How Can You Scale Your Agile Adoption?
by Diego Lo Giudice, February 5, 2014

KEY TAKEAWAYS

Agile Requires Discipline, Scale, And Perseverance
Agile development is all about common sense and discipline, but many organizations find discipline to be a challenge. To scale Agile, application development leaders must apply (and reapply) modern management techniques to spread discipline in their teams. It takes many years -- more than five -- for a large-scale transition to Agile.

Start Early With Both Agile PM And Agile Engineering Practices
It’s not enough to adopt Agile project management techniques like Scrum and postpone Agile engineering ones like XP. Embrace bits of both from the beginning; accelerate your downstream development and delivery practices as you accelerate the streamlining of your upstream requirements and Agile project management process.

Change The Way You Test Or You’ll Never Be Agile Enough
Some Agile engineering practices change the way you need to test and requires big changes -- from the way you deal with test management to the way you automate unit tests. Integrate testing early in your Agile PM practices, get testers closer to developers, and test within sprints as much as possible. Aim to achieve this goal as soon as possible.
How Can You Scale Your Agile Adoption?
Results From Forrester's Q3 2013 Global Agile Software Application Development Online Survey
by Diego Lo Giudice
with Holger Kisker, Ph.D. and Nasry Angel

WHY READ THIS REPORT
Thirteen years after Ken Schwaber, Jim Highsmith, and 15 others signed the Agile manifesto, Agile is still on everyone's lips. Data from Forrester’s Q3 2013 Global Agile Software Application Development Online Survey indicates that a whopping 90% of all teams practicing Agile development have adopted Scrum. As with any paradigm shift, the years of increasing momentum for Agile development are over, to be replaced by digestion and refinement. In the Agile context, “digestion and refinement” really means scaling — that is, adopting more mature Agile practices and extending and replicating point experiences throughout the enterprise to grow and broaden the business benefits. Our data shows that scaling Agile is hard — but it can be done. This report helps application development professionals understand what organizations are doing to scale Agile and recommends some good practices for doing so.
MANY COMPANIES ARE STILL IN THE EARLY STAGES OF AGILE ADOPTION

Forrester’s Q3 2013 Global Agile Software Application Development Online Survey follows up on a similar survey that we ran in 2011. The most recent survey shows that companies are still primarily plucking the low-hanging fruit of Agile — but they’re also becoming more objective in evaluating their Agile maturity. Firms are using waterfall at more realistic levels; 90% of them have adopted Scrum; nearly half use Lean and Kanban; and more than one-quarter have adopted test-driven development (TDD) (see Figure 1). In addition, the survey indicates that 28% of the organizations using Agile have been doing so for more than two years. However, the scale of Agile use is still confined to the project level — where the low-hanging fruit seems to be. What improvements have organizations made in their use of Agile in the past two years?

**There’s less water-Scrum-fall.** Survey data indicates that enterprises are applying Agile techniques more rigorously. This begs the question: Are we finally overcoming water-Scrum-fall? Analysis of hundreds of inquiries from Forrester clients and the results of the Agile survey shows that there’s a “good” water-Scrum-fall and a “bad” water-Scrum-fall. Good water-Scrum-fall is when teams do just enough preparation and planning before normal Scrum sprints happen (sprint 0 planning), testing starts earlier, and “done” is well-defined, with clear and complete testing objectives. Deployable code is staged at every sprint — not necessarily in production, but ready to be deployed. Bad water-Scrum-fall is when most requirements are set up front rather than iteratively refined, sprint 0 is executed like a traditional requirements step instead of a planning one, and testing, integration, and release are afterthoughts.

**Scrum alone is not enough; many mix it with other Agile methods.** Another common practice is to mix and match Scrum with lean techniques like Kanban and increase the focus on minimum viable product (MVP), work in progress, and throughput. One-quarter of respondents adhere closely to the Agile methods they select — up from 17% in 2011. On the other hand, many firms extend Agile project management (PM) with complementary Agile practices to align the upstream and downstream sides of end-to-end application life-cycle management (ALM) processes with DevOps on the downstream side. In order to support broader scale, teams then start adding Agile engineering practices — like Agile testing, extreme programming (XP), TDD, continuous build, integration, and automation — to complement Scrum Agile PM practices.

**Distributed Agile is becoming the norm.** It’s interesting that 36% of survey respondents claim to be doing Agile in globally distributed teams separated by eight time zones or more, and a further 40% say that their Agile teams are separated by up to eight time zones — meaning that only 24% of companies have truly colocated Agile teams! This debunks the common thinking that Agile can only be done if colocated and means that there are practices and tools out there that support more — and more distributed — teams. Finally, the more familiar companies are with Agile, the more they use outsourcing and offshoring to augment their Agile capabilities.
Figure 1 Scrum Is By Far The Most Prevalent Development Approach

“What development approaches does your organization use?”

<table>
<thead>
<tr>
<th>Approach</th>
<th>2013</th>
<th>2011*</th>
</tr>
</thead>
<tbody>
<tr>
<td>Scrum</td>
<td>90%</td>
<td>82%</td>
</tr>
<tr>
<td>Waterfall</td>
<td>54%</td>
<td>44%</td>
</tr>
<tr>
<td>Iterative</td>
<td>53%</td>
<td>59%</td>
</tr>
<tr>
<td>Lean</td>
<td>33%</td>
<td>48%</td>
</tr>
<tr>
<td>Kanban</td>
<td>42%</td>
<td>37%</td>
</tr>
<tr>
<td>Extreme Programming (XP)</td>
<td>30%</td>
<td>36%</td>
</tr>
<tr>
<td>Test-driven development (implement test cases first, then the code; TDD is not unit testing)</td>
<td>27%</td>
<td>37%</td>
</tr>
<tr>
<td>DevOps</td>
<td>10%</td>
<td>22%</td>
</tr>
</tbody>
</table>

Base: 149 technology management professionals from organizations that are implementing or have implemented Agile
*205 technology management professionals from organizations that are implementing or have implemented Agile

Source: Q3 2013 Global Agile Software Application Development Online Survey

SCALING INITIAL PRACTICES AND CLAIMING MORE BENEFITS IS HARD

After trying Agile for a few years in isolated projects, many app-dev leaders and their business counterparts who enjoyed early benefits are looking for more leverage. Initial experience suggests that the expected benefits all accrue to the business side: better business alignment, more predictable releases, and more midcourse corrections (see Figure 2). However, organizations find that achieving those benefits repeatedly and scaling them up, better aligning app-dev efforts with business stakeholder needs, and consolidating engineering practices like continuous build and integration into a reliable process to deliver apps even faster and better is a long, difficult journey. Here’s why:

- **There’s no standard road map to scale Agile.** Different organizations scale Agile differently. Some dive deeply into Agile practices before replicating the experience in other parts of the organization. Most have only adopted Scrum or Agile on the upstream side of their ALM
process, and later added Agile engineering or Agile on the downstream side. One-third of respondents focus on making the upstream process more agile than the downstream one (see Figure 3). It’s clear that, in order to scale Agile, companies need to use bits of both early on — pieces that might be fundamental. The advent of DevOps will balance the adoption of Agile on the upstream and downstream sides, enabling companies to scale Agile faster.

- **It takes years to scale Agile and replicate the initial benefits.** The Microsoft Visual Studio development team — several thousand developers distributed around the world — began adopting Agile in 2005. At the recent Visual Studio 2013 launch in New York, Brian Harry, product unit manager for Team Foundation Server, said that his team has only recently slashed sprints down to three weeks. Our survey also shows that scaling Agile, whether by adopting more mature practices or replicating Agile throughout the enterprise, can take years. There are no shortcuts — and our data shows that the broader the scale, the longer it takes (see Figure 4).

- **Shortening release cycles requires deep testing changes . . .** Forrester believes that one of the most challenging transformation issues in organizations’ Agile adoption is the need to revolutionize testing (see Figure 5). Testing is too slow, has too many impediments, and has too many manual steps and processes to support Agile delivery. Testing has to become a first-class citizen in development: Testers must be peers and partners of developers, must provision testing environments quickly and consistently, get rid of most manual testing, and adopt smart testing approaches rather than full-coverage testing. This is a huge organizational change of the testing center of excellence.

- . . . and automating the build-integration-deployment cycle. Delivering faster and better software applications to the business requires more than just focusing on the upstream side of the development process. Teams need to streamline and automate the downstream side to keep up with the smaller, more frequent chunks coming from the upstream side. Infrequent builds, delayed testing, and infrequent, manually repeated large batch integrations won’t cut it. The risk of poor automation of these three steps and the inability to deliver fast enough will put your business relationship at further risk.
Figure 2  The Top Three Benefits That Early Agile Adopters Expect

“What benefits does your organization expect to get from using Agile?”

- Better business/IT alignment: 28%
- More predictable releases: 20%
- More opportunities for midcourse corrections: 11%
- Greater predictability of results aligned with requirements: 10%
- Increased motivation and morale: 4%
- Improved technical quality: 4%
- Improved functional quality: 3%

Base: 149 technology management professionals from organizations that are implementing or have implemented Agile (multiple responses accepted)

Source: Q3 2013 Global Agile Software Application Development Online Survey

Figure 3  Low Adoption Of Mature Agile Practices Limits Agile Scaling

“Which Agile practices do you apply?”

- Group 2: 13%
- Group 1: 32%

Base: 149 technology management professionals from organizations that are implementing or have implemented Agile (multiple responses accepted)

Source: Q3 2013 Global Agile Software Application Development Online Survey
Figure 4: The Increase in Agile Adoption Is Proportional to the Number of Years Using It

“How long has your organization been using Agile and how widespread is it in the organization?”

<table>
<thead>
<tr>
<th>Duration</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Less than six months</td>
<td>1%</td>
</tr>
<tr>
<td>Six months to one year</td>
<td>6%</td>
</tr>
<tr>
<td>One to two years</td>
<td>6%</td>
</tr>
<tr>
<td>Two to three years</td>
<td>4%</td>
</tr>
<tr>
<td>Three to four years</td>
<td>5%</td>
</tr>
<tr>
<td>Four to six years</td>
<td>14%</td>
</tr>
<tr>
<td>More than six years</td>
<td>3%</td>
</tr>
</tbody>
</table>

Agile adoption has matured and scaled throughout our enterprise

We have adopted Agile at the divisional level

Base: 79 technology management professionals from organizations that have scaled Agile to at least the divisional level

Source: Q3 2013 Global Agile Software Application Development Online Survey
Source: Forrester Research, Inc.
**SCALING AGILE: HERE’S HOW**

A two-speed market adoption appears to have consolidated: 32% of our respondents have adopted a minimum but necessary set of good Agile project management practices, while 13% have instead adopted both Agile PM and Agile engineering practices. Viewing our survey data in the light of what dozens of clients that have successfully scaled Agile have told us, we have defined a recommended minimum set of practices that we believe app-dev leaders should take into consideration in their journey to becoming more Agile (see Figure 6). These include adopting Forrester’s five Agile upstream must-haves; integrating Agile downstream practices early; and sourcing intelligently, measuring, and improving.
Figure 6 Scaling Agile: Here’s How

32% of companies have some Agile maturity on the upstream side — what Forrester refers to as “Agile PM” (e.g., Scrum). But 13% have also adopted Agile on the downstream side or adopted Agile engineering practices like Agile testing, TDD, continuous integration/continuous development, and other XP practices.

Five Agile upstream must-haves

1. Have daily standups to plan and prioritize business value delivery.
2. Report status to all stakeholders using burndown/burnup and Kanban boards.
3. Put product owners from the business in charge and focus on MVP.
4. Break requirements into small chunks; continuously iterate and collect feedback.
5. Connect Agile projects to enterprise PMO planning and reporting.

Source intelligently, measure, and improve

- Get the skills: If you can’t hire for organic growth, find skills elsewhere; 41% of firms that use SIs have more predictable releases.
- Measure progress: You can only manage what you measure. Use metrics like predictability, cycle time, percent automated, and number of bugs in production. Constantly review and adjust metrics.
- Use value metrics: In the age of the customer, Agile’s value is to encourage product owners to use value points to prioritize user stories so developers focus on delivering higher-value software sooner.
- Learn to improve: Learning from each part improves the whole. After every step — not just in sprints — stop and ask what went well, what didn’t, and what you can do to improve.

Add some lean startup thinking to the mix!

Think of Agile as more of a lifestyle than a list of rules. It should permeate daily activities, not just be a static structure with lots of processes that limit people and hinder creativity. You must implement Forrester’s five must-haves and iterate through downstream practices.

Important!

Integrate Agile downstream practices early on

- **Do**
  - Martin Fowler says: “Integrate work frequently — multiple times per day.”
  - 20% say that testing always starts from Sprint 0 and runs continuously.
  - 22% practiced DevOps in 2013.

- **Don’t**
  - Forrester says: “Don’t build and integrate manually!”
  - 12% say that testing never starts from Sprint 0 and runs continuously.
  - 10% practiced DevOps in 2011.

Base: 148 technology management professionals from organizations that are implementing or have implemented Agile

Source: Q3 2013 Global Agile Software Application Development Online Survey

Source: Forrester Research, Inc.
Adopt Forrester’s Five Agile Upstream Must-Haves

Analysis and successful case studies in the last four or five years show some common patterns. Make sure you embrace them:

- **Keep the team aligned as a one-man band.** Most impediments that appear during Agile projects must be resolved quickly; this can only happen if teams are empowered and self-contained. An Agile team has to stay aligned by communicating constantly through daily check-ins (standups) and weekly planning meetings and retrospectives — even if it’s distributed. The best teams have the freedom to make their own decisions. This can only be achieved if the teams are cross-functional and all interested stakeholders participate.

- **Report transparently to all stakeholders.** All team members must constantly share information via burndown/burnup charts or other reporting artifacts like Kanban boards. These boards report how close the team is to delivering the user stories it’s committed to, how fast its work is progressing, how it burns down the activity backlog and gets story points up. Firms should digitize this information with the proper tools so that everyone has access and can view the status. Distributed teams need strong communication, collaboration, and Agile PM tools like Rally, VersionOne, and Leankit.

- **Put product owners from the business in charge and focus on MVP.** Product owners (POs) must be from the business and must enforce the MVP mindset; if necessary, firms should support them with business analysts. POs have to commit 1 to 2 hours of their time every day, either on demand or regularly, and must dedicate at least half a day per week for demos, app testing, user acceptance testing, and product and user story backlog planning and prioritization. Good POs can make a real difference between success and failure in Agile projects. Scaling this concept is crucial.

- **Break up work into small chunks, iterate frequently, and collect feedback continuously.** The secret to Agile is to divide and conquer — something the Romans learned 2,000 years ago and used to rule a good portion of the world! In Agile, break requirements into smaller chunks and allow the business to get deliveries more frequently or on demand. Smaller chunks of software delivered sooner in short iterations allow your business more opportunities for feedback and the opportunity for app-dev teams to leverage frequent feedback to make change easier and less expensive. Set the goal of two-week iterations — and attain it.

- **Connect Agile projects to enterprise PMO planning and reporting.** If your Agile projects have been living in a separately budgeted and managed ivory tower, it’s time to integrate them into your enterprise program governance. This means that the project management office (PMO) needs to link Agile project budgets to its broader program of budgeting and portfolio management. This might require both a process change and a retooling initiative (e.g., Agile PM or portfolio project management).
Integrate Agile Downstream Practices Early On

Organizations must adopt the above practices together with engineering ones. Having both is what enables true scaling and leverage of Agile. Key engineering practices to adopt are:

- **Build and integrate continuously.** Regardless of whether you’re going to deliver daily, weekly, monthly, or less often, practicing build and continuous integration reduces the risk of not fulfilling the promise of faster delivery to your business and upstream partners. Not integrating code early and continuously makes it very unpredictable to estimate how long it will take to integrate a large amount of new code that has been developed over a long period of time. Forrester recommends building daily, automating builds with one command, and aiming to commit builds to an integration environment at least daily.10

- **Test continuously.** In order to build and integrate daily, code needs to be self-testing. Advanced techniques like TDD help in achieving this, but it’s sufficient that all developers commit to writing automated unit tests for all code they develop. Teams must run unit tests on each build and fix bugs immediately. Additional comprehensive end-to-end testing has to happen within sprints. Move testers into development teams and get them involved from sprint 0. They will create a definition of “done” that includes complete testing and then make sure it happens: component, regression, and integration testing for both functional and nonfunctional requirements. This will be difficult in the beginning, and might require improvement over time; initially, the team should allow for “test-hardening” sprints every three or four sprints. Aim for complete continuous testing in each sprint.

- **Set up for more frequent delivery and deployment.** The frequency of delivery and deployment varies and depends on business needs. Build a deployment pipeline to streamline, optimize, and automate the delivery and deployment process.11 Mobile apps require very short deployment cycles, while web apps might take longer. Distinguish between delivery and deployment needs: Delivery is the act of generating integrated, tested, and executable code in a preproduction environment, while deployment is the next step — putting the code into production. The process can be automatic or “controlled” for approval by humans; it requires tight collaboration between development and operations teams. Frequent deployment requires strong practices (DevOps); tools are necessary when scaling Agile.12

Source Intelligently, Measure, And Improve

- **Don’t have the skills? Get them from your partners.** Adopting Agile is about change: learning new practices and developing new skills. As you scale, you will need more resources with new skills and mindset. You have two options: grow organically by hiring and training existing people or augmenting your resources by partnering with external consulting or systems integrators that provide skilled resources. You might also need to source key Agile resources to coach your teams. Our survey shows that organizations are more successful when using external Agile coaches (see Figure 7).
- **Measure progress, quality, and efficiency based on context.** You can't manage what you don't measure. Agile requires a new set of metrics that teams should adopt — but which metrics you choose depends on your goals related to context, complexity, and time.\(^\text{13}\) Start with three types of metrics: progress, quality, and efficiency. Specifically, Agile teams use metrics like velocity (the number of story points per sprint), predictability (delivering when we say we'll deliver), cycle time (e.g., the time it takes to make a build, integrate it, and promote it to the next stage), the percentage of test case automation, the number of bugs in production, and the mean time to repair.

- **Add value measurements to customer satisfaction where possible.** In addition to progress, quality, and efficiency, Agile encourages the adoption of value metrics. Delivering code earlier delivers more value — but value is hard to measure. Some teams measure it by assigning a relative number of points to each user story that the PO provides. This is a simple but effective approach, and a good way to start. Almost everyone tends to measure customer satisfaction; according to our survey, 19% measure it at the end of each significant landmark or release and another 19% at regular intervals but less often than every six months. A plurality (34%) takes an analytical approach to the large amounts of data they generate and collect.

- **Learn from everything and improve the whole.** Retrospectives are a common practice in Scrum. Scaling and sharing that information across all teams to leverage is helpful to steer the scaling strategy in the right direction and harvest good practices. As teams collect more data through tooling, analytics, and business intelligence, they use different approaches to learn from the data and make decisions on how to improve.
**Figure 7** Augmenting Agile Skills With External Consultants And Coaches Works

“What benefits do you expect to get from an Agile systems integrator (SI)?”

<table>
<thead>
<tr>
<th>Benefit</th>
<th>Yes (%)</th>
<th>No (%)</th>
<th>Don't Care (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Better business/IT alignment</td>
<td>62%</td>
<td>29%</td>
<td>14%</td>
</tr>
<tr>
<td>Greater predictability of releases</td>
<td>10%</td>
<td>21%</td>
<td>7%</td>
</tr>
<tr>
<td>More opportunities for midcourse corrections</td>
<td>10%</td>
<td>14%</td>
<td>13%</td>
</tr>
<tr>
<td>Greater predictability of results aligned with requirements</td>
<td>19%</td>
<td>23%</td>
<td>7%</td>
</tr>
<tr>
<td>Increased motivation or morale</td>
<td>21%</td>
<td>13%</td>
<td>3%</td>
</tr>
<tr>
<td>Improved technical quality</td>
<td>3%</td>
<td>5%</td>
<td>8%</td>
</tr>
<tr>
<td>Improved functional quality</td>
<td>7%</td>
<td>5%</td>
<td>8%</td>
</tr>
<tr>
<td>Greater frequency of releases</td>
<td>10%</td>
<td>7%</td>
<td>19%</td>
</tr>
</tbody>
</table>

Source: Q3 2013 Global Agile Software Application Development Online Survey

**WHAT IT MEANS**

**START ACTING LIKE A LEARNING DELIVERY ORGANIZATION**

As organizations get better at Agile, they will need to adopt more and newer practices to scale and better serve larger numbers of customers. The core shift in these practices will focus on the experimentation, change, measurement, and learning necessary to deal with the growing uncertainty of customer needs. In the coming years, we predict that app-dev leaders will:
- **Feed the development pipeline with user feedback.** Developers will receive requirements continuously from two sources: the business, through the definition of user stories, and from production, where the business and end users are using the product. Mobile will push more and more development practices on systems of engagement, creating a probe-sense-respond cycle.¹⁴

- **Enable fast delivery through DevTestOps.** Agile has mostly focused on the upstream side of the delivery process, but the huge shift and momentum of Agile on the downstream side is picking up. App-dev leaders will need to focus efforts on both sides and be more holistic in how they apply mature Agile practices. Cross-functional teams will break down siloed functional structures, enabling tight collaboration between development, testing, and operations (we call this DevTestOps). Leading-edge groups will drive TestDevOps with strong testing practices like TDD.¹⁵

- **Lead with learning development teams — or fail.** Teams will adapt their delivery process to deal with more change faster — but they won’t be able to get there in one leap. By collecting feedback from the business customers that use their apps in production, and by collecting and refactoring the lessons they learn from all team players, teams will be able to connect the dots between cause and effect and learn what to improve next. Teams that don’t have a delivery process to learn from mistakes or that won’t embed feedback into their delivery process will lose.

---

**SUPPLEMENTAL MATERIAL**

**Methodology**

Forrester’s Q3 2013 Global Agile Software Application Development Online Survey was fielded to 149 technology management professionals from organizations that are implementing or have implemented Agile. For quality assurance, we screened respondents to ensure they were at least “Agile beginners,” meaning that they have already started at least one project with Agile development.

Forrester fielded the survey from July to August 2013. Respondent incentives included a summary of the survey results. Exact sample sizes are provided in this report on a question-by-question basis. This survey used a self-selected group of respondents with knowledge of their organization’s Agile practices and is therefore not random. This data is not guaranteed to be representative of the population, and, unless otherwise noted, statistical data is intended to be used for descriptive and not inferential purposes. While nonrandom, the survey is still a valuable tool for understanding where users are today and where the industry is headed.
In addition to sampling error, one should bear in mind that the practical difficulties in conducting surveys can introduce error or bias into the finding of opinion polls. Other possible sources of error in polls are probably more serious than theoretical calculations of sampling error. These other potential sources of error include question wording, question ordering, and nonresponse. As with all survey research, it is impossible to quantify the errors that may result from these factors without an experimental control group, so we strongly caution against using the words “margin of error” in reporting any survey data.

These statements conform to the principles of disclosure of the National Council on Public Polls.

You can find more information about the data on the Survey & Data page online. From this page, you will be able to download the survey instrument.

ENDNOTES

1 For details of the results of the November 2011 Global Agile Software Application Development Online Survey, see the February 8, 2012, “Justify Agile With Shorter, Faster Development” report.

2 Forrester published the bulk of the data from this survey last year. See the October 28, 2013, “What Are Companies Doing To Scale Agile Development?” report.

3 However, it’s not uncommon to find waterfall, iterative, and Agile at the same companies, but used for different types of apps. We estimate that the real picture is that half of the respondents claiming waterfall and iterative are mostly doing water-Scrum-fall, waterfall, or iterative in separate areas of the organization (systems of engagement and mobile with more agile and iterative techniques; systems of record with waterfall).

4 For a definition of water-Scrum-fall, see the July 26, 2011, “Water-Scrum-Fall Is The Reality Of Agile For Most Organizations Today” report.

5 MVP comes from Lean management. Eric Ries, who wrote the New York Times bestseller The Lean Startup, defines a “minimum viable product” as one that has enough features (and no more) that allows to ship a product that resonates with early adopters; some of whom will pay money or give feedback. WIP is also a Lean process concept, while throughput time (TPT) is the time before a product emerges from a particular production facility, once it is entered there.

6 For more on what it means to access the distributed Agile capabilities of tech services partners, see the November 12, 2013, “Offshore Agile Comes Of Age” report.

7 Sprint includes testing and software ready to ship.

8 For more on the changes to the testing process and organization, see the January 15, 2013, “Consistent Performance In Agile Teams Must Include Testing” report and see the July 18, 2013, “Navigating The Agile Testing Tool Landscape” report.
The upstream side referred to here would mean adopting Scrum, MVP for requirements prioritization, and other Agile PM-related practices. The downstream side looks more to engineering steps like testing, TDD, and continuous integration and development.


For more on the principles and technical practices that enable rapid, incremental delivery of high quality, valuable new functionality to users, check out Jez Humble and David Farley, Continuous Delivery, Pearson Education, 2010.

DevOps has been gaining momentum and achieving impressive results. The seven main principles of DevOps have been proven by some of the most dominant technology innovators in the world. See the September 3, 2013, “The Seven Habits Of Highly Effective DevOps” report.

To understand the metrics that successful Agile teams use to measure their progress see the September 9, 2013, “Agile Metrics That Matter” report.

For a number of valuable, commonly collected performance metrics and a description of the tools and frameworks to help collect them, see the November 18, 2013, “Measuring Mobile Apps” report.

Service virtualization and testing (SVT) solutions provide developers and testers with tools to quickly simulate the services of a complex production environment, mainly for automating regression, integration, and performance tests. In doing so, SVT enables companies to define complex test scenarios, provision more consistent production-like test labs more quickly, and test in fast-paced Agile environments to improve testing speed and product quality. In Forrester's 15-criteria evaluation of service virtualization and testing vendors, we evaluated five solutions offered by CA Technologies, HP, IBM, Parasoft, and SmartBear Software. See the January 27, 2014, “The Forrester Wave”: Service Virtualization And Testing Solutions, Q1 2014” report.
About Forrester

A global research and advisory firm, Forrester inspires leaders, informs better decisions, and helps the world’s top companies turn the complexity of change into business advantage. Our research-based insight and objective advice enable IT professionals to lead more successfully within IT and extend their impact beyond the traditional IT organization. Tailored to your individual role, our resources allow you to focus on important business issues — margin, speed, growth — first, technology second.

FOR MORE INFORMATION
To find out how Forrester Research can help you be successful every day, please contact the office nearest you, or visit us at www.forrester.com. For a complete list of worldwide locations, visit www.forrester.com/about.

CLIENT SUPPORT
For information on hard-copy or electronic reprints, please contact Client Support at +1 866.367.7378, +1 617.613.5730, or clientsupport@forrester.com. We offer quantity discounts and special pricing for academic and nonprofit institutions.

Forrester Focuses On Application Development & Delivery Professionals

Responsible for leading the development and delivery of applications that support your company’s business strategies, you also choose technology and architecture while managing people, skills, practices, and organization to maximize value. Forrester’s subject-matter expertise and deep understanding of your role will help you create forward-thinking strategies; weigh opportunity against risk; justify decisions; and optimize your individual, team, and corporate performance.

« ANDREA DAVIES, client persona representing Application Development & Delivery Professionals