McKinsey
& Company

McKinsey Digital

Agile in enterprise resource planning: A myth no more

ERP transformations are never easy. Agile can help improve your results.

by Didier Casanova, Swati Lohiya, Jerome Loufrani, Matteo Pacca, and Peter Peters

©Tinpixels/Getty Images

August 2019

Enterprise resource planning (ERP) solutions are a fundamental asset for most large companies, yet ERP transformations remain time-consuming and complex. An agile approach has the potential to dramatically streamline ERP projects, but IT professionals have long believed agile to be incompatible with ERP. Our experience in helping many organizations adopt agile practices in a wide variety of situations, however, has proved the opposite: that agile can successfully be applied to ERP programs, with quantifiably better results. The methodology simply has to be adapted to the unique requirements of these complex solutions.

Why ERP transformations remain important

Large ERP solutions have slipped to the bottom of IT management's agenda to make room for trendier topics, such as digital, big data, machine learning, and cloud. But the business benefits of ERP solutions—namely, the enablement of seamless, end-to-end integration across functions and the process standardization across geographies and business units—make them a fundamental asset for most large companies. Moreover, the next generation of ERP solutions, such as Oracle Cloud and SAP S/4HANA, offer even more promising capabilities, both functionally and technologically. Companies focusing on digital transformation or advanced-analytics programs are beginning to realize that, to unlock the full potential of their investments, linking the new technologies to their ERP base is essential.

The challenges of ERP transformations

As fundamental as they are, three-fourths of ERP transformation projects fail to stay on schedule or within budget, and two-thirds have a negative return on investment. There are five main reasons (Exhibit 1).

First, all parties may not share the same objectives. For example, a system integrator may have the incentive to increase the program's scope and duration if it makes more revenue from a complex integration. The company, meanwhile, wants to deliver the project and capture its value as soon as possible.

Second, most organizations lack experience in managing major IT projects and multivendor programs. They do not have enough skilled managers, have never set up rigorous governance for such programs, and fail to understand the level of input needed from business sponsors.

Exhibit 1

Three-quarters of enterprise resource planning transformations fail to stay on schedule or on budget, and two-thirds have a negative return on investment.

The challenges of enterprise resource planning (ERP) transformations

Misaligned incentives
All parties may not share the same objectives.

Poor project management
Most organizations lack experience in managing major IT projects and multivendor programs.

Lack of business–IT integration
ERP systems require complex discussions with the business on operating models, data management, and validation rights.

Missing focus on business value
Activities and deliverables tend to drive ERP transformations.

Waterfall methodology
Most ERP projects are undertaken using a linear, sequential waterfall approach, which delays the project's realization of value.

Source: McKinsey analysis

Next, ERP systems cover a vast, integrated, functional scope and thus require complex discussions with the business on operating models, data management, and validation rights. These decisions tend to come up mid-program and require executive-committee-level input based on information that is not yet available. The project must often pause for these decisions to be made, slowing progress and even undermining the initiative's value.

Fourth, activities and deliverables tend to drive ERP transformations; instead, the transformation should be based on business value, which must be quantified, documented, and monitored to drive the program.

Last, most ERP projects are undertaken using a linear, sequential waterfall approach, which delays the project's realization of value.

These challenges often cause ERP implementations to drag on for five or even ten years. The typical implementation involves long phases of design, specifications, and blueprinting but yields no measurable impact—while shareholder value diminishes, day after day.

The misconceptions and truths about agile and ERP

The myth that agile methodology cannot be applied to ERP implementations is based on several misconceptions: that an ERP implementation is too big and complex to be managed and delivered by small agile teams, meaning that highly integrated, intricate ERP requirements cannot be broken down into vertical user stories that can be developed and tested in the short sprints that define agile delivery; that ERP is a standardized software, and that hence an agile approach—which is designed for constantly changing or unknown requirements—is not needed or applicable; and that an ERP solution cannot be shown incrementally to end users, as they will not be able to perceive any value before it is fully built and integrated.

In truth, agile practices can greatly mitigate the risks and challenges that plague typical ERP implementations in a number of ways. Agile has, for example, vendors and system integrators work together as one end-to-end team focused on the same set of key performance indicators (KPIs) and outcomes.

It involves a faster pace and greater transparency, making it easier for managers to make timely, critical decisions. Contrary to popular belief, agile does not mean "no planning"—rather, agile replaces long, opaque project phases with two- to three-week sprints so that managers can track outcomes, progress, and challenges.

Agile calls for the business and IT groups to be integrated into the project team, which is structurally geared toward value creation. These two groups collaborate from the project's beginning, fostering agility for both.

And agile helps to break down the functional scope of ERP into a smaller set of features that small teams can deliver in sprints. This iterative approach helps projects to realize business value quickly.

In short, agile practices are exactly what is needed to manage ERP implementations. It should be no surprise that leading ERP vendors, such as SAP, are now promoting a more agile approach.

How to adapt agile to ERP

Some agile practices can be directly applied to ERP implementations without adaptation: forming small, end-to-end, cross-functional agile teams, with dedicated product owners from the business and end users; working in short cycles of two to three weeks to produce working software (or configurations, interfaces, et cetera) incrementally; adopting scrum-based ceremonies focusing on continuous improvement, with transparency facilitated by the ceremonies and KPIs; and using tools and technologies—such as test automation and continuous integration—that optimize and accelerate the delivery process.

Other agile practices, however, need to be adapted further. For instance, the project's entire scope must be defined up front at a high level to include clear success criteria, as opposed to agile's more common approach toward a minimum viable product. Teams should be allowed, however, to refine the detailed scope and to set priorities as they go along.

In addition, to ensure consistent development, more work must be done on the business process and architecture than in the typical agile implementation, so that the work can be split among small teams.

Strong linkages are needed between the agile teams delivering functionalities and the "transversal" teams, which are nonfunctional teams—for example, the data-migration team, the integration team, or the change team that support the functional or feature teams. All teams should be synchronized so that they work in the same rhythm and meet the finish line together.

"Production ready" software cannot be delivered as frequently as in typical agile software development. A phase of end-to-end (E2E) testing and cut-over is needed to consolidate the increments delivered by individual teams and to test complex interfaces with legacy systems; this often takes longer than one sprint to complete.

Finally, a strong agile program management office (PMO) should be added for faster resolution of issues and cross-team decision making.

Applying agile to the classical approach

A classical ERP implementation has four stages: developing an ERP strategy and road map, setting up the program, implementation, and deployment. Each stage can be adapted for agile delivery.

Developing an ERP strategy and road map results in a target architecture with high-level principles and a business case to implement the new solution. This stage remains largely unchanged, but it can be accelerated by doing a rapid fit-gap analysis at a high level, rather than endless blueprinting, and by working iteratively in sprints—to avoid an overly detailed business case. Product owners should be brought on board and empowered to make key decisions from the beginning, and smaller, cross-functional teams should be set up to achieve program goals.

Setting up the program changes substantially in an agile approach; it is much faster, primarily because the teams are empowered to quickly tackle real-life difficulties instead of engaging in theoretical design. This stage includes rapidly selecting a partner that has experience with the solution and agile—as opposed to engaging in a lengthy request-for-proposal process to try to find a supplier and negotiate a fixed-priced contract; building a high-level, macro-feature road map, based on a list of identified improvements, that is detailed enough to determine the size and form of the agile organization needed to deliver the program; staffing and training the organization in agile ways of working; and establishing a strong PMO that will coordinate the functional and nonfunctional workstreams.

Implementing the solution is dramatically different in an agile approach. Implementation happens in several waves to quickly capture value. Functional delivery teams adopt most of the typical scrum practices. End-to-end teams of eight to ten people, from both a company's business and IT and from the system integrator, complete design, development, and system testing in two- to three-week sprints. E2E testing and user-acceptance testing (UAT) are conducted at regular intervals—as opposed to only once at the end of the development—resulting in better code quality and ongoing test automation. Nonfunctional work (for instance, data migration, training, and deployment) is less affected by the agile approach, although close coordination is needed between functional and nonfunctional teams; for example, because data are required early for frequent functional testing, the data migration team must gather the data to populate the testing environment. Nonfunctional testing and the cut-over phase remain the same as in a classical implementation.

To illustrate, one company undertaking a transformation reorganized people into 26 teams. Of these, 11 were end-to-end, cross-functional agile teams delivering features, while 15 others were transversal teams that supported the agile teams. All agile teams included the capabilities required to deliver an end-to-end solution, including business representation (see sidebar, “How a logistics company used agile for its ERP transformation”).

Deploying the solution largely follows the classical approach, but deployments occur more frequently, and agile practices can help to remove bottlenecks. A “deploy all development rapidly” mind-set can mitigate early deployment risk, analytics can help to optimize the process (for example, the number of “key users” to be trained), and local templates can be designed early by onboarding local users. A shorter hypercare phase can be planned because of the continuous focus on quality. Since releases are more frequent in an agile approach, there is more opportunity to industrialize all steps in the deployment.

It is important to note that, in an agile-adapted implementation, the initial stages are accelerated when compared with the traditional waterfall approach. Most time is spent on later stages, focusing on delivering functionalities.

Benefits of adapting agile to ERP

Much of agile’s popularity is based on its results. Research shows that agile organizations have a 70 percent chance of being in the top quartile of organizational health, the best indicator of long-term performance.[1] Moreover, such companies simultaneously achieve greater customer centricity, faster time to market, higher revenue growth, lower costs, and a more engaged workforce.

Specific to ERP implementation, deploying ERP in an agile way—irrespective of the underlying technology—translates into a range of tangible and intangible benefits (Exhibit 2):

— reduction of program cost by 10 percent, driven primarily by having to do less rework in the E2E testing and UAT phases

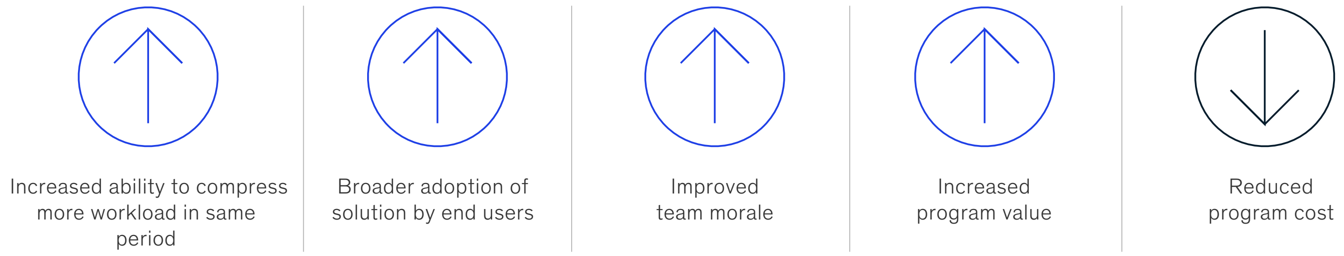
— increase in the program’s value by 20 percent by giving the product owner enough visibility into the project’s progress to focus on high-value items

Article continues on page 8

Exhibit 2

Enterprise resource planning transformations are always challenging, but these challenges can be far less daunting with an agile approach.

Some benefits of adapting agile to enterprise resource planning (ERP) transformations


Increased ability to compress more workload in same period
Broader adoption of solution by end users
Improved team morale
Increased program value
Reduced program cost

Source: McKinsey analysis

[1] Michael Bazigos, Aaron De Smet, and Chris Gagnon, “Why agility pays,” *McKinsey Quarterly*, December 2015, McKinsey.com.

How a logistics company used agile for its ERP transformation

One of the largest shipping and logistics providers in Europe embarked on a multiyear core-technology transformation. The program's goal was to replace the old enterprise resource planning (ERP) system with up-to-date technologies and to provide new functionalities. A few years in, the program was fraught with multiple challenges and lacked a business case, business ownership, and robust vendor and program management. Also, the scope was too large and complex to be delivered in an effective and sequential waterfall manner.

To get the program back on track, the company focused on three steps: (1) rescoping the program around the most valuable elements, (2) designing and implementing an agile ERP delivery methodology, and (3) establishing a rigorous PMO.

To enable agile delivery, a new agile operating model was designed for 300-plus full-time-equivalent (FTE) employees by aligning numerous stakeholders—including business and IT internal clients, the ERP vendor, the company's system-integration partner, the onshore- and offshore-development partner, and the infrastructure partner. The FTEs and the program were then transitioned to agile delivery at an accelerated pace.

To do so, the FTEs were first reorganized into six functional domains, that consisted of 11 cross-functional agile teams, focusing on such tasks as developing and integrating the product catalog. There were six transversal domains with about 15 teams focusing on areas such as data migration and electronic data interchange (EDI). Next, a detailed agile approach was designed to consider ERP specificities, and the new organization was trained and coached in this new approach. The agile PMO steered the program, supported agile ceremonies for the overall project, solved complex issues, facilitated the swift removal of any impediments, and implemented a new backlog management and tracking tool.

The application of agile to this ERP implementation resulted in several significant benefits:

— *Enhanced transparency.* The project teams put their macrofeatures ("epics") into a work-flow-management tool that attached each to the increment or sprint in which it was to be delivered. At the end of every sprint, progress was measured, in the number of user stories and story points delivered, and of the entire project, in the number of epics delivered and analyzed. Because of this transparency, the teams could measure and take ownership of their progress, which enabled them to rapidly correct their course using agile retrospectives as a tool. The teams were also able to promptly escalate any impediments to their managers. The managers enjoyed an unprecedented level of transparency; they could now see the duration, cost, and causes of delays each week and take swift action. Moreover, knowing the project's precise status on an ongoing basis allowed the product owners and leadership to make informed decisions about what to prioritize based on value.

— *Strong coordination among teams.* To foster coordination and communication among teams, the process began with increment planning with all teams—both functional and transversal—together in one room. Then, at the start of every sprint, each team gave its input to the other teams on dependencies. Functional and transversal teams had biweekly huddles. From there, issues were escalated to the weekly ''war room,'' where all teams met to discuss dependencies and performance.

- *Rapid, targeted troubleshooting.* The PMO, comprising roughly 12 to 15 people, in addition to performing classical activities, served as a SWAT team that could address complex issues on an ad hoc basis. Almost half of the PMO's work focused on troubleshooting, such as scenario building to assess the impact of a delay in delivering a large interface; building a complex, CEO-ready document about rescoping; providing extra analytical capacity to plan 3,000 test cases; and reorganizing the full operating model as needs evolved—for example, merging functional teams after rescoping and building an efficient test factory based on lean principles.

- *Agile organization.* The 11 agile teams had the capabilities to deliver an end-to-end solution, including business representation. Their three-week sprints included development, solution testing, and a demonstration to end users at the end of each sprint. In addition, at the end of each increment (or three-sprint period), comprehensive, end-to-end solution- and user-acceptance testing was performed to ensure the quality and integrity of the functionality delivered. The teams followed all scrum-based ceremonies and began to realize benefits after only a few weeks.

- *A detailed agile playbook.* The agile approach—which was tailored to the company and ERP—was documented in detail in a playbook that remained a living document throughout the program. The playbook included elements such as how to translate traditional ERP requirements into epics and user stories to create a product backlog, as well as project documentation that was adapted to agile—for instance, simplified technical specifications, since developers were working directly with product owners and analysts, and new terminology, such as the definition of "ready and done."

As a result, the program was able to meet and even exceed its performance targets. Delivery was 20 percent faster than the previous estimate. This was a result of far less rework due to iterative improvements made by working with end users to inspect and adapt each iteration. It was also a result of the ability to better manage project delays, which made them less likely to affect the overall timeline because of the use of intermediary deadlines and having an incremental scope. Additionally, fewer bugs were found by using integrated, end-to-end, and user-acceptance testing for the agile release—as opposed to the two previous waterfall releases—and by conducting more-frequent testing. The scope delivered was three to ten times greater than in previous releases of similar durations, due to better alignment among functional teams. User acceptance of the new solution was much higher, as users were involved throughout the implementation. Finally, the agile team's morale improved significantly, as measured during agile retrospectives, which contributed to the delivery's success.

Since the project began, more than 100 people have now been trained in the agile mind-set and ways of working, resulting in a new operating model for the company, which reflects the realized benefits of the agile approach—and ultimately disproving the myth that agile does not apply to ERP.

- ability to compress three times more workload into a given period through greater parallelization of functional teams

- wider adoption of the solution by end users, as they are involved throughout the implementation

- improvement in team morale, as they see the solution implementation's measurable progress every day

Although ERP systems are often considered a "necessary evil," they are here to stay and cannot be ignored as companies go digital. The traditional, often complicated approach to ERP transformation should be drastically revised and, whenever possible, adapted to include agile ways of working.

Companies and system integrators should dispense with the myth that agile cannot be applied to ERP and instead industrialize the agile approach for ERP transformation. Further, ERP solutions should become more modular so that deployment can be phased—resulting in lower costs and faster realization of value.

ERP transformations are always challenging, but these challenges can be far less daunting with an agile approach.

Didier Casanova is an associate partner in McKinsey's Brussels office; **Swati Lohiya** is a senior expert in the London office; **Jerome Loufrani** is an associate partner in the Paris office, where **Matteo Pacca** is a senior partner; and **Peter Peters** is a partner in the Düsseldorf office.

Designed by Global Editorial Services