A Forrester Consulting
Thought Leadership Paper
Commissioned By
Microsoft

March 2015

Infrastructure As Code: Fueling The Fire For Faster Application Delivery

FORRESTER®

Table Of Contents

ABOUT FORRESTER CONSULTING

Forrester Consulting provides independent and objective research-based consulting to help leaders succeed in their organizations. Ranging in scope from a short strategy session to custom projects, Forrester's Consulting services connect you directly with research analysts who apply expert insight to your specific business challenges. For more information, visit forrester.com/consulting.

FORRESTER®

Executive Summary

To achieve success in today's digital age, enterprises *must* be customer-obsessed and systematically reinvent themselves to serve informed and empowered customers. Leading companies rely heavily on software applications to win, serve, and retain these customers. Software creates better customer experiences, and delivering innovative software faster gives companies a distinct competitive advantage.

But faster delivery of applications is challenging for both infrastructure and operations (i.e., Ops) professionals and application development and delivery (i.e., Dev) teams. Both are under pressure to increase speed without compromising quality. To maximize their efforts, neither team can go it alone — they must work together to find a common set of processes and tools that help them both. This is the core of the DevOps movement.

Treating infrastructure as code is a key element of DevOps, and it benefits both teams. Developers become more involved in specifying configurations, and Ops teams get involved earlier in the development process. Whether you are a developer or operations professional, you should understand and share your knowledge of and experience with infrastructure as code (IaC) with your colleagues.

In January 2015, Microsoft commissioned Forrester Consulting to evaluate the hypothesis that IaC is indeed core to DevOps and that adopting IaC principles helps companies of all sizes accelerate software delivery. We define infrastructure as code as a strategy in which the techniques, processes, and tool sets used in software development are leveraged to manage the deployment and configuration of systems, applications, and middleware.

For example, a significant number of testing and deployment defects occur when developers' environments defining the application and underlying infrastructure differ from testing and production environments. Standardizing these environment definitions, putting them under version control, and deploying and configuring the infrastructure and application automatically from the code in version control, yields immediate benefits in consistency, time savings, error rates, and auditability.

To test this hypothesis, Forrester conducted an online survey of 300 Dev and Ops professionals in enterprise companies around the world. Fifty percent of respondents were already using IaC, and 50% were currently implementing or interested in IaC. Comparing responses from these two groups highlights that IaC offers tangible and well-understood benefits, helps companies improve collaboration, reduces wait times and errors, and yields positive business outcomes.

KEY FINDINGS

Forrester's study yielded four key findings:

- **IaC removes friction from the most difficult steps in the software delivery life cycle.** Dev and Ops teams agree: Development, testing, and configuration are the most challenging steps in the delivery process. As these stages are highly dependent on each other, removing friction from them significantly improves delivery timelines.
- **IaC fosters better collaboration between Dev and Ops teams, reducing errors and increasing efficiency.** Dev and Ops teams must improve collaboration in order to improve efficiency, reduce troubleshooting times, and reduce errors introduced between life-cycle stages. Even the smallest configuration error can result in long delays, requiring both Dev and Ops teams to identify and resolve problems, ultimately extending release times.
- **Both experienced IaC users and those still in the planning stages agree on its core benefits and how it improves the business.** Those who are already using IaC reported improved customer satisfaction, and they are expanding their business capabilities and maintaining a competitive edge in the markets they serve. Those using IaC and those considering it both called out improved collaboration as the most important benefit they have already experienced or anticipate from using IaC.
- **Dev and Ops teams must establish common tool sets, but that's just the beginning.** Having common scripting languages is critical for an effective IaC implementation — both experienced and novice users agree. But getting the right tools in place is only one piece of the solution. Both teams must also have properly skilled staff to work with these common languages, and a culture of collaboration around them.

Our research confirms that tools alone do not make IaC work. Businesses need to ensure they have the right people, skills, and processes in place as well if they are to achieve the broadest range of IaC benefits.

Companies Must Deliver Applications Faster

In today's software-driven economy, companies cannot afford to sit idly by and assume they'll one day catch up to their competitors. Consumers are demanding more than ever from the businesses that serve them, and modern businesses need to respond quickly to those demands. To be competitive, companies must be able to quickly release new features to market, proactively create and release entirely new applications, and test features more carefully than ever.

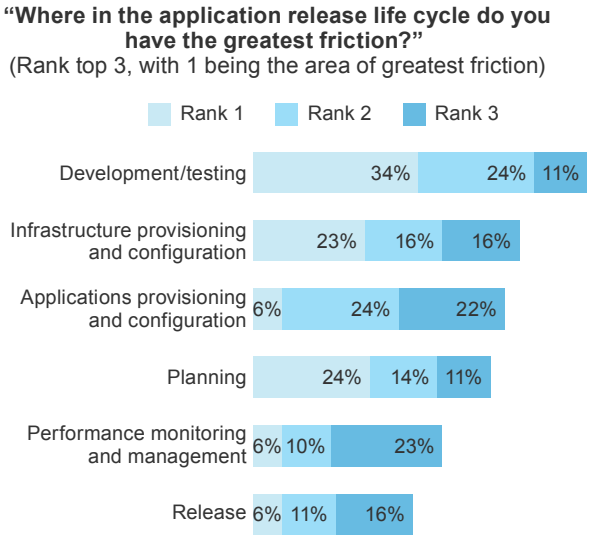
These pressures put a strain on every company's ability to provision, configure, and scale out the application and infrastructure components that make up complex software products. This is especially true for companies moving to Agile development methodologies, where the pressure is even greater on Ops teams to configure and provision apps and infrastructure as quickly as development teams need to test and release new code.

DEVELOPMENT, TESTING, AND CONFIGURATION ARE AREAS OF GREATEST FRICTION

The modern software delivery life cycle consists of several major stages: planning, development/testing, release, infrastructure provisioning and configuration, application provisioning and configuration, performance monitoring, and ongoing management. It includes stages traditionally included in terms such as the software development life cycle (SDLC), application release automation, and configuration management. We combine them together to recognize that software delivery today spans the entire range of Dev and Ops activities, from requirements analysis through configuration and production operations. IaC-aligned processes and tools add value across this entire software delivery life cycle.

Of these stages, survey respondents indicated that they currently have the most friction (i.e., conflicts, errors, and misconfigurations) in development/testing and infrastructure and application configuration (see Figure 1). Respondents told us that these stages, in addition to introducing high levels of friction, also introduce the greatest delays. This means companies need a way to remove friction and delays across all development-test-development sub cycles, and they need faster, more reliable, and more repeatable processes for configuring both apps and infrastructure. In short, modern faster delivery demands better strategies for quickly and consistently setting up development and test environments to reduce wait times and configuration errors between steps.

FIGURE 1
Development And Testing Top Points Of Friction



Base: 300 IT professionals involved with the build and release of software
Source: A commissioned study conducted by Forrester Consulting on behalf of Microsoft, February 2015

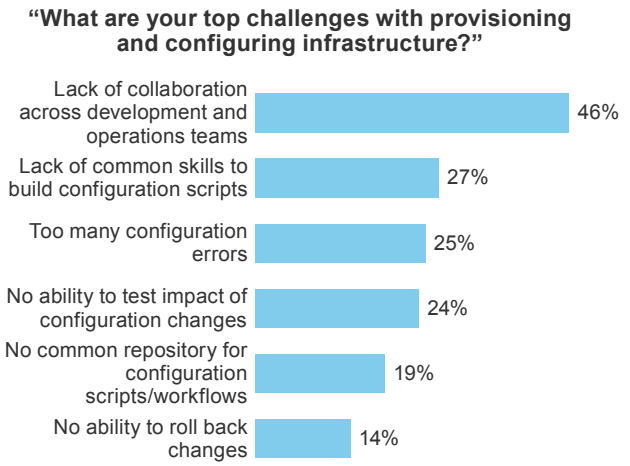
COLLABORATION CHALLENGES BETWEEN DEVELOPERS AND IT OPS ARE THE CORE ISSUE

The top challenge holding Ops and Dev teams back from faster delivery is a lack of collaboration between Dev and Ops teams. This lack of collaboration is exacerbated by a skills mismatch: Dev and Ops teams use different languages to specify configurations, and this leads to miscommunication, misunderstandings, and more frequent errors, especially when development or test environments aren't configured to match production environments (see Figure 2).

Every configuration error (between a developer laptop and a testing machine, for example, or between a testing machine and a production server) introduces delays. Someone must resolve the error, and that usually means that both Dev and Ops must get involved to understand what went wrong, agree on a resolution plan, and implement it. If Dev and Ops don't share a common language or can't easily test the

FIGURE 2

Dev And Ops Challenged By Cross-Team Collaboration



Base: 300 IT professionals involved with the build and release of software
Source: A commissioned study conducted by Forrester Consulting on behalf of Microsoft, February 2015

impact of a configuration change before it's implemented, release times will suffer.

Infrastructure As Code Is Key To Faster Software Delivery

Given these challenges, it's clear that companies must find ways to increase speed that don't kill operational efficiency or introduce new sources of confusion or delay. With limited staff and budget constraints, the most obvious and proven way to speed up any technology process is through automation.

Complexity kills efficiency. If businesses can reduce the complexity at each stage in the delivery life cycle and implement standardized processes that are executed automatically, speed can be increased without decreasing operational efficiency. Automation takes confusion and error-prone manual processes out of the delivery life-cycle stages. We found that while IaC is relatively new to the market, it offers clear automation benefits to Dev, Ops, and, indeed, to the entire business.

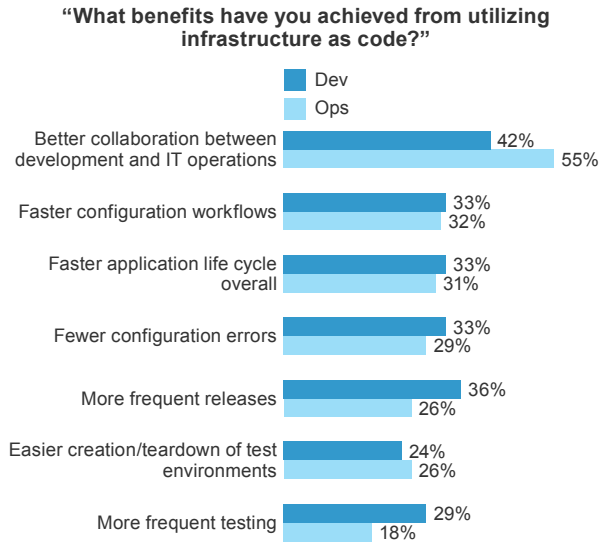
Within our survey we targeted two audiences: those who are currently using IaC, and those who are not yet using it but are gearing up to do so soon. Our comparisons between the two groups revealed many benefits of the use of IaC.

INFRASTRUCTURE AS CODE ADOPTERS REPORT IMPROVED COLLABORATION BETWEEN DEV AND OPS TEAMS

Both Dev and Ops teams agree on the primary benefits of IaC. For both, the top benefit of IaC is improved collaboration, and this means less friction. In practice, IaC allows the Ops team to configure apps and infrastructure repeatedly, faster, and more reliably, thus reducing errors, and allows them to do so using languages and methodologies familiar to developers. This, in turn, means that Dev produces higher-quality code that can be tested more often and released to production faster (see Figure 3). When Dev and Ops work better together, errors go down and release frequency goes up.

FIGURE 3

Improved Collaboration Is A Critical Benefit



Base: 150 IT professionals who utilize IaC
Source: A commissioned study conducted by Forrester Consulting on behalf of Microsoft, February 2015

BOTH NOVICE AND EXPERIENCED INFRASTRUCTURE-AS-CODE USERS AGREE THAT IAC WILL ACCELERATE SOFTWARE DELIVERY

For companies new to IaC, Dev and Ops teams often come to the table with compatible yet potentially competing agendas. The Dev team wants to release more often, test more often, and spend less time waiting for someone to configure apps and infrastructure — they don't want to battle with Ops. Ops teams want better collaboration, too, to

stay in sync with their developers, and they want IaC to help them improve operational efficiency by spinning up test environments faster, reducing troubleshooting times, resolving problems faster, and rolling back changes more easily.

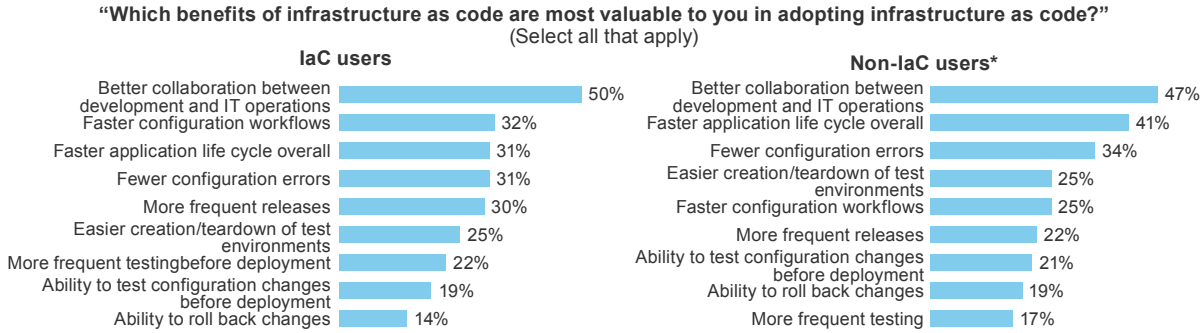
Thus, IaC, implemented properly, not only helps Dev and Ops collaborate with one another better but helps each team achieve its own goals, including increased release frequency for Dev and efficient environment control for Ops (see Figure 4). Both experienced and novice users agree that IaC delivers broader business benefits as well, including:

- **Improved customer experience.** Customers increasingly interact with companies through applications, and customer experience is now critical to competitive success. Companies using IaC can fix problems, release new versions, and enhance applications faster and more reliably because IaC allows them to test new code more frequently, configure components more reliably, and reduce time spent waiting for someone on another team to complete an SDLC task.
- **Expanded ability to reach new markets and customers.** IaC allows Dev and Ops teams to spend less time overall on managing features and updates for existing applications, freeing up time to develop new apps to reach new markets. Companies without IaC spend more time tweaking existing applications and have less time to be proactive with new initiatives. For example, 44% of companies surveyed that are using IaC said that entering new product areas was a top reason for wanting faster software delivery, compared with only 25% of companies not using IaC.
- **Higher team efficiency and output.** The more a company is able to integrate and unify its development and operations teams, tools, and processes, the more efficient it will be. In addition, integration of multiple teams can free up additional Dev resources to focus on new business initiatives. When Dev and Ops work together, instead of at odds, they can release better quality software more frequently.
- **Increased revenue and business performance.** While Dev and Ops teams both understand the technology benefits of IaC, our research shows that Ops users are more aware of the impact IaC can have on business-critical metrics such as increased revenue. Ops users recognize the tangible benefit that better software has on their company's bottom line. When asked about the positive business impacts of using IaC, 55% of Ops respondents selected increased revenue, just slightly behind improved user satisfaction (57%).

Challenges Of Using IaC

Infrastructure as code has incredible potential, but only when combined with organizational change and skills development. The most significant technical hurdle faced during implementation of IaC is getting Dev and Ops teams to agree on a common set of scripting languages and making sure both teams are adequately skilled in using those languages (see Figure 5). This is closely linked to the need for better collaboration overall, since decisions about which scripting languages and tools are best and how to

FIGURE 4
IaC And Non-IaC Users Agree On Benefits



Base: 150 IT professionals who utilize IaC

*Base: 150 IT professionals who do not utilize IaC
Source: A commissioned study conducted by Forrester Consulting on behalf of Microsoft, February 2015

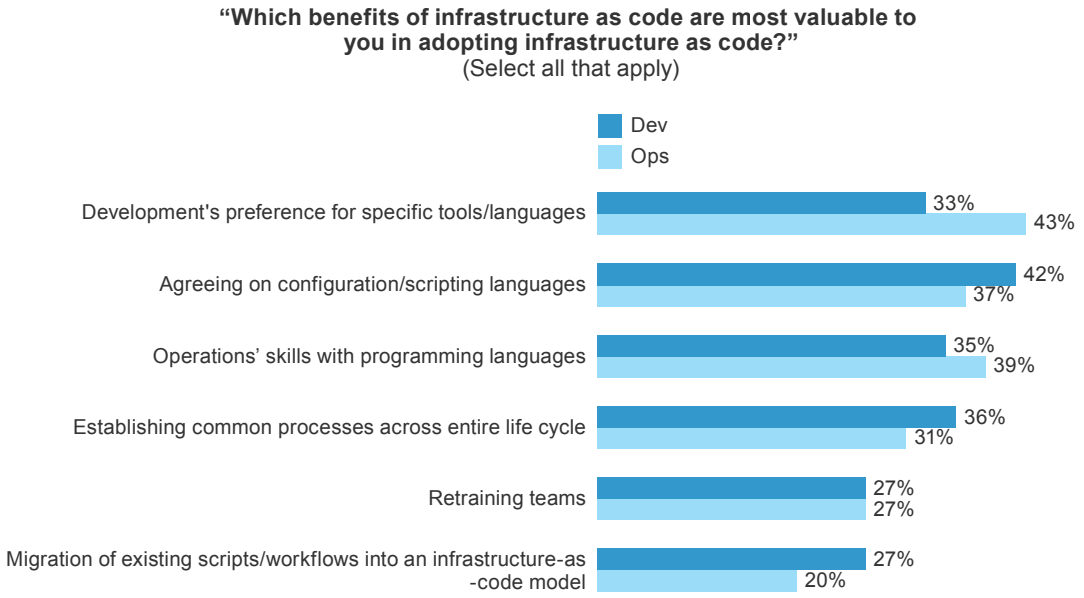
share skills across teams require close communication, cross-training, and trust between Dev and Ops.

Challenges also extend beyond tools and skills to processes. To provide some guardrails for effective use of IaC, companies must establish common processes across the entire development life cycle, from coding through test, release, configuration, and provisioning. Common tools will only be valuable if Dev and Ops can agree on how and when they should be used, and by whom. Indeed, our research found a number of key organizational and business constraints companies must overcome, including:

- **Budget restrictions and organizational resistance to change.** Companies already implementing IaC find the largest organizational challenges stem from budget limitations, as they have no ability to hire or retrain staff, and an overall resistance to change. Ops is more concerned with budgeting issues than Dev. Both teams are fairly united on the organization's resistance to change, and both teams are fairly skeptical about whether the other group has the time or motivation to learn new tools.
- **Lack of expertise in-house to understand and manage an IaC implementation.** For those companies not currently using IaC, lack of in-house expertise is the largest technical hurdle they anticipate as they consider IaC implementation. Nearly a third of companies in the planning stages of IaC think this lack of internal expertise means that they don't understand the value of IaC well enough.
- **Lack of tools, skills, and fear of loss of control.** The Ops team is less confident in its ability to effectively use modern cloud-style IaC languages than Dev (53% versus. 44%). IaC languages are more code-like than script-like, so developers are more comfortable with them in general. Ops is also more concerned with configuration control conflicts, because they have traditionally had all control over configurations and feel they will still be responsible for fixing problems, regardless of who specifies configurations.

As companies reduce their use of traditional provisioning and configuration tools and languages in favor of newer cloud-style design languages, it's important to note that this shift alone will not accelerate software delivery enough. Process and organizational changes must also be a part of any move to IaC.

FIGURE 5
Dev And Ops Must Overcome Scripting And Programming Language Hurdles



Base: 150 IT professionals who utilize IaC
Source: A commissioned study conducted by Forrester Consulting on behalf of Microsoft, February 2015

Key Recommendations

Software drives customer experience, satisfaction, and retention, more than ever. In order to succeed today, market-leading companies must accelerate the pace of software delivery without sacrificing quality or further burdening Dev and Ops teams. Our research confirms that an IaC approach can help companies from any industry or geography meet the demand and create a more collaborative, efficient, and integrated software delivery organization in the process. IaC tools alone can't provide the improvement most organizations need, but if these tools are used in conjunction with collaborative processes and skills transfer, companies will see not only faster, more accurate software delivery, but also happier customers and better business outcomes. While market understanding of the benefits of IaC is still maturing, our research from those using IaC and planning to use it in the near future exposed some lessons for those considering this approach:

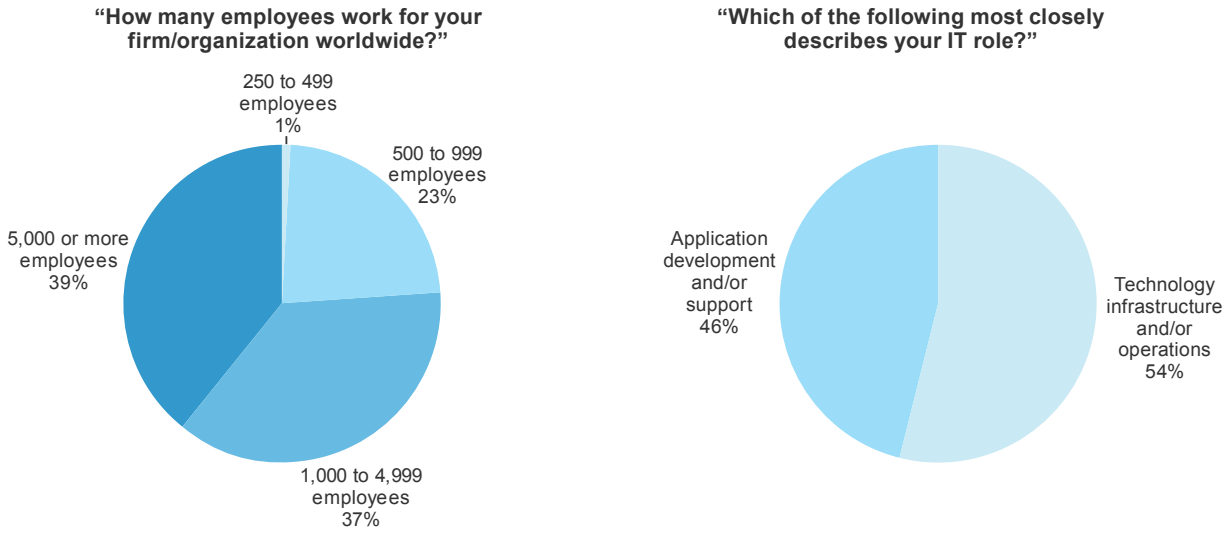
- **Focus the Ops and Dev teams on the long-term business and technical benefits of IaC.** Users of IaC have achieved better collaboration and faster configuration workflows and software delivery life cycles. They've been able to release more frequently, test more often, and have fewer errors that cause friction and delays. Make sure your team understands how these technical benefits translate into satisfied customers and the ability to enter new markets — both of which improve your company's business prospects.
- **Build processes to allow for collaboration and minimize points of friction.** An evaluation of current provisioning and configuration processes is essential to find points of contention between the Ops and Dev teams. Determine where there's agreement on goals, code languages, or tools to implement IaC, and identify any gaps. Be sure you have buy-in across both teams before implementation; shared and common methods for provisioning and configuration will only work if your developers and operations teams feel that they have been consulted, are confident they have the right skills, and believe their counterparts are also fully on board. Reward automators over administrators; focus your efforts on aggressive automation and standardization, using common tools to simplify implementation.
- **Be sure to budget dollars and time for training.** Both Ops and Dev teams told us they lack time and motivation to learn the necessary skills for IaC tools, and they are skeptical that their counterparts have the right skills. Budget time for training, encourage cross-domain knowledge sharing, and help Ops become familiar with coding languages. In a DevOps environment, Dev teams must take on some configuration responsibility, and Ops teams must learn the key tenets of Agile delivery, but each team will only do so if they are given time and encouragement.
- **Start small and learn as you grow. IaC is not an all-or-nothing proposition.** Identify your applications that change frequently or are the cause of too many problems or delays, along with those that aren't tested frequently enough because configuring test environments is too difficult or time-consuming. In the short term, stress efficiency and speed benefits, and look for quick wins that will motivate the teams to deploy IaC more widely. Over time, increased customer satisfaction will drive increased revenues and other business outcomes.

Appendix A: Methodology

In this study, Forrester conducted a global online survey of 300 IT professionals involved with the build and release of software from enterprise organizations (1,000 employees and above) to evaluate their application development process and tools used. Companies surveyed included a mix of 50% who were using infrastructure-as-code tools and 50% who were interested in the tools but not yet using them. Respondents included individual IT contributors, as well as managers or higher from a variety of industries. Respondents were offered a small incentive as a thank you for time spent on the survey. The survey began in January 2015 and was completed in February 2015.

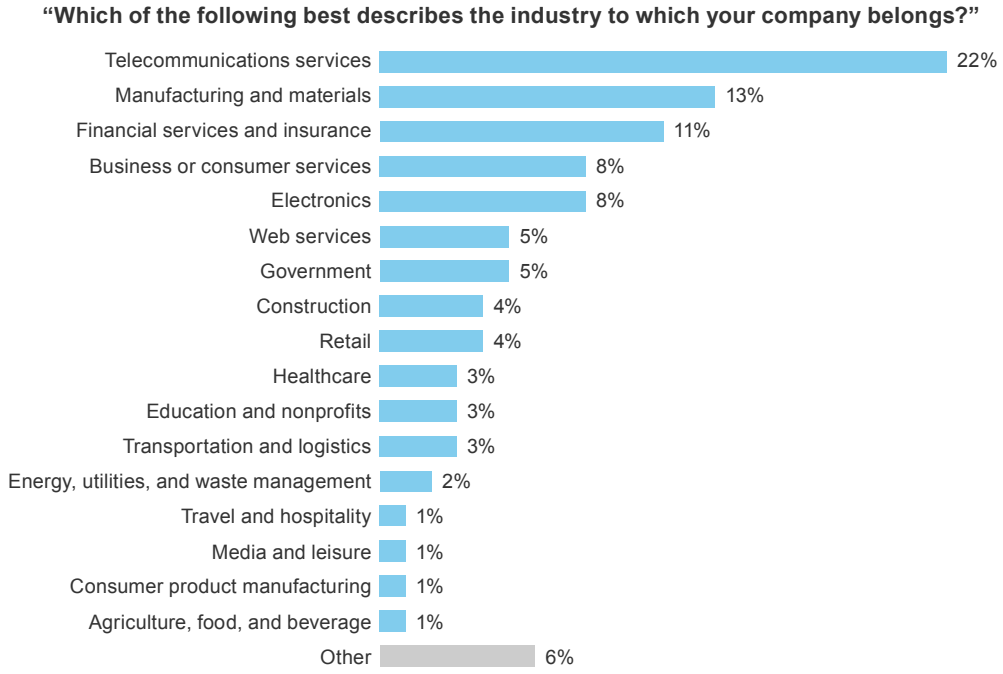
Appendix B: Survey Demographics

FIGURE 6
Company Size And Respondent Role



Base: 300 IT professionals involved with the build and release of software
Source: A commissioned study conducted by Forrester Consulting on behalf of Microsoft, February 2015

FIGURE 7
Respondent Industry



Base: 300 IT professionals involved with the build and release of software

Source: A commissioned study conducted by Forrester Consulting on behalf of Microsoft, February 2015