborative, cross-functional individuals who can understand business requirements, interact with customers, design features, and develop and test code. Agile teams should, whenever possible, place experienced developers in those roles and pair junior developers with more senior team members.

• **Senior management commitment:** Senior executives will probably be unaware of agile methods, and project managers may resist apparent loss of power. However, executives and project managers can be swayed by understanding the threat posed by more nimble competitors and seeing early wins at accelerated rates.

Much of the success of agile at scale depends on setting proper expectations and then diligently exercising project management methods and skills to meet those expectations.

Giving agile a go
Scaling agile to large projects requires multi-year roadmaps, strong governance, coordination of cross-team dependencies, consolidated reporting, and increased E2E testing and QA.

With these practices—and the methods described above—large financial services organizations can accelerate product and systems development while fulfilling business and regulatory needs so as to remain competitive with, or even outpace, innovative entrants in their markets.
Contacts

Industry leadership
Kenny M. Smith
Vice Chairman
US Banking & Securities Leader
Deloitte LLP
+1 415 783 6148
kesmith@deloitte.com

Authors
Joseph Cody
Principal
Deloitte Consulting LLP
+1 612 659 2630
jcody@deloitte.com

Divakar Goswami
Specialist master
Deloitte Consulting LLP
+1 615 718 5910
dgoswami@deloitte.com

Rajiv Ahuja
Director
Deloitte Consulting LLP
+1 408 704 4421
rahuja@deloitte.com

Acknowledgements
We would like to thank Vamsee Tadepalli, Ashish Kumar, Raj Pagare, and Alex Dixon of Deloitte Consulting LLP for their research support and contributions.
This publication contains general information only and Deloitte is not, by means of this publication, rendering accounting, business, financial, investment, legal, tax, or other professional advice or services. This publication is not a substitute for such professional advice or services, nor should it be used as a basis for any decision or action that may affect your business. Before making any decision or taking any action that may affect your business, you should consult a qualified professional advisor. Deloitte shall not be responsible for any loss sustained by any person who relies on this publication.

About Deloitte
Deloitte refers to one or more of Deloitte Touche Tohmatsu Limited, a UK private company limited by guarantee ("DTTL"), its network of member firms, and their related entities. DTTL and each of its member firms are legally separate and independent entities. DTTL (also referred to as "Deloitte Global") does not provide services to clients. Please see www.deloitte.com/about for a detailed description of DTTL and its member firms. Please see www.deloitte.com/us/about for a detailed description of the legal structure of Deloitte LLP and its subsidiaries. Certain services may not be available to attest clients under the rules and regulations of public accounting.

Copyright © 2015 Deloitte Development LLC. All rights reserved.
Member of Deloitte Touche Tohmatsu Limited.