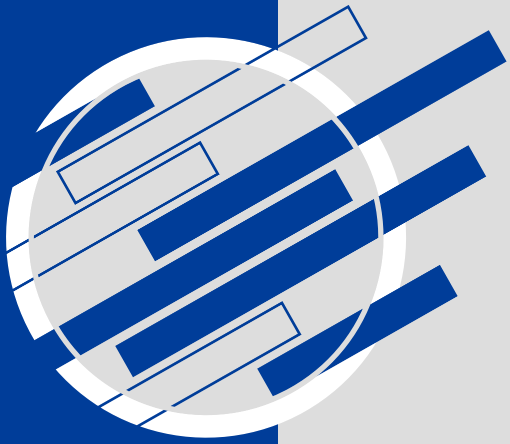
A Forrester Consulting
Thought Leadership Paper
Commissioned By BMC

October 2017

Making Business Workflows First-Class Citizens In The Modern Software Delivery Life Cycle

Enabling DevOps Through Jobs-As-Code



FORRESTER®

Table Of Contents

Project Director:
Christopher Taylor,
Senior Market Impact Consultant

Contributing Research:
Forrester's Infrastructure &
Operations research group

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

Jobs-as-code means handling "jobs" (batch workflow automation for business apps) the same way you manage the other components of an application (i.e., code, logic, infra, test cases, security, etc.). By building jobs in code early, all the manual and creative work is done together. The delivery pipeline can then be run through automation, as the delivery pipeline is essentially stored in an SCM, where all pieces can be built and tested together automatically. This left-shifts the jobs to developers with the other application components.

Batch processing, often overlooked as archaic or boring, is the backbone of managing today's mammoth computing workload. In fact, our custom study revealed that 92% of companies say batch is here to stay, with 74% of operations professionals and 57% of application development respondents expecting use of batch to increase. As DevOps forces organizations to reimagine how they use technology to achieve faster, more frequent, high-quality releases, batch processing must be part of this revolution.

In June 2017, BMC commissioned Forrester Consulting to evaluate the benefits a jobs-as-code approach creates in driving greater efficiency across the software development life cycle. Our hypothesis stated that by adopting jobs-as-code and shifting batch processing left in a code form factor familiar to developers, organizations could then leverage their DevOps philosophies and tooling to drive greater efficiencies across the software delivery life cycle (SDLC). This would help transform batch processing into a first-class citizen in the business technology portfolio.

Forrester surveyed 212 IT application development and operations professionals to test this hypothesis and found that app dev and operations professionals have yet to integrate their skill sets across the SDLC, and they experience organizational and technical challenges in configuration and integrations. As a result, organizations need to evolve their approach, infusing their ops processes throughout the complete SDLC.

KEY FINDINGS

- **All teams recognize provisioning and configuration as a pain point.** Thirty percent of companies cite a lack of automation for configurations and integrations across the complete delivery cycle as a top technical challenge. As well, a third of respondents said that navigating silos was their top organizational challenge due to its negative impact on delivery velocity.
- **Jobs-as-code is on the rise.** Ninety-three percent of respondents see value in using jobs-as-code to improve software delivery. There is also a tremendous opportunity for growth with 46% of companies expressing interest or planning to implement jobs-as-code.
- **Lack of collaboration hinders adoption of jobs-as-code.** Sixty-five percent of respondents experience friction between app dev and ops teams in development and testing.
- **Jobs-as-code improves the quality of code in production.** Forty-five percent of companies report or anticipate fewer errors in production with jobs-as-code, and 44% cite fewer errors in code. Forty-six percent said jobs-as-code increases efficiency between ops and app dev teams.

Product Teams Are Siloed Across The Life Cycle — Creating Islands

Enterprise IT product teams have traditionally operated in functional silos, addressing designated activities of the software delivery cycle. This siloed approach to work creates problems that manifest as slow manual handoffs, rework, and delayed time to resolution during service downtime.[1] If organizations expect to succeed in today's digital landscape, they must speed up release cycles of applications and services to offer a better customer experience.[2] Creating one transparent pipeline from development to production demands integrating the work capabilities of essential product teams. Organizations that achieve this enjoy increased speed, responsiveness, and quality of delivery.

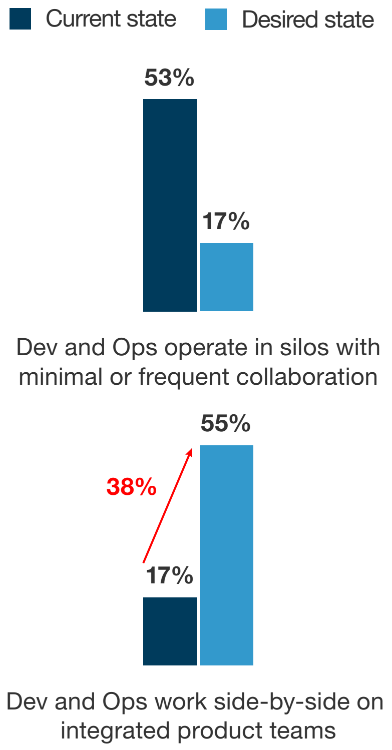
Our survey of application development and operations professionals revealed that most companies have yet to integrate their app dev and ops teams into product-centric, cross-functional teams. This results in friction across the application life cycle. Specifically, we found that:

- **App dev and ops teams currently operate in silos but desire integration.** Over half of app dev and ops teams currently work in silos with minimal or frequent collaboration. According to survey results, 47% of respondents agree that once code is handed over to operations, developers have little involvement with production or even knowledge of how their code performs. However, IT decision makers show a readiness to move away from this approach and better integrate expertise. In fact, 55% desire a state in which dev and ops work side-by-side on a single development team (Figure 1).
- **Lack of collaboration delays the SDLC.** Integrated product teams involve IT groups that collaborate throughout the delivery cycle, yet collaboration is lagging. This is particularly evident in prerelease activities like development and testing as well as provisioning and configuration (Figure 2).
- **Friction between teams affects quality and performance.** Forty-eight percent of respondents in our study said they encounter quality and performance issues with code in production. Siloed teams make it more difficult to use this data to improve future processes. It is easier for integrated teams to model the target environment for batch across all stages of the life cycle (dev, testing, and production), continuously test, and use the same configuration for production. This helps organizations avoid these types of quality and performance issues.

Figure 1

APP DEV AND OPS COLLABORATION

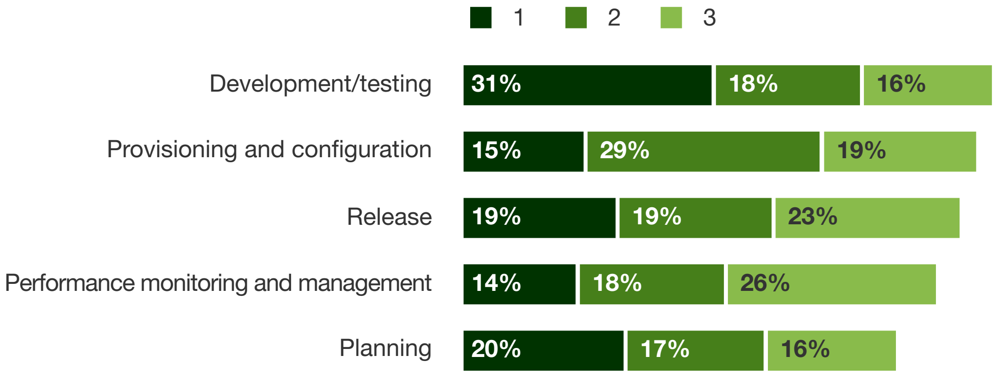
"What best describes the current state of collaboration between your company's application development and IT operations teams? What is your desired state of collaboration?" (Select one per column)



Base: 212 IT professionals with direct responsibility for the software development life cycle at enterprises in North America and EMEA
Source: A commissioned study conducted by Forrester Consulting on behalf of BMC, August 2017

Figure 2: Friction Across The SLDC

"Where in the application delivery and deployment process do you have the greatest friction between dev and ops teams?"
(Rank top three)



Base: 212 IT professionals with direct responsibility for the software development life cycle at enterprises in North America and EMEA
Source: A commissioned study conducted by Forrester Consulting on behalf of BMC, August 2017

48% of development teams encounter quality and performance issues with code in production.

Automation, Configuration, and Integrations — The Secret To Success

Dev, test, and ops teams alike are more readily recognizing the benefits from consistent automation, configuration management, and integration approaches. Automated provisioning of testing environments removes delays and uncertainties, assuring configurations are deployed. It also increases the certainty of success and assures that the configurations used in testing reflect in production to ensure consistency and quality.

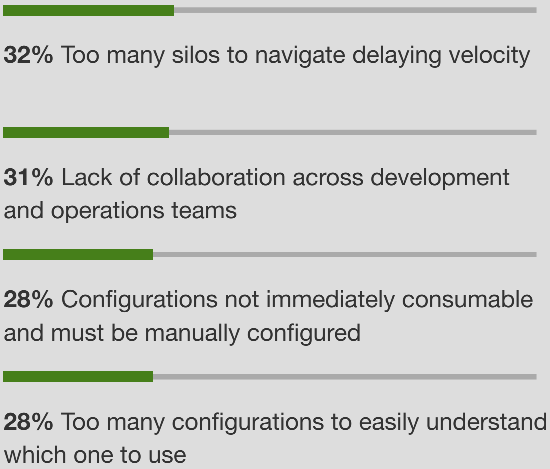
Our survey revealed most companies face significant hurdles in determining how to select the configuration solution to deliver consistency across the complete SDLC. Specifically, we found that:

› **Manual configurations and siloed capabilities drive a need for greater collaboration.** Survey results show that app dev and ops have yet to evolve their environments to avoid key organizational provisioning and configuration challenges such as siloed capabilities and lack of collaboration. For example, navigating silos is the top organizational challenge respondents face in provisioning and configuration — 32% said the sheer number of silos slows delivery velocity. In addition, manual configurations create an added layer of complexity to provisioning environments (Figure 3).

Figure 3: Top Provisioning Challenges

"What are the organizational challenges your company faces with provisioning and configuration?" (Select all that apply)

TOTAL RESPONDENTS



App dev	Ops
• Too many silos to navigate, delaying velocity • Lack of collaboration across dev and ops team • Configurations not immediately consumable and must be manually configured	• Limited awareness of processes across dev and ops teams • Ineffective environment for batch processing • Too many configurations to easily understand which one to use

Base: 212 IT professionals with direct responsibility for the software development life cycle at enterprises in North America and EMEA
Source: A commissioned study conducted by Forrester Consulting on behalf of BMC, August 2017

- **App dev and ops teams have different challenges.** When exploring core challenges through the lens of app dev and ops respondents, application development professionals show responses consistent with the total group. In contrast, operations demonstrated greater recognition that current configuration environments are ineffective for batch processing. They also report lacking awareness of processes across teams.
- **Configuration, integration, and automation challenges drive delivery inefficiencies.** Our survey found that 39% of all respondents have too many configurations, increasing complexity and confusion about which configuration to use. Further, the adoption of disparate tools across the delivery cycle requires increased technical integration and maintenance. All of this negatively affects key areas such as release, continuous integration/continuous deployment (CI/CD), and workload automation. Thirty percent of respondents reported an absence of automation across the life cycle.[3]

30% of IT decision makers cite absence of automation across the delivery cycle as a top technical challenge of provisioning and configuration.

Continuous Delivery Mandates Automation, Testing, And Full Pipeline Visibility

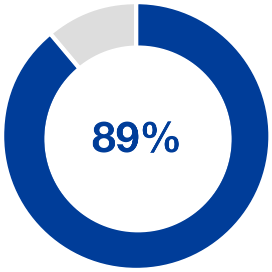
According to a recent Forrester report, if organizations are to transform the way they deliver solutions to their customers, they will need to shift from a traditional development and delivery model to a continuous delivery model — one that infuses ops processes into the development workflow and mandates complete automation. Further, this model requires shared knowledge between dev and ops teams and leverages continuous testing across the life cycle.[4]

Our survey results indicate that respondents are already on their way to exploring these capabilities:

- **Almost all companies have moved operational processes into the delivery cycle.** In fact, only 4% of companies are not planning to move operational processes into the SDLC.
- **Nearly half of all app dev and ops decision makers are utilizing higher levels of automation for testing, and 41% are increasing adoption of DevOps principles.** As part of this, companies are infusing ops activities — such as testing, performance management, and batch scripts — into the development workflow.
- **Companies are increasing usage of batch processing as part of their delivery of differentiated business services across all environments — mainframe to cloud.** Seventy-four percent of ops professionals and 57% of app dev professionals have increased their usage of batch processing over the past two to three years.
- **Automated movement of code is increasingly important to the SDLC.** Survey respondents consider automation central to the software delivery cycle; 89% say automation for jobs/ batch processing is important or very important. This includes more automated movement of code throughout the life cycle, which eliminates manual handoffs and increases the speed of releases (Figure 4).

Figure 4

Automation for jobs/batch processing is important to the SDLC



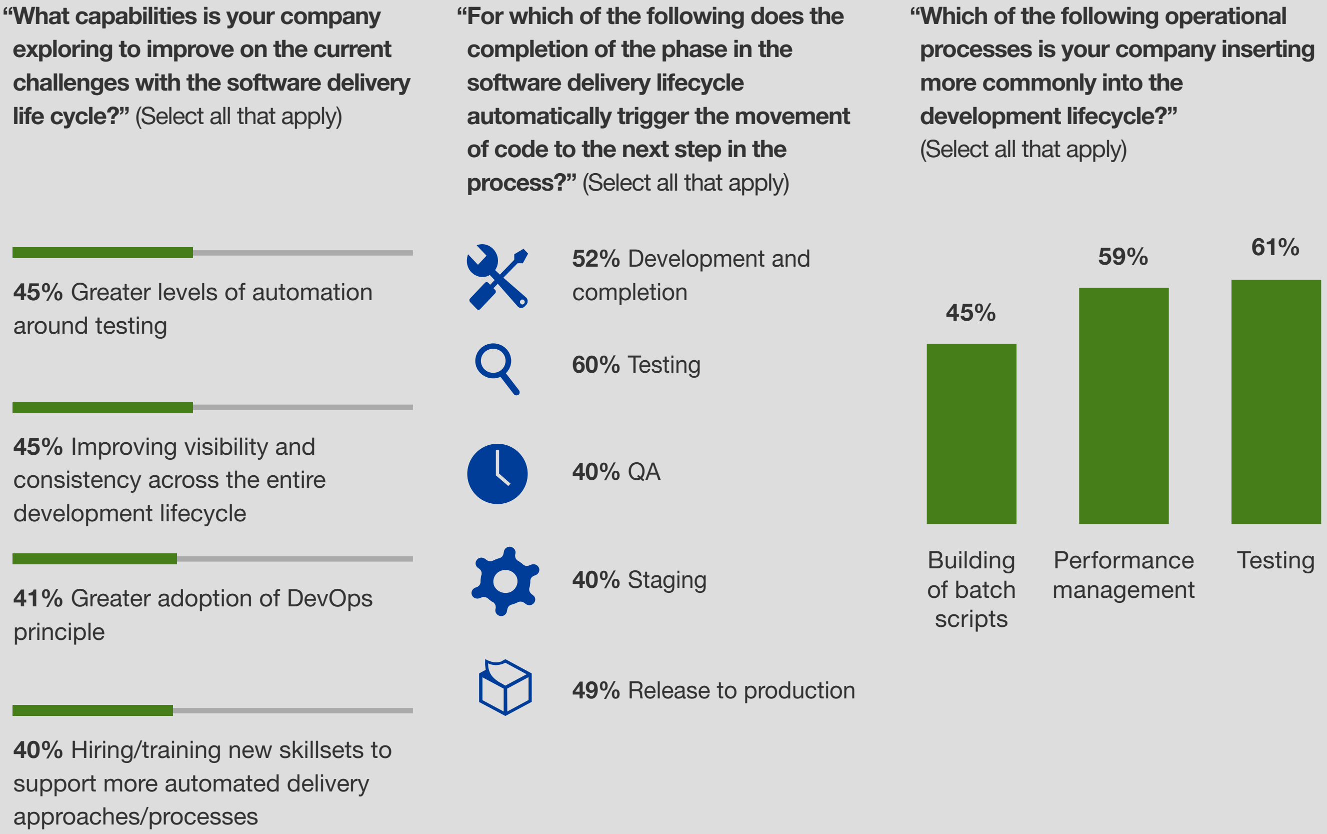
89% say automation for jobs/batch processing is important or very important for the software delivery lifecycle.

Base: 212 IT professionals with direct responsibility for the software development life cycle at enterprises in North America and EMEA
Source: A commissioned study conducted by Forrester Consulting on behalf of BMC, August 2017

Jobs-As-Code Drives Greater Efficiency And Quality Across The Life Cycle

Batch processing grew up in the mainframe era and has found new relevance managing today's mammoth workloads, whether on the mainframe or in the public or private cloud. In an era of velocity, batch processing has been ignored and manually built, often based on the perception that it is no longer relevant or not a first-class citizen in the SDLC. Introducing jobs-as-code as a DevOps best practice, in which automation rules are created by the developer, and transitioning batch to the proven SDLC automation toolchain allows development, testing, and automation of the complete business workflow life cycle. This solves the primary issues identified by our survey respondents (Figure 5). The adoption of batch by developers requires an environment in which they can operate — code.

Figure 5: Continuous Delivery Requires Automation, Testing, And Visibility



"What capabilities is your company exploring to improve on the current challenges with the software delivery life cycle?" (Select all that apply)

"For which of the following does the completion of the phase in the software delivery lifecycle automatically trigger the movement of code to the next step in the process?" (Select all that apply)

"Which of the following operational processes is your company inserting more commonly into the development lifecycle?" (Select all that apply)

45% Greater levels of automation around testing
45% Improving visibility and consistency across the entire development lifecycle
41% Greater adoption of DevOps principle
40% Hiring/training new skillsets to support more automated delivery approaches/processes
52% Development and completion
60% Testing
40% QA
40% Staging
49% Release to production
45%
59%
61%
Building of batch scripts
Performance management
Testing

Base: 212 IT professionals with direct responsibility for the software development life cycle at enterprises in North America and EMEA
Source: A commissioned study conducted by Forrester Consulting on behalf of BMC, August 2017

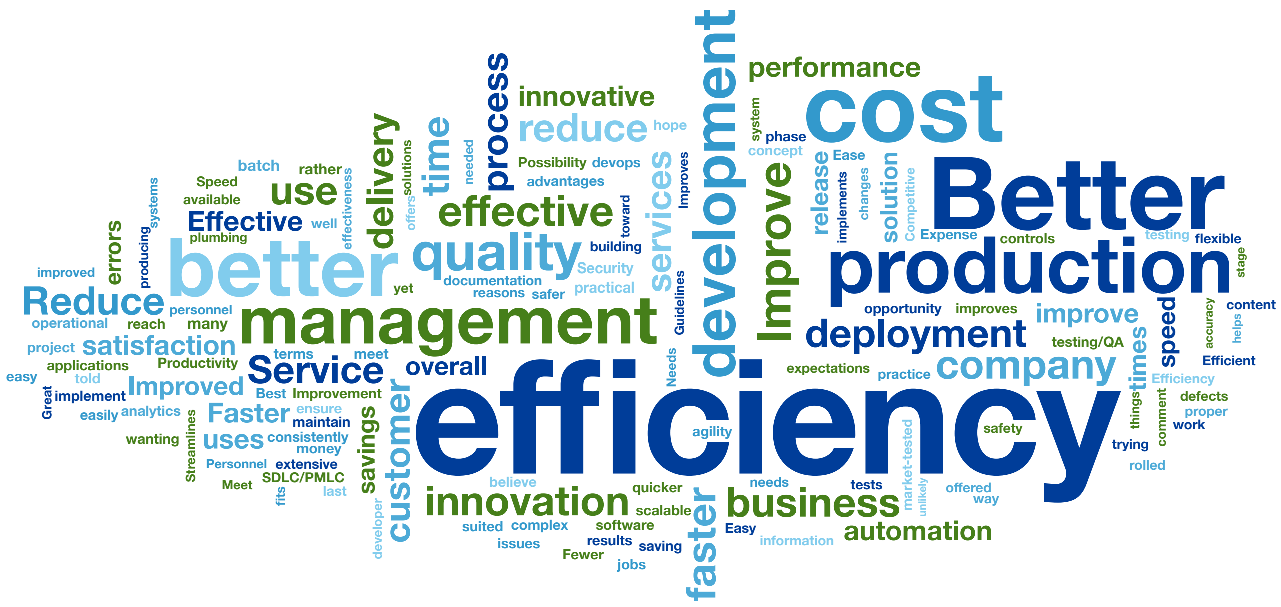
For this study, we define jobs-as-code as handling “jobs” (batch workflow automation for biz apps) the same way you manage the other components of an application (i.e., code, logic, infra, test cases, security, etc.). By building jobs in code early, all the manual and creative work is done together. The delivery pipeline can then be run through automation, as the delivery pipeline is essentially stored in a software configuration management tool (SCM), where all pieces can be built and tested together automatically. This left-shifts the jobs to developers with the other application components. When asked about the value of jobs-as-code, our survey uncovered the following:

- **IT decision makers see promise in jobs-as-code.** Ninety-three percent of respondents see value in using jobs-as-code to improve the software delivery cycle. The increase in batch jobs in conjunction with automation of the batch processing has the potential to yield even greater value for organizations in process improvements.
- **Two out of five companies are interested in implementing jobs-as-code for the SDLC.** Nearly half of companies are interested in or planning implementation of jobs-as-code, taking the lead from the 35% that have already or are in the process of implementation.
- **Speed, efficiency, and quality drive jobs-as-code adoption.** When asked about the primary drivers for implementing jobs-as-code, survey respondents cited the ability to accelerate stages of delivery, reduce costs, increase efficiency, and allocate increased time to innovation. These drivers act as critical components of faster, more reliable delivery. (Figure 6)

35% of companies have implemented or are in the process of implementing jobs-as-code; 46% are planning or interested.

Figure 6

“What is the primary reason why your company has chosen to utilize jobs-as-code?”



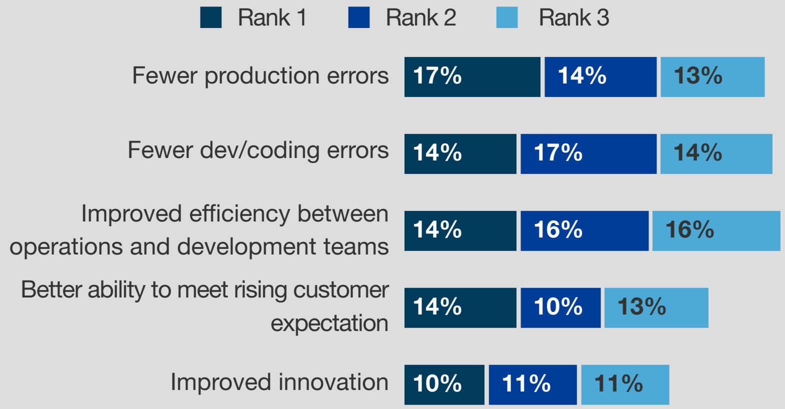
Base: 212 IT professionals with direct responsibility for the software development lifecycle at enterprises in North America and EMEA
Source: A commissioned study conducted by Forrester Consulting on behalf of BMC, July 2017

- **The benefits of jobs-as-code are clear.** According to our study, jobs-as-code supports business objectives by improving quality of code, accelerating release cycles, and equipping companies to meet customer expectations. Companies cite quality improvements like fewer production and coding errors as top business advantages of jobs-as-code. In addition, 46% report improved efficiency between development and operations teams, and 37% experience faster application life cycles (Figure 7).
- **Jobs-as-code benefits both app dev and ops teams.** Jobs-as-code provides benefits to both ops and dev teams while delivering business advantages. Ops professionals can deliver batch with better quality, including decreased production failure rates — supporting rising customer expectations. App dev professionals will be able to accelerate life cycles, with immediate results such as testing and reducing defects that appear in production. Building jobs-as-code into the automation of the complete life cycle delivers efficiency and effectiveness gains, allowing dev and ops to accelerate velocity while assuring quality for the business (Figure 7).

> "Automation of code would **reduce manual tasks** that require intervention such as slow ticket entry and approvals required to promote code at each stage, from development through deployment. **This saves valuable time and resources and ultimately reduces project costs.** The manual intervention can slow deployment by days on each project."
>
> *IT manager at a large healthcare company*

Figure 7

"What do you see as the top business advantages for using jobs-as-code?"



App dev

- Faster application lifecycle Less dev/coding errors
- Fewer dev/coding errors
- Improved efficiency between ops and dev teams
- Fewer production errors
- Better ability to meet rising customer expectations

Ops

- Better ability to meet rising customer expectations
- Improved innovation
- Faster application lifecycle
- Improved efficiency between ops and dev teams
- Increased revenue

Base: 212 IT professionals with direct responsibility for the software development life cycle at enterprises in North America and EMEA
Source: A commissioned study conducted by Forrester Consulting on behalf of BMC, August 2017

"What benefits have you achieved, or do you expect to achieve, by adopting jobs-as-code?"

- **37%** Faster application life cycle overall
- **36%** Fewer configuration errors
- **35%** Greater success implementing batch into production
- **31%** Ability to test configuration changes before deployment

Key Recommendations

Batch processing is crucial to managing today's modern computing workloads. From mainframe to cloud, batch continues to grow and evolve — and is here to stay. Yet, in an era of velocity, where speed is pivotal to business success, batch has been primarily ignored. DevOps rallies integrated teams to collaborate throughout the life cycle for gains in speed and quality; batch processing and jobs-as-code are part of this transformation. Organizations in their quest to satisfy the business' insatiable appetite for speed can no longer treat batch processing as anything other than integral part of their DevOps toolchains. They must do so in a form factor that allows participation from the developer through to production delivery. To participate in the complete life cycle, from development to production, batch needs to be treated as code, like any other artifact. To realize the benefits of jobs-as-code organizations must:

Treat everything, including jobs, as code. Effective DevOps practices have transitioned everything to code — the same is true of batch. Code is managed through the SDLC processes that allow effective governance and oversight to all concerned. Batch, like any other software, should be developed as code and managed via the same SDLC as all software in a highly automated manner. Behaviors that support the transition to everything as code and automation should be encouraged across the organization.

Leverage DevOps practices for batch processing. DevOps is built on the concept of multidisciplined integrated teams that are focused on business outcomes and support this with automation of the SDLC, deep automated testing, and automated deployment. These practices are equally applicable to batch processing where batch processes are often highly integrated and may involve multiple systems. Changes should be kept small and deployed frequently following testing to assure defects are identified in development, not production. This should include modeling the target production environment(s) across all stages of the life cycle (dev, testing, and production), continuously testing, and, when complete, transitioning to production.

Automate provisioning and configuration across the life cycle. Evaluate your current provisioning and configuration management processes to identify where there are inefficient handoffs, multiple configurations, or no transparency. This should include agreement and delivery of consistent processes for configuration management and automation of the processes to deliver configurations. The processes should support the development and delivery of consistent configurations across all stages of the SDLC — development, testing, staging, and production — and should leverage common tools across all environments.

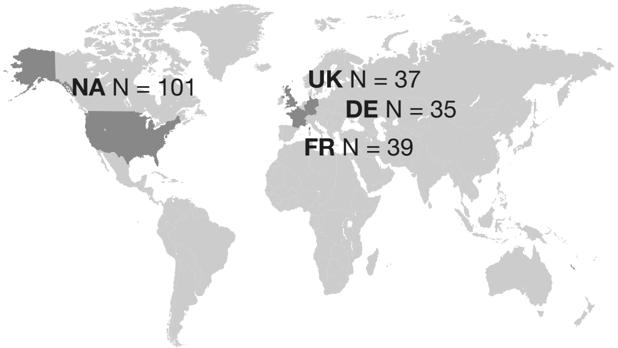
Start small and evolve. Jobs-as-code will require a learning curve with changed processes, roles, responsibilities, and handoffs. To get started, begin with a pilot application, forming an integrated team that has the dev and ops skills required to effectively deliver. The team will provide the templates that can be leveraged by those who adopt at later stages.

Appendix A: Methodology

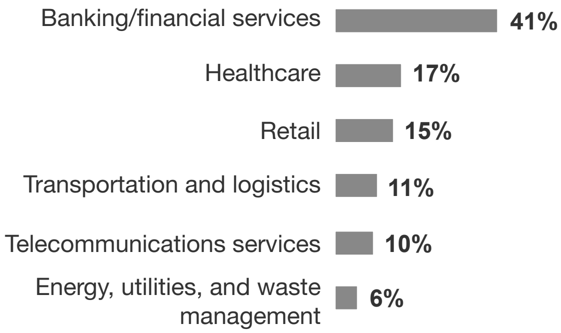
In this study, Forrester conducted an online survey of 212 IT application development and operations professionals across industries. Questions provided to the participants asked about the software delivery life cycle, challenges they face, and use of jobs-as-code. Companies surveyed in North America had 1,000 employees or more, and companies in EMEA had 500 employees or more. Respondents were offered a small incentive as a thank you for time spent on the survey. The study was completed in August 2017.

Appendix B: Demographics/Data

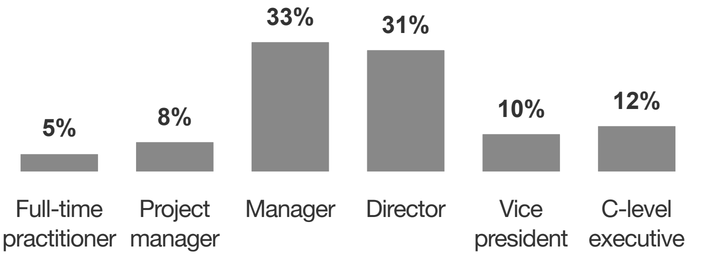
COUNTRY



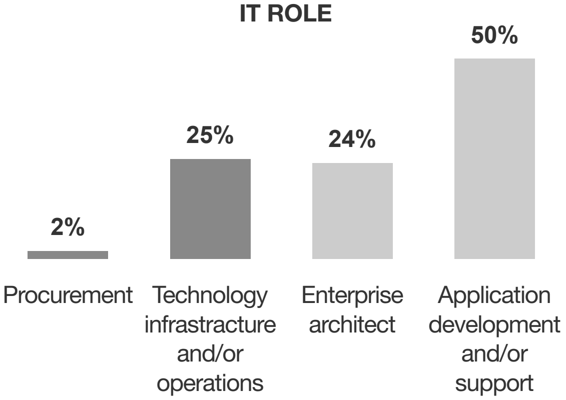
INDUSTRY



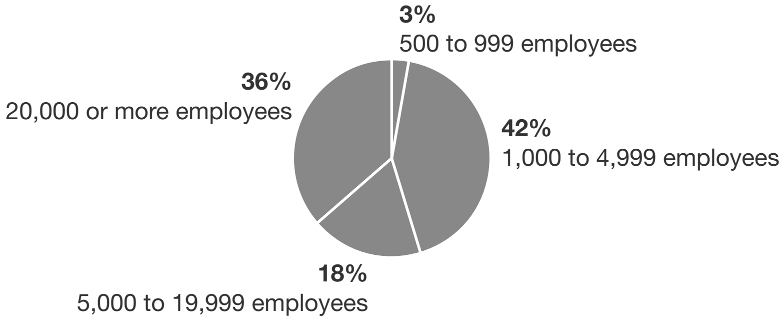
RESPONDENT LEVEL



IT ROLE



COMPANY SIZE



Base: 212 IT professionals with direct responsibility for the software development life cycle at enterprises in North America and EMEA
Note: Percentages may not total 100 because of rounding
Source: A commissioned study conducted by Forrester Consulting on behalf of BMC, August 2017

Appendix C: Endnotes

[1] "Organize And Staff I&O Pros For Successful DevOps Practices," Forrester Research, Inc., August 8, 2017.

[2] "The Digital Business Imperative," Forrester Research, Inc., February 15, 2017.

[3] Enterprise Automation is dismal in multiple key areas. Source: "Six Trends That Will Shape DevOps Adoption In 2017 And Beyond," Forrester Research Inc., August 3, 2017.

[4] "Case Study: Dynatrace's Journey Toward Delivering Business Transformation," Forrester Research, Inc., August 7, 2017.