Gartner

Hype Cycle for Agile and DevOps, 2020

Published: 15 July 2020 **ID:** G00441807

Analyst(s): George Spafford, Joachim Herschmann

DevOps initiatives must be grounded in customer value and will leverage a range of people, processes and technologies that span the software delivery value stream. I&O leaders must pursue organizational learning, continual improvement and automation strategies to deliver the required capabilities.

Table of Contents

List of Tables

List of Figures

Analysis

What You Need to Know

DevOps is a customer-value-driven approach to deliver solutions using agile methods, collaboration and automation. DevOps emphasizes people, culture and collaboration between development, operations and other stakeholders to improve the delivery of customer value. DevOps implementations seek to continually improve the flow of work by removing constraints with the intent of improving the delivery of customer value as a result. In the 2019 Gartner DevOps Adoption and Implementation Challenges Survey, 65% of respondents say they are piloting or in the early stages of adoption. Twenty-nine percent report they have adopted DevOps, 8% plan to by the end of 2021 and only 8% report no plans. DevOps has moved from being a emergent perspective for organizations requiring high-speed change to being a proven collection of practices and tools that can enable all organizations to deliver customer value faster and/or more predictably.

Many approaches and technologies are used to support DevOps, resulting in high DevOps diversity, both from company to company and from one DevOps implementation to another, as teams seek to improve the flow of work and automate risk-mitigating controls. This means technology adoption decisions, which may hinge on factors like the ability of a tool to integrate with others, must be made based on an understanding how the technology enables the flow of value, the ability to scale and the mitigation of risks.

Scaling DevOps efforts will require management of the overarching value stream and optimizing aspects of human factors, organization, process, technology and information exchanges. Organizational learning, continual improvement and the effective use of automation to break constraints will be critical. Communities of practice, DevOps Dojos, job rotation, brown bag lunches and internal tech conferences can all help to varying degrees with enhancing organizational learning, and we recommend a strategy that leverages multiple approaches (see “DevOps Requires Faster Organizational Learning”). The product and platform model can also aid organizations seeking to scale by creating teams that provide shared capabilities across the business-facing product teams (see “How to Scale DevOps by Building Platform Teams”). To improve the delivery of value, the platforms must be managed as products (see “How to Manage and Market Platforms as Products for DevOps Teams”).

For more information about how peer I&O leaders view the technologies aligned with this Hype Cycle, please see “2019-2021 Emerging Technology Roadmap for Large Enterprises.”

The Hype Cycle

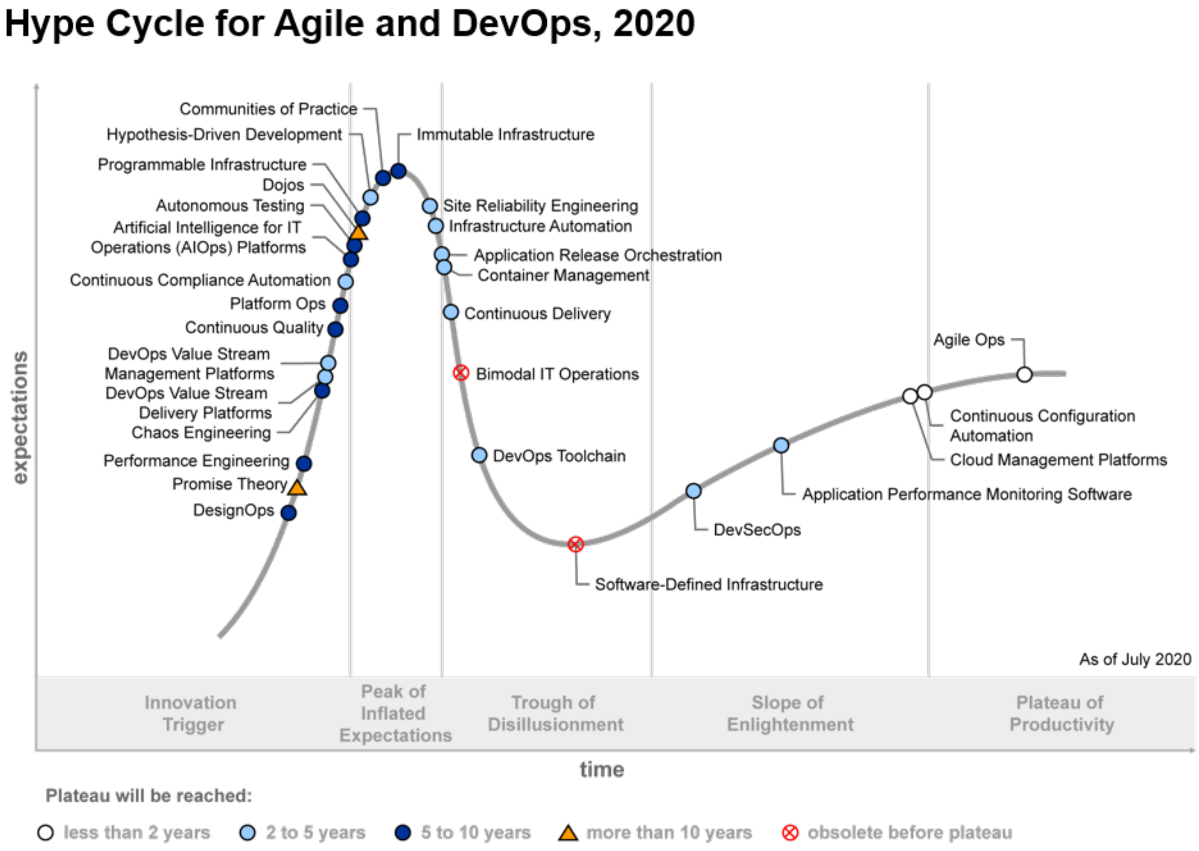
The DevOps Hype Cycle includes key approaches and technologies used to support a DevOps initiative. DevOps continues to grow, and the percentage of respondents saying they do not have any plans to adopt DevOps has dropped from 28% in 2016 to only 8% in 2019. New technologies and approaches are being introduced on an increasingly frequent basis. Clients report that the three largest hurdles to DevOps success are a shortage of experienced staff (55%), organizational culture (51%) and insufficient training/staff development (35%). DevOps leaders must recognize that multiple disciplines must work together to deliver customer value at speed and at scale while managing risk.

DevOps practitioners and IT leaders must identify DevOps technology and process focus areas to improve the flow of customer value, and then map them against Gartner's Hype Cycle to get an understanding of both capabilities and maturity. Early adopter organizations will find technologies that are poised to change the delivery of IT through unique DevOps solutions. The innovation profiles in this research are also a valuable guide for more mature, proven technologies.

When DevOps technology choices are driven from specific organizational teams without representing the entire DevOps scope and strategy, there may be some local optimization that is unavoidable. There will be opportunities where emergent architectures will evolve and refactoring will be needed to manage debt. Leaders must devise a DevOps strategy and create a culture that enables organizational learning through the use of communities of practice to continually evaluate technologies that can be used to support a DevOps initiative. Avoid creating tool and capability "islands" that inhibit the delivery of an effective DevOps practice. Toolchains are evolving to maximize the flow of work and have three paths forward — value stream delivery platforms (VSDPs), value stream management platforms (VSMPs) and niche tools (see "The Future of DevOps Toolchains Will Involve Maximizing Flow in IT Value Streams").

Gartner clients are emphasizing the use of agile and DevOps as their "go forward" strategies for application and product development. Methodology items in this Hype Cycle reflect practices that can be used to enhance an agile organization's effectiveness. It is important to note that the challenges are not purely related to technology and tool shifts, but to a broad shift in a culture that must extend beyond the IT organization (see "Agile and DevOps Primer for 2020"). DevOps adopters are achieving positive results in improving time to value, thus enabling their customers to pursue opportunities and respond to threats more quickly (see "How DevOps Can Deliver Continual Customer Value Faster").

Figure 1. Hype Cycle for Agile and DevOps, 2020



Source: Gartner
ID: 441807

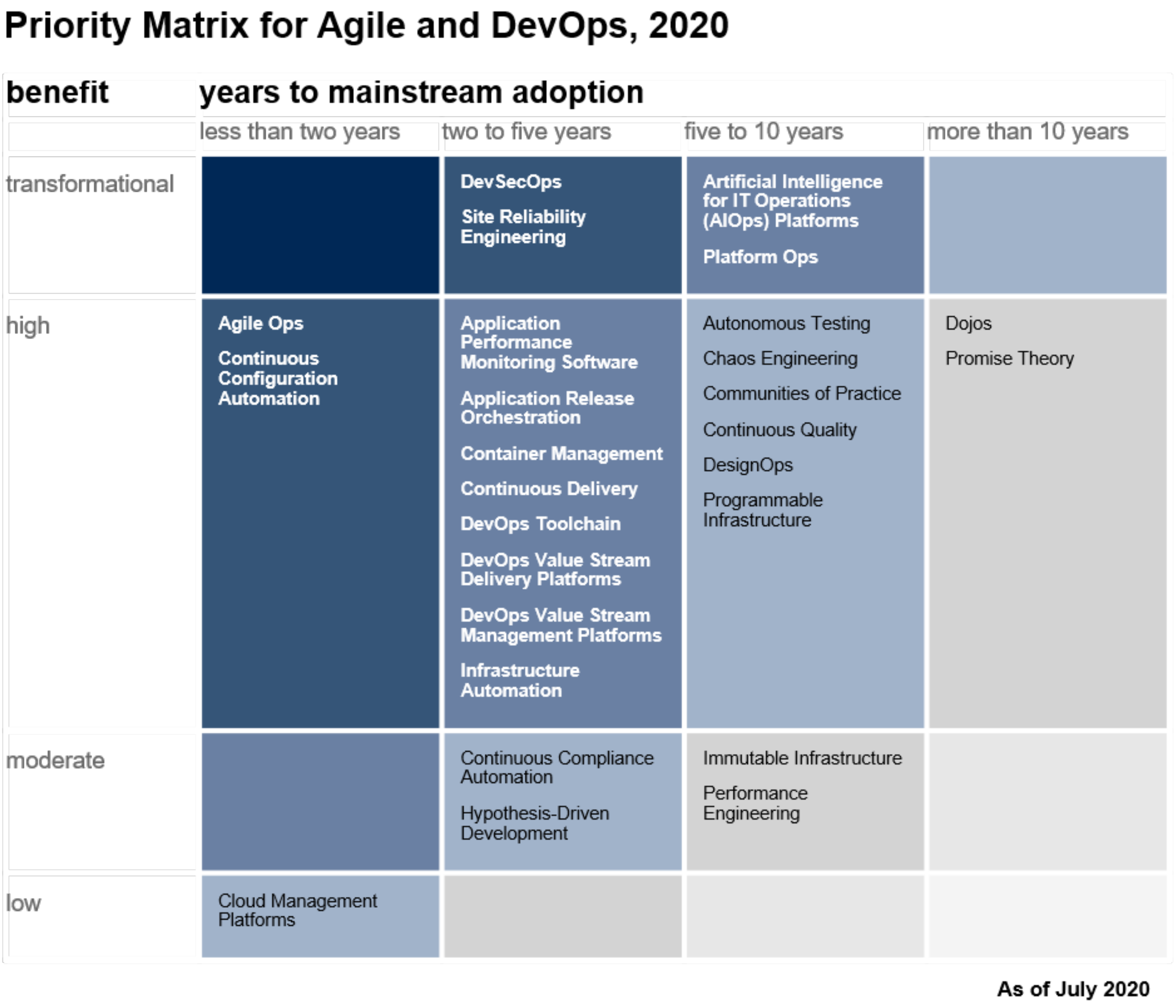
The Priority Matrix

The Priority Matrix maps the time to maturity of technologies and frameworks in an easy-to-read grid format. It answers two high-priority questions:

- How much value will an organization receive from an innovation?
- When will the innovation be mature enough to provide this value?

It is important to note that, for most of the profiles listed, the truly transformative impact is delivered by interlocking technology adoption with people and process frameworks that are aligned to a clear business objective. As highlighted in Figure 2, the years to mainstream adoption for most of the innovation profiles is two or more. Generally, this has more to do with IT organizational readiness and process maturity than with product technical capabilities. More mature IT organizations will successfully adopt automation and transform their efficiency, reliability and predictability faster than indicated in this matrix. Conversely, cultural resistance, unrealistic expectations, a lack of collaboration and a lack of process discipline will slow the time to adoption for many IT organizations.

Figure 2. Priority Matrix for Agile and DevOps, 2020



Priority Matrix for Agile and DevOps, 2020

benefit	years to mainstream adoption			
	less than two years	two to five years	five to 10 years	more than 10 years
transformational		DevSecOps Site Reliability Engineering	Artificial Intelligence for IT Operations (AIOps) Platforms Platform Ops	
high	Agile Ops Continuous Configuration Automation	Application Performance Monitoring Software Application Release Orchestration Container Management Continuous Delivery DevOps Toolchain DevOps Value Stream Delivery Platforms DevOps Value Stream Management Platforms Infrastructure Automation	Autonomous Testing Chaos Engineering Communities of Practice Continuous Quality DesignOps Programmable Infrastructure	Dojos Promise Theory
moderate		Continuous Compliance Automation Hypothesis-Driven Development	Immutable Infrastructure Performance Engineering	
low	Cloud Management Platforms			

As of July 2020

Source: Gartner
ID: 441807

Off the Hype Cycle

The following innovation profiles were not included in the 2020 DevOps Hype Cycle:

- Application performance monitoring suites — has been renamed application performance monitoring software.
- DevOps toolchain orchestration — has been replaced by value stream delivery (VSD) and value stream management (VSM).

- Mobile DevOps tools — Mobile DevOps tools have largely been absorbed into software test automation tools and DevOps platforms. Apple and Google also are providing more of the tooling and services as part their development stacks. There are still specialized tools to ensure consistent DevOps for mobile apps, but the category is now obsolete before the plateau.
- Software-defined networking (SDN) — SDN remains a major topic of discussion in the networking market; however, true SDN technologies have not achieved any significant market traction. The hope that SDN would result in the abstraction of hardware and software and foster independent software innovation never came to fruition.

On the Rise

DesignOps

Analysis By: Brent Stewart

Definition: DesignOps is a set of operational practices that enables centralized design team management and product-level delivery of design assets. The organizational management side of DesignOps focuses on strategic alignment to the business, operations for the central design function, and career development. The product delivery side of DesignOps combines UX, product management and technology operations to enable efficient and DevOps-compatible plans, estimates and processes that increase quality, enable collaboration, and feed ongoing innovation.

Position and Adoption Speed Justification: DesignOps introduces formalized approaches to governance, operations and people management that have been long absent from design disciplines, including UX teams. As a set of easy-to-use operational standards, DesignOps continues to gain in popularity as digital product companies (e.g., Airbnb, Adobe and InVision) and agencies alike discover the tremendous value of a proven operational approach for UX team management and design delivery on product teams. While UX team management is an important component of DesignOps, its growth is due — primarily — to the value it creates during the delivery of design assets for one or many digital products. Here, DesignOps does not alter the core skills and activities of a UX team, rather it reorganizes them in a way that supports ongoing feature enhancement and idea generation without interrupting the continuous workflow of development teams. It represents the first widespread implementation of operational methods and techniques created not only for designers but also for developers. Modeled to be compatible with DevOps and agile practices, DesignOps structures and organizes design work to enable early and frequent feedback via collaboration between the user, the designer and the developer as well as ongoing, iterative delivery of assets and design decisions to the development team. This allows product teams to run parallel tracks of work (dual-track agile) in which UX teams employ “continuous discovery” to understand the user, engage in research, explore various design directions, test possible solutions and document outcomes while also progressively supporting early development activities such as tech design and story creation.

User Advice: Application leaders should educate themselves about the practice of DesignOps, train their UX teams in the basics of agile and pilot the approach with a high-performing, multi-disciplinary feature team. Following a successful pilot, application leaders and the pilot team members should engage in a product-wide rollout that involves training, updated product plans and the allocation of one or more people to the role of design manager — essentially, a UX-focused product manager. It should be noted that a successful rollout of DesignOps at the product level requires complete buy-in from product management, design and development teams as well as robust logistical and administrative skills.

Business Impact:

- When coupled with DevOps, DesignOps leads to more innovative solutions. As a practice, DesignOps employees dual-track agile which sets aside ongoing tracks of work dedicated to new discovery, idea generation and design exploration. This work acts as a constant source of evidence-based, multidisciplinary innovation.
- DesignOps reduces the time to market for major updates and incremental feature enhancements alike. Due to the concepts of continuous discovery and continuous delivery, developers engage in tech design, architectural explorations and proofs of concept sooner than before, and with much deeper understanding of the overall vision.
- DesignOps increases communication and camaraderie between design and development teams. The design-development gap exists for many reasons, one of them being culture. DesignOps promotes multidisciplinary teams in workshop settings, design sprints or one-on-one “pairing and sharing” that promotes understanding, empathy and relationship-building between these two critically important groups.

Benefit Rating: High

Market Penetration: 1% to 5% of target audience

Maturity: Emerging

Sample Vendors: Adobe; Figma; InVision

Recommended Reading: “DesignOps: Organize, Collaborate and Innovate Product UX at Speed”

“Build Links Between Customer Experience, Multiexperience, User Experience and Employee Experience”

“2019 Strategic Roadmap for Becoming a Digital Product Delivery Organization”

“Application Architecture, Development, Integration and Platforms Primer for 2020”

Promise Theory

Analysis By: Roger Williams

Definition: Promise Theory is an approach to modeling the interactions between entities to understand uncertainty, improve coordination and solve problems. Agents (entities in a system that can independently change) make explicit, voluntary promises to others about unverified behaviors. This enables an agent to examine a system and assess the likelihood of desired outcomes.

Position and Adoption Speed Justification: Promise Theory was proposed by Mark Burgess (creator of CFEngine) in the mid-2000s to avoid the pitfalls of using command and control instructions to model interactions between IT components. Interest in the topic was resurrected in the mid-2010s with the rise of DevOps. While writing on Promise Theory has far exceeded industry adoption, the recent inclusion of support for promises within the Apache Software Foundation's Brooklyn framework for application management is a sign of growing influence.

Promise Theory can provide a common language for the various aspects of configuration management that exist within organizations due to the bottom-up nature of its approach. However, Gartner has not encountered organizations using it for this purpose as of mid-2020. While the explicit use of Promise Theory for performance management has not yet been observed within organizations, there are many parallels with techniques used in Site Reliability Engineering that echo the use of promises. Gartner expects that the clarity and consistency of such an approach will lead to its adoption within many organizations.

User Advice: Product and DevOps teams seeking to use Promise Theory for improvement should begin by identifying the agents that relate to the situation at hand. This includes the devices and the staff that enable the system to deliver the desired result. For each agent, determine the promises (either explicitly or inherent) that contribute to the result and identify each of these attributes:

- The intention and the agent that is undertaking it voluntarily.
- How that intention is being communicated to another agent(s).
- An assessment of the level of commitment and intensity.
- What benefit of value other agent(s) can expect to obtain when the promise is kept.

As an example, a storage device implicitly makes a promise to provide certain types of data upon receiving an authorized request for enable a decision to be made. Likewise, a developer may explicitly promise to complete a code review by the end of the next business day to ensure that security requirements have been met.

Once an initial set of promises are identified for an improvement, stakeholders should review the network of promises for gaps and conflicts. Agents can then identify what new and changed promises they are willing to make to fill gaps, reduce friction or eliminate conflicts.

For configuration management, use the attributes of a promise to provide a common template for representing desired configuration regardless of domain (security, CMDB, version control, etc.), which Gartner refers to as Promised Baseline Configuration Management (PBCM). PBCM makes it easier for product and DevOps teams and cross-functional stakeholders to share their concerns and ensure that all business and technical requirements are accounted for in planned actions.

Business Impact: Promise Theory's open structure, use of simple concepts and acknowledgment of uncertainty inherent in promises is applicable to building trust in digital business. Given the importance of building trust to succeed with Agile and DevOps, Promise Theory's use to improve coordination between subject matter experts in product teams and for product teams' performance management can be a catalyst for scaling these efforts. The result is a set of decentralized and distributed systems that are nimble and resilient.

Promise Theory is most suitable for more autonomous domains where the amount of obligations is lower and there is greater latitude for choosing work activities and commitments. IT leaders responsible for digital products and charged with designing digital business ecosystems can benefit from using Promise Theory to model the dependencies and interactions of agents outside of the control of the organization that are crucial for sustained success.

Benefit Rating: High

Market Penetration: Less than 1% of target audience

Maturity: Embryonic

Sample Vendors: Apache Software Foundation; Cisco Systems; Northern.tech (CFEngine)

Recommended Reading: "Innovation Insight for Promise Theory"

"Digital Business Demands a New Leadership Style — The Why, What and How"

"Reset Your Information Governance Approach by Moving From Truth to Trust"

Performance Engineering

Analysis By: Joachim Herschmann

Definition: Performance engineering is a systematic approach for continuous application performance improvement that involves practices, techniques and activities during each DevOps phase to achieve the performance quality goals of the business. It focuses on the architecture, design and implementation choices that will impact application performance and encompasses the practices that help mitigate performance risks before progressing to subsequent phases.

Position and Adoption Speed Justification: DevOps practices include continuous integration and continuous deployment, yet testing practices remain immature and mostly focused on functional testing. The ability to consistently deliver products that satisfy end-user expectations of overall user experience, quality of service (QoS), availability and performance has become crucial for digital businesses. Performance engineering promotes a holistic and proactive approach with performance requirements driving the design, development and delivery of products as part of a continuous quality strategy.

Although performance engineering continues to gain interest and grow, challenges remain. While its individual elements have been known for many years, performance engineering as a holistic discipline has not yet been widely adopted. Organizational silos, traditional top-down management structures and lack of experience with continuous quality can impede adoption rates. As a result, the speed of adoption will vary between verticals and organizations.

User Advice: Application leaders must create awareness for raised end-user expectations for application quality, specifically nonfunctional characteristics such as performance efficiency. ISO/IEC 25010 — the international standard for software quality — provides a template for understanding quality characteristics and includes performance efficiency as one of the top-level nonfunctional domains. It includes overall user experience, QoS, availability and performance. Application leaders can take this as an example to drive a more holistic view of quality that will help reduce technical debt, mitigate application performance risks and satisfy the quality expectations of the users/business. They should ensure that performance is an overarching requirement for development by engaging the key stakeholders from business, the users and IT right from the start. They should foster a culture that makes performance an explicit requirement by enhancing functional requirements with performance-related criteria, and stating precise and measurable performance targets that DevOps teams can write tests against.

By supporting their teams to adopt performance engineering practices, application leaders will help improve application performance and the behavior of production systems. This will also aid better understanding of end-user experience, which is crucial for developing applications that are fit for purpose. Together, performance engineering practices constitute a closed-loop system, providing continuous information about critical performance indicators as part of a DevOps toolchain. Operations teams, particularly cloud operations, can define and provide performance metrics based on business requirements, not just technology KPIs.

Business Impact: Performance is one of the most critical nonfunctional characteristics of applications. Functional defects can typically be fixed relatively quickly and with comparatively little effort. Design flaws that impact performance are much harder to fix because they affect a larger number of features in different parts of the application. Such architectural or structural defects are typically some of the main contributors to technical debt.

Performance engineering includes both “shift left” and “shift right” testing practices and will provide additional value through tighter integration in DevOps toolchains. The adoption of a performance engineering strategy significantly improves an organization’s ability to consistently deliver solutions that delight customers by exceeding their performance expectations. It provides the framework for application performance excellence that drives value and supports the realization of business outcomes for customers.

Benefit Rating: Moderate

Market Penetration: 1% to 5% of target audience

Maturity: Emerging

Sample Vendors: AppDynamics; BlazeMeter; Dynatrace; Eggplant; IBM; Micro Focus; Neotys; New Relic; Parasoft; SmartBear

Recommended Reading: “Adopt a Performance Engineering Approach for DevOps”

“Innovation Insight for Continuous Quality”

“Take a ‘Shift Left’ Approach to Testing to Accelerate and Improve Application Development”

“DevOps Success Requires Shift-Right Testing in Production”

“Maverick* Research: Software Testing and the Illusion of Exterminating Bugs”

Chaos Engineering

Analysis By: Jim Scheibmeir; Dennis Smith

Definition: Chaos engineering is the use of experimental and potentially destructive failure or fault injection testing to uncover vulnerabilities and weaknesses within a complex system. It is systematically planned, documented, executed and analyzed as an attack plan to test components and whole systems both pre- and postimplementation. CE is often utilized by site reliability engineering teams to proactively prove resilience during fault conditions, and to eliminate those potential sources of system downtime.

Position and Adoption Speed Justification: Chaos engineering has emerged from the practices first pioneered by Netflix (such as Chaos Monkey, Chaos Kong and the Simian Army) and Google (via their DiRT exercises). Early efforts at chaos engineering took simple actions (such as unexpectedly killing a virtual machine). The practice is moving beyond innovative early adopters and being utilized in leading enterprises in the financial services and online retail industry. While there continues to be substantial interest in the wider IT community, chaos engineering will eventually find its way to more enterprises over the next few years as many mature their digital initiatives.

As companies attempt to build scalable, highly available systems, they must demonstrate resilience in the face of not only worst-case scenarios (such as unexpected load and component outages), but also “corner case” and cascade events that start from minor issues. The practice is moving beyond innovative early adopters and becoming integrated into leading enterprises in the financial services and online retail industry. There continues to be substantial interest in the wider IT community, and it will find its way to more enterprises over the next few years as many commence their digital initiatives.

User Advice: We recommend the following:

- Utilize a test-first approach by practicing chaos engineering in preproduction environments and move findings into production environments.
- Incorporate as a part of your system development and/or testing process.
- Build-out incident response protocols and procedures, as well as monitoring, alerting, and observability capabilities in part with advancement of any chaos engineering practice.
- Utilize scenario-based tests, known as “Game Days,” to evaluate and learn about how individual IT systems would respond to certain types of outages or events.
- As your practices mature, investigate opportunities to use chaos engineering in production to facilitate learning and improvement at scale, although we believe that there are still very few organizations purposely using it in their production environments.

- While not a security technique, chaos engineering does sometimes discover security issues.
- By adopting and then extending chaos engineering with autoremediation and continuous validation in the live operational environment, a “digital immune system” may be developed within an application or system.

Business Impact: With chaos engineering, we minimize time to recovery and change failure rate, in addition to uptime and availability — all of which help improve customer experience, customer satisfaction, customer retention and new customer acquisition.

Benefit Rating: High

Market Penetration: 1% to 5% of target audience

Maturity: Adolescent

Sample Vendors: Alibaba Cloud; Bloomberg; ChaosIQ; Gremlin; Microsoft; Netflix; OpenEBS; Verica; VMware

Recommended Reading: “Innovation Insight for Chaos Engineering”

“How to Safely Begin Chaos Engineering to Improve Reliability”

“DevOps Teams Must Use Site Reliability Engineering to Maximize Customer Value”

“Not Just Microservices: Choose the Right Service Granularity for Your Applications”

“Market Guide for Service Orchestration and Automation Platforms”

“Maverick* Research: Software Testing and the Illusion of Exterminating Bugs”

DevOps Value Stream Delivery Platforms

Analysis By: Manjunath Bhat

Definition: DevOps value stream delivery platforms provide a fully integrated set of capabilities for planning, version control, build automation, continuous integration, test automation, continuous deployment (and rollback), release orchestration and automated security policy enforcement to enable continuous delivery of software. However, they may not offer best-of-breed tools across the entire DevOps pipeline. DevOps VSDPs either include native support or expose APIs to optimize the software delivery value stream.

Position and Adoption Speed Justification: DevOps value stream delivery platforms (VSDPs) are actively used in production by organizations worldwide and continue to gain traction. VSDP providers continue to mature their solutions through building/integrating needed functionality or through acquisition. DevOps VSDPs are used by both product and platform teams. Although organizations expect VSDPs to cover all phases of the software development life cycle, VSDP providers aim to provide key capabilities that differentiate their platform while leaving gaps to be filled either by open-source tools or third-party integrations. As a case in point, some VSDP providers include a source code repository while others include a binary artifact repository.

DevOps VSDP providers specialize in a few phases of the application development life cycle although they support the complete application delivery value stream. Where there are gaps, they aim to gradually build or acquire other solutions. This poses a challenge for organizations that aim to replace an assorted array of toolchains with a single platform.

Gartner believes that the growing pace of adoption of these platforms will result in rapid innovation thus enhancing their value proposition to buyers in the next two years. For example, there was a three-fold increase in the amount of inquiries around a single cloud-based DevOps VSDP platform between April 2019 and April 2020 compared to the year before.

User Advice: Gartner recommends DevOps VSDPs in the following scenarios:

- When the benefit of an integrated toolchain outweighs the cost of managing a complex toolchain. Operating a DevOps toolchain is not a one-time exercise and requires resources and skills for patching, scripting, integration and a deep understanding of development workflows and deployment architecture.
- When specialized VSDP capabilities in one area of the application delivery value stream (such as tight-integration between security policy enforcement, code reviews, continuous integration and version control) are a must-have. Make sure that the “good-enough” capabilities outside of the specialization meet enough of your needs to justify the trade-off. For instance, capabilities for canary and blue-green deployment, manual change approvals, automated rollbacks may not be native to all VSDPs.
- When licensing and pricing advantages of DevOps VSDPs make it more attractive than stitching together a complex DevOps toolchain through multiple vendor relationships.

Gartner cautions against DevOps VSDPs in the following scenarios:

- When the DevOps VSDP choice dictates the application architecture. For example, some VSDPs only support serverless architectures. This may be exactly what you need in some cases, but it is not suited for all types of workloads.
- When VSDPs do not integrate with other tools to support missing gaps in their functionality. For example, VSDPs should either provide end-to-end visibility into agile and DevOps metrics to optimize flow of work or support integration with other DevOps value stream management platforms (VSMPs). When SaaS-based DevOps VSDPs do not integrate with on-premises source code repositories, you must either be willing to migrate source code to cloud repositories or look for alternatives.
- When product teams are not ready to replace an existing toolchain but platform engineering teams try to standardize the toolchain across all product teams. The DevOps VSDP choice must enable and not inhibit developer productivity and business agility. In addition, DevOps VSDPs may create affinity to one vendor. Therefore, we recommend that you validate the vendor’s product roadmap and strategy to ensure it meets your technology and architectural requirements.

Business Impact: DevOps VSDPs become the primary engine for digital transformation as organizations map their value streams and aim to optimize both business and IT value streams. DevOps VSDPs enable continuous delivery of applications through a seamless integration between an organization's development and deployment workflows. Using a common platform for development, security and operations bridges gaps and fosters collaboration between organizational silos. DevOps VSDP allows all teams participating in the DevOps value stream to have shared visibility into business, development and operational metrics thus allowing them to be measured using shared business outcomes.

Gartner's definition of DevOps VSDP does not describe a system that cobbles together disparate DevOps tools. Instead, we view VSDPs as integrated platforms that enhance productivity, collaboration, communication and business outcomes for the product team through the interactions among the platform components. A platform approach to toolchains should result in a significant, rather than an incremental flow of value.

Gartner sees the following drivers for the rapid adoption of DevOps VSDPs in the next two to three years:

- Cloud adoption
- Container-native architectures
- Increased use of agile development methodology
- Digital transformation initiatives
- Competitive pressure and existential threat that demand an accelerated return on agile initiatives

Benefit Rating: High

Market Penetration: 5% to 20% of target audience

Maturity: Adolescent

Sample Vendors: Amazon Web Services; Atlassian; CloudBees; GitHub; GitLab; Google Cloud Platform; JFrog Platform; Microsoft Azure

Recommended Reading: "The Future of DevOps Toolchains Will Involve Maximizing Flow in IT Value Streams"

"Agile and DevOps Primer for 2020"

"Flattening the Application Organization — Everyone Must Be Part of the Agile Value Stream"

DevOps Value Stream Management Platforms

Analysis By: Hassan Ennaciri

Definition: DevOps value stream management platforms (VSMPs) enable organizations to manage their software delivery process as a value stream to maximize customer value. They provide visibility and traceability to every process in software delivery — from ideation through development to release and production, and extending to documenting feedback from customers. DevOps VSMPs continuously measure and surface constraints and non-value-added work, enabling organizations to develop a strategic approach focused on customer value delivery.

Position and Adoption Speed Justification: To scale agile and DevOps practices, organizations need a DevOps VSMP that provides visibility into higher-level metrics. DevOps VSMPs integrate with multiple data sources to provide DevOps-related telemetry. Visible metrics enable product owners, application leaders and I&O leaders to make data-driven decisions in an agile manner and to correct course quickly. This is accomplished because the VSMP is able to integrate with the plethora of tools in use and provide curated, higher-level metrics.

DevOps VSMPs serve as an orchestrator by integrating with existing DevOps tools. They provide product teams with flexibility and freedom in selecting their tools by simplifying integration with existing DevOps toolchains or value stream delivery platforms (VSDPs). DevOps VSMPs also provide governance capabilities, including security enforcement and compliance controls. Product teams can extend those capabilities by integrating other best-of-breed security and compliance tools into the platform.

DevOps VSMPs do not include native continuous integration/continuous delivery (CI/CD) capabilities; execution of the delivery pipeline is a function of VSDPs or other best-of-breed or existing tools. However, by integrating with those tools, DevOps VSMPs provide access to detailed data that enables creation of real-time analytics and robust metrics.

Gartner clients can expect existing enterprise agile planning tools and DevOps VSDPs to expand their value stream management capabilities. VSMP vendors will continue to expand their value and capabilities with advanced analytics and additional integrations to more tools as they are adopted (see “Innovation Insight for AI-augmented Development”).

User Advice: Gartner recommends using DevOps VSMPs to:

- Scale DevOps initiatives across the organization.
- Provide greater visibility into team performance, business value creation or slowdowns in delivery. DevOps toolchains often produce fragmented key performance indicators (KPIs) and metrics, as each tool generate its own siloed metrics. DevOps VSMP aim to address the challenges that arise from disjointed dashboards and toolsets by providing a unified view across the DevOps toolchain.
- Analyze leading indicators that affect the lagging indicators of business value. Examples of leading indicators include the difference between planned versus unplanned work, planned versus actual work, average cycle time and amount of work in progress.

Gartner cautions against DevOps VSMPs in the following scenarios:

- When the VSMP does not integrate with deployed client tools.

- When client are just starting to build a CI/CD toolchain.

Business Impact: DevOps VSMPs provide organizations with the capabilities to scale DevOps adoption and enable successful digital transformation. They bridge the gap between business and IT, and enable stakeholders to align their priorities to focus on delivering customer value. DevOps VSMPs can provide CXOs with strategic views of product delivery health and pipelines, allowing them to expedite data-driven decisions about future investments in products.

The visualization capabilities of DevOps VSMPs can help product teams analyze customer value metrics against the cost required to deliver that value. Customer value metrics include lead time, deployment frequency, defect escape rate, feature adoption, time to respond to failures and resource cost.

VSMPs allow all teams participating in the DevOps value stream to have shared visibility into business, development and operational metrics. This allows them to measure shared business outcomes between teams, identify constraints and foster collaboration between teams.

DevOps VSMPs map end-to-end processes involved in software delivery and provide shared visibility and metrics, allowing teams to reduce waste and optimize flow. Teams can improve their value as well as reduce bottlenecks due to cross-team dependencies.

We believe the following trends will drive rapid adoption of DevOps VSMPs in the next two to three years:

- Scaling agile and DevOps adoptions
- Digital transformation initiatives
- Migrations of workload to the cloud
- Migration of legacy applications to cloud-native architecture, such as microservices (resulting in increased deployments and a greater need to measure the quality of deployments)

Benefit Rating: High

Market Penetration: 5% to 20% of target audience

Maturity: Adolescent

Sample Vendors: Digital.ai; Eficode; IBM; Microland; Plutora; ServiceNow; Tasktop; TCS; Tech Mahindra

Recommended Reading: “The Future of DevOps Toolchains Will Involve Maximizing Flow in IT Value Streams”

“Agile and DevOps Primer for 2020”

“Flattening the Application Organization — Everyone Must Be Part of the Agile Value Stream”

Continuous Quality

Analysis By: Joachim Herschmann; Jim Scheibmeir

Definition: Continuous quality is a systematic approach toward process improvement to achieve the quality goals of business and development. A continuous quality strategy fosters a companywide cultural change to achieve the goal of making “quality” the responsibility of all. It synchronizes quality assurance and testing with DevOps processes and encompasses the practices that help mitigate risks before progressing to subsequent stages of the software development life cycle.

Position and Adoption Speed Justification: Many organizations are on a journey with DevOps, practicing continuous integration and continuous deployment, yet a continuous approach to quality is often missing. The ability to consistently deliver business value with high quality has become critical for organizations seeking to mature their DevOps processes. Continuous quality encourages a holistic and proactive approach with functional and nonfunctional requirements driving the design, development and delivery of products.

Continuous quality involves engaging stakeholders across the organization and empowering them to be more accountable and to seek out opportunities for improvement. Such a holistic approach can be seen as restrictive and requires consensus for usage across all team members. Organizational silos, traditional top-down management structures and lack of experience with continuous quality can impede adoption rates. These challenges will be overcome only when organizational maturity and the necessary culture change expand effectively across the broader organization, and as DevOps practices and projects mature.

User Advice: Application and I&O leaders should:

- Move from a traditional application or project-centric model of quality to a holistic quality approach by adopting an ecosystem-centric view of quality.
- Promote a continuous quality mindset by involving stakeholders across the organization.
- Allocate ownership and appoint staff with the required skills needed for continuous quality by identifying the required roles, technologies and practices.
- Enable collaboration with UX design and CX teams to infuse quality right from the inception of an idea.
- Establish relevant quality metrics based on the objectives that your DevOps teams are trying to accomplish. Draw on ideas from Gartner’s DevOps metrics pyramid.
- Task your teams with developing continuous quality practices before choosing tools.

Business Impact: The adoption of a continuous quality strategy significantly improves an organization’s ability to serve and delight its customers. Continuous quality enables solutions to be delivered at a greater release rate and with fewer defects than traditional quality control practices. It provides the framework for operational excellence that drives value, supports the realization of business outcomes for customers and streamlines operational processes. Operating this way challenges traditional quality control practices, and is instrumental in breaking down silos, rewarding collaboration and creating continual improvement.

Continuous quality supports both “shift left” and “shift right” testing. Testing can occur before software development begins through requirements vetting, and can exist in future states, such as A/B testing or using canary releases in production. Managing quality proactively and continuously will increase the speed at which digital solutions are delivered by creating a timely and accurate feedback loop about what is working and what is not working.

Benefit Rating: High

Market Penetration: 1% to 5% of target audience

Maturity: Emerging

Sample Vendors: AppDynamics; BlazeMeter; Eggplant; IBM; Micro Focus; Neotys; New Relic; SmartBear

Recommended Reading: “Innovation Insight for Continuous Quality”

“Promote Continuous Quality for APIs to Support Digital Business”

“Quality Is the Key to Avoiding ‘Digital Distortion’ With Your Augmented Reality Strategy”

“Take a ‘Shift Left’ Approach to Testing to Accelerate and Improve Application Development”

“Adopt a Performance Engineering Approach for DevOps”

“Avoid Digital Quality Disasters With Continuous Autonomous Validation and Verification”

“Maverick* Research: Software Testing and the Illusion of Exterminating Bugs”

Platform Ops

Analysis By: Daniel Betts; George Spafford

Definition: Platform Ops is an approach to scaling DevOps that involves dedicating a team (platform team) to the operation and development of a codified, highly automated, shared self-service platform. This enables multiple agile product teams to accelerate delivery while managing quality, security and compliance, and standardization.

Position and Adoption Speed Justification: A growing number of Gartner clients are utilizing Platform Ops to scale their DevOps initiatives but there is confusion over how to start and grow Platform Ops efforts.

Platform Ops practices are adopted by organizations needing to scale DevOps initiatives, often to reduce overlap and redundancy, enable economies of scale, and establish high standards of governance. These practices require a culture that supports learning and improvement, highly skilled automation practices, and usage of infrastructure as code (IaC) capabilities.

Platform Ops uses automation to reduce manual processes, leverages resilient system and software engineering principles, and agile development processes. Platform teams build the infrastructure as a product to be consumed by each product team. Enhancements to the platform reflect the evolving needs of developers as informed by collaborative organizational learning practices. Platform Ops approach to tasks can be different, but high-level operations' functions are common, such as security, access control, compliance and performance management.

User Advice: Platform Ops teams must be designed to deliver agile infrastructure for multiple product teams to consume. Build teams, processes and culture to continually improve — not just sustain — the platform. The complexity of maintaining and continuously improving multitenant, shared platforms requires infrastructure and operations (I&O) leaders to create cross-functional teams of SMEs. Platform Ops teams need longevity, full-time commitment and dedicated funding throughout their life cycle. Platform Ops teams must have the resources needed to deliver viable platforms for the product teams.

I&O professionals and software engineers/application developers are essential to the long-term success and reliability of the shared self-service platforms. I&O professionals have the knowledge and experience of operating mission-critical systems, performance and security. Software engineers bring software engineering principles and help drive IaC adoption.

Treat your platform as a product. Your users are your customers, and their requirements should define the platform and your priorities. You are a provider, enabling your internal customers (development teams) to deliver services to your organization's customers.

Platform teams must adopt agile methods when designing the architecture and should always deliver customer-centric solutions. Marketing and championing the self-service platforms is critical to their success, if product teams do not perceive value or are unaware of the platforms, they are likely to seek alternatives.

Before embarking on Platform Ops — picking technologies, hiring a team — you should decide that Platform Ops is right for your organization to scale DevOps. If your organization is just starting DevOps or struggling to find DevOps maturity, Platform Ops will not help, efforts should be focused on an iterative learning approach and scale when ready.

Business Impact: Using a product and platform approach can help enable the business to respond faster to threats and opportunities while also managing cost and risk. The product teams can focus on customer requirements and, in turn, the platform teams can specialize and seek to enable the product teams. This makes it well suited to support digital business initiatives such as analytics, mobile applications, portals and so forth.

Benefit Rating: Transformational

Market Penetration: 5% to 20% of target audience

Maturity: Adolescent

Recommended Reading: “How to Scale DevOps by Building Platform Teams”

“How to Fix the Software Engineering Resource Gap in I&O”

“How to Lead Digital Disruption With Programmable Infrastructure”

“How to Start Executing a Successful Automation Strategy”

“Foster Communities of Practice to Ensure Successful DevOps”

Continuous Compliance Automation

Analysis By: Hassan Ennaciri

Definition: Continuous compliance automation (CCA) integrates compliance and security policy enforcement into a DevOps delivery pipeline. CCA codifies and continuously applies compliance policies and controls while monitoring, correcting and protecting against vulnerabilities resulting from coding defects and misconfiguration. It reduces manual execution steps in adhering to regulatory requirements, enhancing consistency, traceability and audibility. CCA helps reduce costly delays caused by fixing compliance problems that often occur late in the SDLC.

Position and Adoption Speed Justification: Growing DevOps initiative success continues to drive enterprise investments in compliance automation. CCA improves release velocity and reliability while simplifying compliance enforcement via automation. Policy-driven, automated controls improve compliance without impacting the flow of the software delivery value stream.

Traditional compliance practices are incompatible with continuous software delivery processes, leading to slower delivery and unexpected remediation work. CCA is a necessary approach that DevOps teams need to implement to realize the benefits of continuous delivery of value.

Compliance automation tools for DevOps are changing and evolving rapidly as there many vendors that cover different subdomains across applications, infrastructure and databases.

User Advice: DevOps teams must adhere to compliance, governance and security requirements while creating a leaner operating environment. CCA tools enable them to achieve both goals: improving value stream delivery and mitigating risks.
DevOps teams must consider the following actions when integrating compliance testing into DevOps delivery pipeline:

- A shift-left approach to ensure that compliance controls are well understood earlier in the development process. This includes implementing automated compliance checks at every phase of the pipeline.
- Investments in tools that enable CCA at scale and can provide a continuous approach to prevent, detect and correct audit failures.
- Enforcement of security and compliance across all domains, including databases, application code, infrastructure and open-source software (OSS). Since there is no single vendor tool that

covers all those domains, DevOps teams must use multiple compliance automation tools and integrate them across all phases of the delivery pipeline.

Business Impact: Organizations evolving their DevOps practices and avoid risks and penalties by embedding automated compliance into their delivery pipelines.

Platform and product engineering teams must use compliance automation tools to meet the requirements of regulatory frameworks such as the SOC 2 System and Organization Controls, the National Institute for Standards and Technology [NIST] Special Publication (SP) 800-53, NIST CSF, Payment Card Industry (PCI), the International Organization for Standardization (IOS)/International Electrotechnical Commission (IEC) 27001, the Health Insurance Portability and Accountability Act (HIPAA), or the Sarbanes-Oxley Act (SOX).

As organizations face an increasing number of regulatory obligations, automating compliance will become even more valuable to I&O leaders. In our 2019 DevOps Survey, 98% of respondents indicated that they are using DevOps in regulated situations. Further, respondents indicated that they need to comply with an average of three regulations.

Benefit Rating: Moderate

Market Penetration: 5% to 20% of target audience

Maturity: Adolescent

Sample Vendors: Check Point Software Technologies; Checkmarx; Chef; Fugue; Liquibase (formerly Datical); Qualys; Rapid7 (DivvyCloud); Redgate; Sonatype; Veracode

Recommended Reading: “3 Steps to Ensure Compliance and Audit Success With DevOps”

“3 Steps to Integrate Security Into DevOps”

“Market Guide for Compliance Automation Tools in DevOps”

At the Peak

Artificial Intelligence for IT Operations (AIOps) Platforms

Analysis By: Charley Rich; Pankaj Prasad

Definition: Artificial intelligence for IT operations (AIOps) platforms combine big data and machine learning through support of all primary IT operations functions through the scalable ingestion and analysis of the ever-increasing volume, variety and velocity of data generated by IT operations. The platform enables the concurrent use of multiple data sources, data collection methods, and analytical and presentation technologies.

Position and Adoption Speed Justification: Increased demand for AIOps platform capabilities is fueled by the growing need to intelligently drive the acceleration and automation of IT operations functions through analysis of both historical and real-time data. This is happening as roles and responsibilities converge (with DevOps and SRE as a leading examples) in the pursuit of greater agility as well as the ever increasing momentum behind digital transformation. The desire to intelligently drive automation requires continuous insights derived from machine learning algorithms based on data generated by ITOM disciplines like APM, ITIM, NPMD, DEM and ITSM. AIOps platform adoption — in particular, machine-generated data including logs, metrics and traces, as well as human-processed data such as incidents dependencies and changes — continues to rise in support of ensuring high-quality digital experience.

Interest and investment will continue to rise due to:

- Rapid growth in data volumes generated by the IT systems, networks and applications
- Increasing data variety — velocity at which data is generated and changing
- Challenges in maintaining observability and improving engagement due to the adoption of cloud-native and ephemeral architectures
- The need to intelligently and adaptively drive the automation of recurring tasks and predict change success and SLA failure

AIOps capabilities have evolved across multiple dimensions:

- The domain-agnostic AIOps platforms with vendors offering a general-purpose AIOps platform.
- Domain-centric AIOps vendors, that have the key components, but with a restricted set of use cases focused on one domain (for example, network, endpoint systems, APM or ITSM).
- Do it yourself (DIY), where end users can mix and match the components, essentially assemble tools for data ingest, a big data platform, ML and a visualization layer from multiple providers or open-source projects.

Machine learning uses multiple analytical approaches, while remediation requires significant maturity. Gartner still sees event correlation as the predominant practical use case, while aspirational goals like real-time business insights requires end-users to invest in resources like time, effort and skills. Remediation is still being handled via rule-based approaches although vendors are beginning to deliver ways to systemize and recall the problem resolution process for later reuse.

User Advice: I&O leaders must build a strategic AIOps platform investment plan that is tied to driving measurable business outcomes though analysis of performance, digital experience and delivery automation while utilizing stagewise implementation of AIOps capabilities:

- Begin with practical goals, such as reducing operational noise through event correlation and anomaly detection, and later moving on to root-cause analysis.
- Start proactively detecting the signals that indicate emerging problems before users are impacted.
- Use NLP to democratize the automation of reoccurring workflows, making it easier to initiate them without deep specialist skills.

- Apply the pattern-matching capabilities of AIOps to the DevOps build-deploy process in order to detect potential impacts to production prior to deployment.

The AIOps strategy must account for the following:

- Balancing ease of implementation/use with interchangeability of platform capabilities
- ITOM tool portfolio rationalization
- Key technology gap investment

Before embarking on an AIOps journey, I&O leaders must determine whether using a domain-centric AIOps solution such as a monitoring tool that leverages machine learning is sufficient or whether a separate AIOps solution is necessary for their use cases. The domain-centric solution will likely have a shorter time to value, but its scope will be narrow and impact will be less. Domain-agnostic solutions may address a broad scope, and while their time to value will necessarily be longer their impact can be greater. If a domain-centric solution is already deployed for its primary purpose, evaluate its capabilities for AIOps in relation to the data sources that must be analyzed before considering a domain-agnostic solution.

Business Impact: By enabling I&O teams to enhance and transform major operational functions with a real, automated insight generation capability, organizations across all verticals stand to realize:

- Agility and productivity gains — via active combined analysis of both IT and business data, yielding new insights on user interaction, business activity and supporting IT system behavior.
- Service improvement and cost reduction — via a significant reduction in time and effort required to identify the root cause of availability and performance issues. Behavior-prediction-informed forecasting can support resource optimization efforts.
- Risk mitigation — via active analysis of monitoring, configuration and service desk data identifying anomalies from both operations and security perspectives.
- Competitive differentiation/disruption — via superior responsiveness to market and end-user demand based on machine-based analysis of shifts, beyond those that are immediately obvious to human interpretation.

Benefit Rating: Transformational

Market Penetration: 5% to 20% of target audience

Maturity: Emerging

Sample Vendors: Aisera; Appnomic; BigPanda; BMC; Digitate; Moogsoft; ScienceLogic; ServiceNow; Splunk; StackState

Recommended Reading: “Market Guide for AIOps Platforms”

“Avoid the Unexpected Consequences of IT Change Management With AIOps and CMDB”

"Assess Approaches to AIOps for a Comprehensive Solution"

"Deliver Cross-Domain Analysis and Visibility With AIOps and Digital Experience Monitoring"

"Augment Decision Making in DevOps Using AI Techniques"

Autonomous Testing

Analysis By: Joachim Herschmann; Jim Scheibmeir

Definition: Autonomous testing comprises AI- and ML-based technologies and practices to make software testing activities independent from human intervention; it continuously improve testing outcomes by learning from the collected data from performed activities. It extends traditional test automation beyond the automated execution of test cases to include fully automated planning, creation, maintenance and analysis of tests.

Position and Adoption Speed Justification: Autonomous testing is in its early stages, based on the foundation of test automation and early AI-augmentation. As DevOps leaders seek to improve release velocity without degrading quality, it becomes imperative to drive higher levels of effective testing. Autonomous testing promotes a holistic approach to automating the broader set of testing activities related to requirements quality, design quality, code quality, release quality and operational resilience in an integrated way, increasing the degree of autonomy for those activities.

Autonomous testing is quickly gaining interest as evidenced in inquiries from Gartner clients, yet several challenges remain. AI and ML technologies in general are very hyped but existing AI-augmented test automation offerings are still focused on very narrow problem areas, such as "smarter" UI object recognition or "intelligent" test result insights. While this adds incremental value, the lack of tool integration and little experience with AI and ML technologies can impede adoption.

User Advice: Application and I&O leaders should:

- Start autonomous testing initiatives by setting simple and clear goals. Examples include risk reduction, strengthening confidence in a release candidate, freeing up testers' time, and delivering working software frequently.
- Set the right expectations about where autonomous testing can provide value, what its current limitations are and what is needed for it to be successful.
- Maximize the impact of autonomous testing by leveraging it as an enabler of a systematic approach to achieve the quality goals of business and development. Focus on key business value enablement and determine where it can help with revenue, cost and risk management.
- Allocate ownership and appoint staff with the required skills needed by identifying the required roles, technologies and practices.
- Focus on the areas with greatest benefit for automation by searching for the greatest constraint in the software development life cycle (SDLC). Autonomous testing can provide value in these areas: automated requirement analysis and effort estimation, automated test-case design,

automated test-case creation and maintenance, automated test-set creation and maintenance, automated test-case selection, automated test-set optimization, automated test data generation, automated provisioning of test data and environment, automated test execution, and automated test reporting and decision making.

- Select tools that best match available skills and development style by assessing application profiles and different teams' needs.

Business Impact: The adoption of autonomous testing has the potential to significantly improve an IT organization's ability to serve and delight its customers. It can be an enabler for adjusting testing scenarios and overall software quality parameters as part of a continuous quality initiative aimed at optimizing end-user experience. It will also help to constitute a closed-loop system that provides continuous feedback about critical quality indicators.

Though many aspects of autonomous testing are still relatively immature, its existing capabilities in the areas of automated test-case design, test-case execution and test result analysis provide a solid foundation to drive value, support the realization of business outcomes for customers and accelerate operational processes. Organizations that are starting to invest in autonomous testing now will be well-positioned to take advantage of new capabilities in the future and further improve the level of confidence they have in the quality of their releases.

Benefit Rating: High

Market Penetration: 1% to 5% of target audience

Maturity: Emerging

Sample Vendors: Applitools; Autify; Functionize; mabl; retest; ReportPortal; test.ai; TestCraft; Testim

Recommended Reading: "Magic Quadrant for Software Test Automation"

"Critical Capabilities for Software Test Automation"

Dojos

Analysis By: Katherine Lord

Definition: To drive and scale cultural change, I&O and application leaders must accelerate organizational learning by creating a dojo. A dojo is a physical gathering space where people learn together in an experiential manner, students learn by doing and people with a variety of skills are present. New people can expand their skill set by learning about methods and tools, while more experts can share new lessons learned and refine skills. Additionally, dojos are useful beyond DevOps and be leveraged for other areas such as agile and lean.

Position and Adoption Speed Justification: Continuous learning and growth are essential for DevOps progress and scalability. Without effective approaches to embed and support a continual, iterative approach to mindsets, skills and culture many DevOps initiatives will fail. In fact, in the 2019 Gartner DevOps Survey, 30% of respondents stated that developing the required skills for delivery staff is one of the top three challenges encountered when adopting DevOps. The survey also found that 33% of respondents stated that organizational learning is one of their top three challenges when scaling DevOps efforts.

A dojo is an industry-proven learning approach to facilitate the iterative, experiential approach required for success.

User Advice: We recommend the following:

- Shift from relying on traditional learning practices that are only completed once by each student and emphasize one-way learning, such as online or classroom courses.
- Avoid modalities that focus on disseminating theory and best practices, which are ineffective at embedding agile and DevOps practices and mindsets.
- Integrate a dojo into their DevOps strategy by starting small and then developing a plan to improve and scale the dojo over time.
- Co-create a dojo charter with all stakeholders by establishing a shared vision for the dojo and using metrics that track progress and success.

Business Impact: Dojos are experiential learning entities that can support shifts in culture, enable new ways of working and help adjust mindsets and behaviors. These are all critical elements for success when it comes to digital transformation, and a foundational shift in focus from building systems, components and code to delivering business value to internal customers.

Benefit Rating: High

Market Penetration: 1% to 5% of target audience

Maturity: Adolescent

Recommended Reading: “4 Steps to Create a DevOps Dojo That Accelerates Learning and Cultural Change”

“How to Scale DevOps by Building Platform Teams”

“Embrace an I&O Operating Model to Support Digital Transformation”

“Hack Your I&O Culture to Accelerate Digital Transformation”

“Foster Communities of Practice to Ensure Successful DevOps”

Programmable Infrastructure

Analysis By: Nathan Hill; Philip Dawson; Milind Govekar

Definition: Programmable infrastructure is the concept of using and applying methods and tooling from the software development area to management of IT infrastructure. This includes, but is not limited to APIs, immutability, resilient architectures and agile techniques. It is also referred to as "infrastructure as code."

Position and Adoption Speed Justification: Programmable infrastructure comprises a composable set of programmable building blocks. Programmable infrastructure goes beyond "aaS" (as a Service) offerings that expose programmable interfaces and enable new ways for delivering infrastructure services. Programmable infrastructure strategies can be applied to private cloud, hybrid cloud and infrastructure platforms, as well as public cloud. Its goal is managing the life cycle of infrastructure delivery from provisioning, resizing and reallocation to reclamation, or in the case of elastic external resources, the termination of consumption.

APIs provide programmatic access to I&O services and data (e.g., depending on the workload requirements, an API that fires off automation that sets up a compute environment with CPU, memory and storage; installs software; assigns IP addresses). These are implemented so that I&O consumers (such as developers) can consume services and data to create new business solutions. Thus, I&O staff should be trained in using web technologies (such as HTTP and JSON) to develop these APIs. I&O leaders also should manage APIs as a technology product and implement full life cycle management, including version control and roadmaps.

The maturity of APIs that enable integration across different infrastructure platforms, combined with the scarcity of programmatic skills within I&O, account for the current maturity of programmable infrastructure.

User Advice: Organizations cannot simply apply automation to existing monolithic infrastructure components. Doing so will result in frustration due to the awareness of agility and response demands without fundamental infrastructure components to deliver on requirements — in essence, automation without platform agility.

Infrastructure and operations leaders must:

- Prioritize agility as one of their top goals in pursuit of digital business outcomes.
- Implement a programmable infrastructure by investing in infrastructure automation tools and AIOps (example vendors for these markets are listed below, but no single vendor or platform can enable an organizationwide programmable infrastructure strategy).
- Invest in infrastructure and DevOps, and modernize legacy IT architectures to implement an API-driven infrastructure.
- Look for reusable programmable building blocks as they extend their programmable infrastructure strategy.

Moving to an API-driven infrastructure is the key first step to enabling anti-fragile and sustainable automation through programmatic techniques. Achieving platform agility is not just about refreshing data center infrastructure to modern platforms like HCIs, although this may form part of the strategy. I&O leaders should consider all areas of the platform — cloud-native architectures, public and private cloud, new infrastructure for new products and services, as well as the modernization of legacy infrastructure.

Business Impact: A continuous-delivery approach requires continuous insight and the ability to automate application responses. This ensures that (only) the right infrastructure resources are available at the right time and location, and this is achieved through a programmable infrastructure. Thus, programmable infrastructure ensures optimal resource utilization while driving cost-efficiencies. However, greater value (than cost reduction) can be achieved via programmable infrastructure's ability to drive adaptive automation — responding faster to new business infrastructure demands, driving service quality and freeing staff from manual operations. It helps reduce technical debt, and enables a sustainable and highly responsive IT infrastructure service to the business.

Benefit Rating: High

Market Penetration: 1% to 5% of target audience

Maturity: Embryonic

Sample Vendors: Alibaba Cloud; Amazon Web Services (AWS); Google; IBM; Microsoft; Pivotal; Tencent Cloud; VMware

Recommended Reading: "Digital Platforms Need Programmable Infrastructure"

Hypothesis-Driven Development

Analysis By: Jim Scheibmeir

Definition: Hypothesis-driven development is software development based on running a series of tests in production to find the best solution. The core principle is that every release is a test of a hypothesis, and is constructed to be the simplest way to prove or disprove that hypothesis.

Position and Adoption Speed Justification: Popularized by the book "The Lean Startup" by Eric Reis in 2011, this methodology has been widely adopted in software product and game development. It is less common in IT development projects, and its position reflects IT adoption. In Gartner's 2019 Agile in the Enterprise Survey, 3% of respondents identified as using hypothesis-driven development (HDD) in 2019 or before with an additional 4% stating they would begin using the practice prior to the end of 2020.

The biggest obstacle to adoption is related to the experimental nature of HDD initiatives. Many IT organizations refuse to start a project where both the solution and the user base are not well-understood. HDD will become more important as companies become comfortable with formal methods of experimentation.

User Advice: Recognize that HDD allows the business to create software solutions in extreme agile problem spaces where little is known about the best solution and the users who will use it. Even in less-innovative initiatives, treat releases as experiments and validate that the new functionality achieves the desired outcome.

Adopt HDD to guide development in customer-facing projects and internal projects where there is a large and diverse user base. Because HDD requires the quick creation of new functionality to run tests and validate hypotheses, it requires agile and DevOps proficiency.

Adopt a hypothesis pattern for stories:

> Instead of the traditional story format of:
>
> “As an <identify user>
>
> I need to do <task>
>
> in order to accomplish <outcome>“

Or the BDD/Gherkin format of:

> “given <initial state>
>
> when <event>
>
> then <result>“

Adopt an HDD pattern such as:

> “We believe that <this capability> will result in <this outcome>
>
> We will know we have succeeded when <we see a measurable outcome>“

Business Impact: When HDD is combined with agile and DevOps practices, it gives the business an opportunity to accelerate the rate of innovation while reducing waste and managing risk. In addition, HDD allows business to create software solutions in extreme Mode 2 problem spaces where little is known about the best solution and the users who will use it. It can prevent the wide-scale deployment of software that is not fit for purpose. It can prevent developing a solution that has no customers. Either way, it allows the team to test the market to see if the offering and its features are valued.

HDD is not just trying things in an ad hoc way. It is a disciplined approach to innovation that requires that functional decisions be based on verified results, not the opinions of forceful or highly ranked individuals. The greatest benefits are to be seen in organizations where hypotheses can come from many different people and can be quickly tested. Organizations using HDD need to also understand that proving a hypothesis false is not a failure.

Benefit Rating: Moderate

Market Penetration: 5% to 20% of target audience

Maturity: Adolescent

Sample Vendors: CloudBees; GitLab; LaunchDarkly; Optimizely; Split

Recommended Reading: “Extend Agile With DevOps for Continuous Delivery”

"Agile and DevOps Primer for 2020"

"Guidance Framework for Adopting Lean UX"

"DevOps Success Requires Shift-Right Testing in Production"

Communities of Practice

Analysis By: Thomas Murphy

Definition: A community of practice (CoP) is a community- or stakeholder-owned structure often initiated by organizations' leaders to enable problem solving, empowerment and organizational learning through collaboration. Agile and DevOps teams require a shift in culture to support higher degrees of collaboration and cross-functional learning, and a breaking down of traditional silos and their associated centers of excellence (COEs). CoPs support knowledge management and professional development by enabling organizational learning and knowledge sharing.

Position and Adoption Speed Justification: CoPs have been widely adopted by agile and DevOps leaders, and the concepts they embody are well-understood. However, the adoption by mainstream clients remains limited, and we still see a tendency to rely on top-down and functionally aligned COEs. COEs are demotivational — they replace problem solving with adherence to a policy. CoPs, by contrast, are motivational if there is a clear vision and space to collaborate, learn and evolve.

The continued shift toward a product model and the adoption of DevOps practices will increase the need to support a CoP culture.

User Advice: The concept of a CoP is not new, but it is now growing fast due to the demand to gain new skills and improve the pace of delivery. The disruption of COVID-19 and its forced movement to remote work will be disruptive to initiating new CoPs but for those that already had these communities are a continued source of strength through interactions and learning. Leaders must continue to create and communicate a vision and empower cross-functional teams to learn and discover through experimentation, collaboration and knowledge sharing.

Digital support is critical both for meeting and collaborating as well as the ability to capture information, share examples, and work on projects. Equally important to having a vision and tools will be making space on the calendar and adopting a culture that makes it acceptable to try something different, as well as to try and fail — the key is to learn. Job rotations, hackathons and technology conferences can also help.

Business Impact: An effective shift to an agile, DevOps and product-based culture requires new skills and models and will be most effective when driven from the community, rather than the top. In addition, teams must break through traditional functional silos to speed up delivery and support the innovation needs of digital business.

Gartner's research finds that CoPs:

- Shorten the learning curve for new employees

- Respond more rapidly to customers' needs and inquiries
- Reduce rework and prevent "reinvention of the wheel"
- Spawn new ideas for products and services
- Develop the capabilities of their members in line with organizational needs

Benefit Rating: High

Market Penetration: 5% to 20% of target audience

Maturity: Adolescent

Sample Vendors: Atlassian; GitHub; Microsoft; Slack; Zoom

Recommended Reading: "How to Build Successful Communities of Practice for Knowledge Management"

"Foster Communities of Practice to Ensure Successful DevOps"

"Ignition Guide to Creating Agile Communities of Practice"

"Toolkit: 2019 Collection of 85 Culture Hacks From the Real World"

"Hack Your I&O Culture to Accelerate Digital Transformation"

Immutable Infrastructure

Analysis By: Steve Riley

Definition: Immutable infrastructure is not a technology capability, rather it is a process pattern in which the system and application infrastructure, once instantiated, is never updated in place. Instead, when changes are required, the infrastructure is simply replaced. Immutable infrastructure could encompass the entire application stack, with in-versioned templates provisioned via APIs, which are most commonly available in cloud IaaS and PaaS.

Position and Adoption Speed Justification: Immutable infrastructure is typically used by organizations that take a DevOps approach to managing cloud IaaS or PaaS; however, it can be used in any environment that supports infrastructure as code. It represents a significant change in process for traditional infrastructure and operations groups. It may manifest as:

- Native cloud capabilities, such as Amazon Web Services (AWS) CloudFormation or Microsoft Azure Resource Manager templates
- Cloud management platforms, such as Flexera
- Software tools, such as HashiCorp's Terraform
- The customer's own automation scripts

Some or all of an application stack will be instantiated in the form of virtual machine images or containers, combined with continuous configuration automation tools that run after initial boot. Containers can be quickly replaced during runtime, while VM replacement is slower and requires greater coordination among other workload components. Containers improve the practicality of implementing immutable infrastructure and will drive greater adoption.

User Advice: Immutable infrastructure ensures that the system and application environment is accurately deployed and remains in a predictable, known-good-configuration state. It simplifies change management, supports faster and safer upgrades, reduces operational errors, improves security, and simplifies troubleshooting. It also enables rapid replication of environments for disaster recovery, geographic redundancy or testing. Cloud-native workloads are more suitable for immutable infrastructure architecture than traditional on-premises workloads. And, because redundancy may be required by CSP terms of service to receive service-level agreement relief, workloads designed with an immutable infrastructure approach lend themselves to easier replication.

The application stack for immutable infrastructure is typically composed of layered components, each of which should be independently versioned and replaceable. The base OS for the master image may be updated using traditional patching tools, or automatically or manually updated. Automation is then used to bundle components into artifacts suitable for atomic deployment, including VM images, container images, storage objects, network connections, and other necessary resources. The scripts, recipes, and other code used for this purpose should be treated similarly to the application source code itself, which mandates good software engineering discipline.

Some organizations that use immutable infrastructure reprovision only when a change is necessary. Others automatically refresh the infrastructure at frequent intervals (known as systematic workload reprovisioning) to eliminate configuration drift, to update components in which vulnerabilities were discovered, or to possibly eliminate advanced persistent threats. Frequent refresh is only practical in environments with fast and reliable provisioning; thus, it benefits strongly from containers. Integrate with a ticketing system so that refreshes can be initiated and tracked to completion.

The use of immutable infrastructure requires strict operational discipline. IT administrators should eliminate the habit of making one-off or ad hoc modifications to avoid configuration drift. Updates must be made to the individual components, versioned in a source-code-control repository, then redeployed so that everything is entirely consistent. No software, including the OS, is ever patched in production. Organizations that use immutable infrastructure may turn off all normal administrative access to instantiated compute resources — for example, not permitting SSH or RDP access. IT leaders should set a hard date for when all new workloads will use immutable infrastructure if technically feasible; deadlines can be effective motivators of behavior change.

None of the vendors listed in this innovation profile sell a product called “immutable infrastructure.” Rather, they offer one or more elements that help to establish an immutable infrastructure style. Expect to purchase multiple tools.

Business Impact: Taking an immutable approach to server and compute instance management simplifies automated problem resolution by reducing the options for corrective action to, essentially, one. This is to destroy and recreate the compute instance from a source image containing updated software or configuration that addresses the problem. Although immutable infrastructure may appear simple, embracing it requires a mature automation framework, up-to-date blueprints and bills of materials, and confidence in your ability to arbitrarily recreate components without negative effects on user experience or loss of state. In other words, getting to that single corrective action is not without effort. Treating infrastructure immutably is an excellent test of the completeness of your automation framework and the confidence of your platform. The immutable approach is a management paradigm, not a technology capability. The long-term outcome is one in which the workload defines the infrastructure, which is the opposite of traditional scenarios.

Benefit Rating: Moderate

Market Penetration: 5% to 20% of target audience

Maturity: Adolescent

Sample Vendors: Amazon Web Services; Ansible; Chef; Fugue; Google; HashiCorp; Microsoft; Puppet; SaltStack; Turbot

Recommended Reading: “Top 10 Technologies That Will Drive the Future of Infrastructure and Operations”

“Programmable Infrastructure Is Foundational to Infrastructure-Led Disruption”

“Adapting Vulnerability Management to the Modern IT World of Containers and DevOps”

“Solution Path for Infrastructure Automation”

“How to Make Cloud More Secure Than Your Own Data Center”

Site Reliability Engineering

Analysis By: George Spafford; Daniel Betts

Definition: Site reliability engineering (SRE) is a collection of systems and software engineering principles used to design and operate scalable resilient systems. Site reliability engineers work with the customer or product owner to understand operational requirements and define service-level objectives (SLOs). The site reliability engineer then collaborates with IT stakeholders to design and continuously improve systems that will meet the SLOs. For products or platforms that meet SRE guidelines, the engineer may choose to provide operational support.

Position and Adoption Speed Justification: SRE is a discipline originally created by Google, and was described in the 2016 book, “Site Reliability Engineering: How Google Runs Production Systems.” Adoption interest continues to grow both by digital-native organizations as well as traditional enterprises. SRE emphasizes the engineering disciplines that lead to resilience, but individual organizations implement SRE in widely varying ways. SRE is a complementary practice for organizations seeking to scale their DevOps activities.

SRE is intended to help manage the risks of rapid change, through the use of service-level objectives (SLOs), “error budgets,” monitoring, automated rollback of changes and organizational learning. SRE teams are often involved in code review, looking for problems that commonly lead to operational issues (for instance, an application that does not do log cleanup and therefore may run out of storage). They also ensure that the application comes with appropriate monitoring and resilience mechanisms, and that the application meets SRE approved standards or guidelines set to achieve negotiated SLOs. SRE teams can serve as an operations function and nearly all such teams have a strong emphasis on blameless root-cause analysis. This is to decrease the probability and/or impact of future events and enable organizational learning, continual improvement and reductions in unplanned work.

SRE practices are being adopted by organizations that need to deliver digital business products reliably. These practices require a culture that supports learning and improvement, highly skilled automation practices (and usually DevOps), usage of infrastructure as code capabilities (which usually requires a cloud platform). SRE also uses automation to reduce manual processes, leverages resilient system engineering principles, and an agile development process that employs continuous integration/continuous deployment (CI/CD).

User Advice: Organizations can benefit from SRE principles even if they are not sufficiently mature, agility-focused, or large enough to adopt SRE as a primary operations model. The SRE principles for risk management, release engineering, handling service-level objectives, monitoring, automation, and self-healing can be applied to a broader range of products and platforms. SRE also represents a useful means to scale DevOps initiatives.

An SRE initiative should have an executive sponsor. The first opportunity to begin with should have the following characteristics:

- The target application must change rapidly yet maintain high availability in order to maximize business value. Stakeholders should be politically friendly.
- The pilot must demonstrate sufficient value to improve credibility and support, yet also have an acceptable level of risk, allowing the stakeholders to learn.
- The initial SRE team must have a collaborative engineering mindset, strive to continuously learn and improve, and desire to automate tasks to reduce repetitious manual work, which is known as “toil.” It is often easiest to move DevOps-skilled employees from different parts of the organization, due to the relative difficulty of hiring engineers with SRE experience. A site reliability engineer is typically a software engineer with an excellent understanding of operations, or, less frequently, an infrastructure and operations engineer with strong programming skills.
- There must be clear SLOs that can be continuously monitored and reported against.
- The SRE collaborates with developers to help them learn how to design and build their product to meet the defined SLOs — the SRE is not doing the actual development work or inspecting quality in.

- The application development team must collaborate with the SRE team to meet SLOs. Developers are responsible for a resilient architecture and reliable code. SREs should not spend more than 50% of their time on ad hoc operational activities. Any excess should go to the developers for support.

An iterative approach must be used to start and evolve SRE practices. The teams involved must share experiences and lessons learned.

Business Impact: The SRE approach to DevOps is intended for products and platforms that need to deliver customer value at speed at scale while managing risk. The two primary use cases are to improve reliability of existing products or platforms as well as to in creation of new products or platforms that warrant the investment in reliability.

Benefit Rating: Transformational

Market Penetration: 5% to 20% of target audience

Maturity: Adolescent

Recommended Reading: “DevOps Teams Must Use Site Reliability Engineering to Maximize Customer Value”

“SRE and DevOps: End-to-End Accountability”

“Agile and DevOps Primer for 2020”

“Innovation Insight for Chaos Engineering”

“Maverick* Research: Software Testing and the Illusion of Exterminating Bugs”

Infrastructure Automation

Analysis By: Chris Saunderson

Definition: Infrastructure automation (IA) tools allow DevOps and I&O teams to design and implement self-service, automated delivery services across on-premises and cloud environments. IA tools enable DevOps and I&O teams to manage the life cycle of services through creation, configuration, operation and retirement. These infrastructure services are then exposed via API integrations to complement broader DevOps toolchains, or consumed via an administration console.

Position and Adoption Speed Justification: As a discipline, infrastructure automation evolved from the need to drive speed, quality and reliability with scalable approaches for deploying and managing systems. DevOps and I&O teams are using IA tools to automate delivery and configuration management of their IT infrastructure at scale and with greater reliability.

I&O leaders must automate processes and leverage new tools to mature beyond simple deployments of standardized platforms and deliver the systemic, transparent management of platform deployments. IA tools deliver the following key capabilities to support this maturation:

- Multicloud/hybrid cloud infrastructure orchestration
- Support for immutable infrastructure
- Support for programmable infrastructure
- Self-service and on-demand environment creation
- Resource provisioning
- Configuration management

IA tools have become increasingly similar in the breadth of their configuration management content and enterprise capabilities. IA vendors are developing greater knowledge of configuration artifacts and state, activity patterns, roles, and policy. Vendors are leveraging these insights to prevent misconfigurations, resolve problems and provide more advanced deployment and optimization capabilities.

As IA tools are increasingly accepted by development and I&O groups, organizations are looking to replace their tribal implementations with an enterprisewide IA tool strategy.

User Advice: Because IA tools provide a programmatic framework, the costs associated with them extend beyond just the licensing cost (or the lack thereof), so enterprises should include professional services and training requirements in cost evaluations. In particular, most I&O organizations should expect to invest in training because not all infrastructure administrators have the skills needed to use these tools successfully. IA tools have a learning curve, and it is tempting for developers and administrators to revert to known scripting methods to complete specific tasks. DevOps and IT operations leaders who want to maximize the value of IA tool investments must ensure that their organizations' culture can embrace IA tools strategically.

Use the following criteria to determine which IA vendor and product is appropriate:

- Internal IT skills
- Ecosystem surrounding IA tools
- Method for interacting with managed systems
- Security and compliance capabilities
- Authentication and authorization support
- Alignment to other tools within operating environment
- Orchestration functionality
- Scalability
- Platform and infrastructure content support

Business Impact: By enabling infrastructure administrators and developers to automate the deployment and configuration of settings and software in a programmatic way, organizations across all verticals stand to realize:

- Agility improvements — By enabling continuous integration and delivery concepts to IT infrastructure management.
- Productivity gains — Via faster deployment and repeatable, version-controlled configuration of infrastructure.
- Cost-reduction improvements — Via significant reductions in required manual interactions by highly skilled and high-cost staff by automating "day 2" operational tasks. Licensing cost reductions may also be achieved.
- Risk mitigation — Compliance improves via the consistent use of standardized, documented processes and configurations across physical and virtual infrastructures.

IA tools can drive efficiencies in operational configuration management, as well as provide a flexible framework for managing the infrastructure of DevOps initiatives. They achieve this by integrating with other toolchain components — continuous integration (CI) and application release orchestration — in support of continuous delivery.

Benefit Rating: High

Market Penetration: 20% to 50% of target audience

Maturity: Early mainstream

Sample Vendors: Amazon Web Services (AWS); Chef; HashiCorp; Inedo; Microsoft Azure; Pulumi; Puppet; Quali; VMware

Recommended Reading: "Market Guide for Infrastructure Automation Tools"

"The Future of DevOps Toolchains Will Involve Maximizing Flow in IT Value Streams"

"To Automate Your Automation, Apply Agile Practices and DevOps Tools to Infrastructure and Operations"

"How to Lead Digital Disruption With Programmable Infrastructure"

"Assessing HashiCorp Terraform for Provisioning Cloud Infrastructure"

Application Release Orchestration

Analysis By: Daniel Betts

Definition: Application release orchestration (ARO) tools combine of deployment automation, pipeline and environment management with release orchestration capabilities to simultaneously improve the quality, velocity and governance of application releases. ARO tools enable organizations to scale release activities across multiple diverse teams (e.g., DevOps), technologies, development methodologies (e.g., Agile), delivery patterns (e.g., continuous), pipelines, processes and toolchains.

Position and Adoption Speed Justification: ARO remains a valuable investment for clients to make, and where vendors are able to map to client internal delivery challenges/opportunities, success will be had. The market has changed: the current landscape of vendors has built feature-comparable products, with some incremental differences. This poses a challenge covering the space, as it is both maturing (feature parity across supplier platforms) and disrupted (the environments around the ARO space are pressing inwards on the core functionality).

Demand for new applications and features delivered faster to support business agility continues to and will grow for the foreseeable future. As a result, the tumultuous and transformative activity (often in the form of DevOps initiatives) that results has created multiple buyers for ARO solutions. These buyers often desperately need ARO's cohesive value yet are challenged to articulate and/or gain consensus around the business criticality of release activities to drive their acquisition.

User Advice: Simplify and speedup the transition to automated workflows by prioritizing and documenting current application release procedures, activities and artifacts performed by both traditional and DevOps teams. Organize activities into three main categories that align with ARO critical capabilities: deployment automation, pipeline and environment management, and release orchestration. Prioritize capabilities according to your current and future needs prior to evaluating vendor offerings. The better understanding you have of your current release activities (especially where they are done manually), the faster you are likely to see value from any ARO tool.

When evaluating ARO tools, features should remain the No. 1 driver for selection of vendors to evaluate. Requirements should be mapped to vendor capabilities as a part of the evaluation process and ongoing assessment of fit should be encouraged. Where legacy environments (most cases, legacy — older tech stacks, some cases, first-generation client products) are in place, those features should be weighted more heavily than others. Vendors continue to improve overall dashboard delivery of not only release performance, but also the underlying platforms metrics — code deployment cycles, lead time from commit to deploy, incident recovery, change failure rate — and the contributing factors that drive them.

Areas of future opportunity exist to incorporate evolving needs into ARO platforms: value stream mapping, AI Ops, software delivery management and DevSecOps. While many ARO purchases are built around supporting a modern enterprise release management capability, just as many successful ARO implementations started with a focus around a specific platform, application or team, extending their value to others.

Business Impact: By automating the deployment of code, management of environments and coordination of people in support of a continuous delivery pipeline, organizations across verticals stand to realize:

- Agility and productivity gains — Via faster delivery of new applications and updates in response to changing market demands

- Cost reduction — Via a significant reduction of manual interactions by high-skill and high-cost staff, freeing them to work on higher-value activities
- Risk mitigation — Via the consistent use of standardized documented processes and configurations across multiple technology domains
- Improvement and remediation — Via use of dashboard views over metrics outlining and predicting release quality and throughput

ARO tools provide transparency improvements to the release management process by making visible bottlenecks and wait states in areas such as infrastructure provision or configuration management. Once these constraints are visible and quantifiable, business-value decisions can be made to address them and measure the improvement. This speeds the realization of direct business value as new applications and enhancements/bug fixes can be delivered more quickly and reliably.

Benefit Rating: High

Market Penetration: 20% to 50% of target audience

Maturity: Early mainstream

Sample Vendors: Broadcom (CA Technologies); Chef; CloudBees (Electric Cloud); Digital.ai; GitLab; IBM (UrbanCode); Microsoft

Recommended Reading: “Magic Quadrant for Application Release Orchestration”

“Critical Capabilities for Application Release Orchestration”

“How to Build and Evolve Your DevOps Toolchains”

Container Management

Analysis By: Dennis Smith

Definition: Container management supports the management of containers at scale. This category of software includes container runtimes, container orchestration and scheduling, resource management and other container management capabilities. Container management software brokers the communication between the continuous integration/continuous deployment (CI/CD) pipeline and the infrastructure via APIs, and aids in the life cycle management of containers. It can also be used to more efficiently package COTS applications.

Position and Adoption Speed Justification: Gartner surveys show that the demand for containers continues to rise. This is likely due to the growing adoption of container runtimes, which have introduced common container packaging formats that are more easily consumable by, and useful to, application developers and those with a DevOps approach to IT operations. Container runtimes, frameworks and other management software have increased the utility of containers by providing capabilities such as packaging, placement and deployment, and fault tolerance (e.g., cluster of nodes running the application). The emergence of de facto standards (e.g., Kubernetes) and offerings from the public cloud providers are also driving adoption. Container management integrates these various elements to simplify deploying containers at scale. Many vendors enable the management capabilities across hybrid cloud or multicloud environments by providing an abstraction layer across on-premises and public clouds. Container management software can run on-premises, in public infrastructure as a service (IaaS) or simultaneously in both for that purpose.

The most common use of containers is focused specifically on Linux environments, and management software follows accordingly; however, there has been a gradual adoption of Windows containers. Container-related edge computing use cases have also increased, along with deployments involving bare-metal servers and the emergence of operational control planes that support containers and VMs.

Among the functionalities that container management systems provide are orchestration and scheduling, monitoring and logging, security and governance, registry management, and links to CI/CD processes. Among the vendor offerings are hybrid container management software, public cloud IaaS solutions specifically designed to run containers and PaaS frameworks that have incorporated integration with container management software. All major public cloud service providers are now deploying on-premises container solutions.

There is a high degree of interest in, and awareness of, containers within global organizations. Though many enterprises are planning or have recently commenced container deployments, few have containerized a significant portion of their application workloads. Additionally, there is significant grassroots adoption from individual developers who use containers with increasing frequency in development and testing — particularly for Linux. Container management software has progressed from an early-adopter technology to adolescent, where it remains.

User Advice: Organizations should begin exploring container technology as a means for packaging and deploying applications and their runtime environments. Depending on the environment, container management tools are often deployed complementarily with continuous configuration management tools. As container integration is added to existing DevOps tools and to the service offerings of cloud IaaS and PaaS providers, DevOps-oriented organizations should experiment with altering their processes and workflows to incorporate containers. An organization may be a good candidate if it meets the following criteria:

- It’s DevOps-oriented or aspires to become DevOps-oriented.
- It has high-volume, scale-out applications with a willingness to adopt microservices architecture, or has large-scale batch workloads.
- It has aspirational goals of increased software velocity and immutable infrastructure.
- It intends to use an API to automate deployment, rather than obtaining infrastructure through a self-service portal.

Organizations must also factor in their desire for hybrid and/or multicloud deployments into vendor selection, as many vendors offer container management software that can be deployed in different cloud environments.

Business Impact: Container runtimes make it easier to take advantage of container functionality, including providing integration with DevOps tooling and workflows. Containers provide productivity and/or agility benefits, including the ability to accelerate and simplify the application life cycle, enabling workload portability between different environments and improving resource utilization efficiency and more. Container management software simplifies the art of achieving scalability and production readiness, and optimizes the environment to meet business SLAs.

Benefit Rating: High

Market Penetration: 20% to 50% of target audience

Maturity: Adolescent

Sample Vendors: Amazon Web Services; Google Cloud Platform; IBM; Microsoft Azure; Mirantis; Rancher Labs; Red Hat; VMware

Recommended Reading: “Best Practices for Running Containers and Kubernetes in Production”

“Market Guide for Container Management”

“Best Practices to Enable Continuous Delivery With Containers and DevOps”

Sliding Into the Trough

Continuous Delivery

Analysis By: Hassan Ennaciri

Definition: Continuous delivery (CD) is an approach that enables DevOps teams to create an automated pipeline for producing software in short cycles. CD ensures that software can be reliably released any time via a DevOps toolchain and is a key capability of a DevOps initiative.

Position and Adoption Speed Justification: Growing DevOps initiative success continues to drive enterprise investments in CD capabilities. CD improves release velocity and reliability while simplifying compliance enforcement via automation. Continuous integration (CI), automation and testing are core to CD. These functions provide environment models that can be leveraged throughout the software development life cycle (SDLC) to more consistently deploy application builds and updates.

CD is a nonprescriptive, evolving approach that DevOps teams can deliver and realize in many ways. Given the emerging state of CD, market demand and vendor responses have been fragmented. DevOps teams typically start by automating functions that can clearly demonstrate the value of CD (e.g., application deployment and release configuration) when integrated with CI and testing. As a logical linkage between CI and operational functions, CD plays a critical role in building scalable DevOps toolchains.

User Advice: DevOps teams should incorporate CD processes to help reduce friction throughout the SDLC. They must also evaluate and invest in associated tooling, such as application release orchestration tools, containers and continuous configuration automation tools. These tools provide some degree of environment modeling and management, which can prove invaluable for scaling CD capabilities across multiple applications.

When starting a CD initiative, enterprises must consider all associated technologies and take an interactive approach to adoption. This will require collaboration with all stakeholders from product, development, security and operations. To enable a higher likelihood of CD success, DevOps teams must also establish consistency across application environments and implement a continuous improvement process that relies on proficiency metrics. DevOps product teams should assume that there will be continual discoveries about roles and responsibilities, required skills, automation details and documentation, especially during the early phases of adoption. DevOps teams should build requirements for CD tools with a broader view than just one environment (development, test, quality assurance, preproduction or production) and one application (for example, Java and .NET). The primary application of CD is to extend CI processes, but organizations also need to consider applying CD principles to infrastructure automation.

Business Impact: CD is a key capability of a DevOps initiative that reduces build-to-production cycle time. This accelerates the positive impact of new applications, functions, features and fixes by increasing velocity across the application life cycle. The positive impacts include improved business delivery and end-user satisfaction, improved business performance and agility, and risk mitigation via rapid delivery of updates.

Benefit Rating: High

Market Penetration: 20% to 50% of target audience

Maturity: Adolescent

Sample Vendors: Broadcom; Chef; CloudBees; GitLab; Harness; IBM; Microsoft; Puppet

Recommended Reading: “How to Build and Evolve Your DevOps Toolchains”

“The Future of DevOps Toolchains Will Involve Maximizing Flow in IT Value Streams”

“Magic Quadrant for Application Release Orchestration”

“Critical Capabilities for Application Release Orchestration”

Bimodal IT Operations

Analysis By: George Spafford

Definition: Bimodal IT operations comprise a strategy that enables infrastructure and operations (I&O) leaders to support customers based on the certainty of requirements using two complementary sets of work style capabilities (Mode 1 and Mode 2). Mode 1 capabilities are used when requirements are well-understood and can lead to predictable and well-modeled IT services or products. Mode 2 is used when requirements are uncertain and exploration is required, such as during the pursuit of digital business strategies.

Position and Adoption Speed Justification: I&O leaders are experiencing an acceleration toward approaches a new “normal” that embraces Mode 2 ways of work. Capabilities developed in regards to people, processes and technologies can traverse both certainty and uncertainty. Improved guidance is provided in Gartner’s operating model research. As such, this innovation profile is being obsoleted before the plateau.

User Advice: The pursuit of new digital business strategies, improved customer experience, IoT, AI/ML, and advanced analytics and other initiatives that involve exploration and high uncertainty place a high priority on agility and mean time to value for project and product teams. At the same time, these strategies will involve relatively high levels of uncertainly in the business and IT environment. These strategies require Mode 2 capabilities, different from what I&O traditionally delivered.

Mode 2 will require streamlined and automated approaches, such as DevOps and continuous delivery, plus changes to existing processes and functions. I&O leaders must also identify which teams building what types of solutions will require Mode 2 capabilities. They must then assemble a core team to focus on how to start in a focused iterative manner, how to learn and how to improve.

Although the I&O capabilities required for Mode 1 systems are familiar to enterprise I&O, with an emphasis on reliability, risk mitigation and long cycle times, these too are starting to evolve. As enterprises invest in more Mode 2 initiatives, this will put pressure on the systems of record to increase change frequency, especially for those systems that make up the digital business platform. Therefore, these systems must be renovated and will require agile development and DevOps. As a result, although both modes are necessary, many I&O personnel will eventually need to learn new skills, methodologies and tools to increase enterprise agility and improve business outcomes.

Business Impact: To be better positioned to support the needs of the organization in the long term, I&O must utilize infrastructure-led disruption and develop an operating model to deliver customer value.

Benefit Rating: High

Market Penetration: 20% to 50% of target audience

Maturity: Mature mainstream

Recommended Reading: “Embrace an I&O Operating Model to Support Digital Transformation”

“What Is an I&T Operating Model, and How Do You Accelerate Its Design Process?”

“Dare to Disrupt With ILD: A Gartner Trend Insight Report”

“Agile and DevOps Primer for 2020”

DevOps Toolchain

Analysis By: Thomas Murphy

Definition: A DevOps toolchain comprises tools used to support DevOps pipeline activity and provide fast feedback. It has come to focus primarily on the code to cloud build/test/deploy sequence. Pipeline activities have started with discrete tools for various steps, but we are increasingly seeing vendors deliver solutions right across the application development and delivery cycle. The mix of tools is determined by business need, product platform, language, product domain and the skills of the people who will use the tools.

Position and Adoption Speed Justification: DevOps toolchains emerge from the need to deliver new and changed applications faster. They can include dozens of unintegrated tools, which makes automation a technically complex and arduous task. But the biggest challenge facing organizations does not arise from the tools themselves, or their diversity, but rather from the belief that DevOps is achieved simply through tooling. Even if it is tooling, Gartner's DevOps survey found that organizations have, on average, 28 toolchains, which represents a large undertaking to create and maintain.

The market continues to evolve via acquisitions, the emergence of open-source and new commercial products, and the continued development of cloud architecture. Core tooling around CI is evolving with new componentized systems that make it easier to build and maintain a build script. Pipelines are gaining integrated security features and evolving support around package management and containers. As these core pipelines evolve, we also are seeing a new wave of "toolchains" that is broader and more encompassing emerging around value stream management. We expect organizations will have multiple toolchains of the pipeline variety and these will feed data into value stream tools.

User Advice: We recommend that I&O and application leaders develop a toolchain strategy to establish business objectives, identify practices to achieve those objectives, and then select tools to support those practices. The selection of tools should be the last part of the process.

Software engineering practices such as version control, code management and managed distribution should be utilized. The toolchain should be focused on removing execution barriers and automating the development and continuous delivery process. Each DevOps product or platform team member should understand the capabilities and contribution of each tool in the DevOps toolchain in order to avoid tool overlap, conflict and toolchain functionality gaps.

Remember that tools — even open-source ones — are not free. There is a cost attached to learning, integrating and (especially when they are integrated) replacing them.

Therefore DevOps leaders should:

- Expect to have more than one toolchain to support the different technology stacks and delivery platforms targeted (e.g., COTS, cloud, mainframe, container-native)
- Seek to utilize pipeline integrated security and compliance and focus on effective test automation

- Establish a toolchain community of practice to help manage and evolve toolchains and monitor new technology developments

Business Impact: Delivering business value is central to DevOps. This requires a collaborative product team focused on delivering applications when the business demands them, which, in turn, demands an agile mindset and the technology to support the activity needs of individual team members. A well-designed, integrated and automated DevOps toolchain enables development and operations team members to work together, with common objectives and metrics, to ensure quality, on-time application delivery to the business.

Benefit Rating: High

Market Penetration: 20% to 50% of target audience

Maturity: Early mainstream

Sample Vendors: Amazon Web Services; Atlassian; CloudBees; Codefresh; GitLab; Harness; HashiCorp; Microsoft

Recommended Reading: "The Future of DevOps Toolchains Will Involve Maximizing Flow in IT Value Streams"

"How to Build and Evolve Your DevOps Toolchains"

"Guidance for Securing the DevOps Toolchain With IAM"

"Ignition Guide to Managing a DevOps Toolchain"

"Four Steps to Adopt Open-Source Software as Part of the DevOps Toolchain"

Software-Defined Infrastructure

Analysis By: Philip Dawson

Definition: Software-defined infrastructure (SDI) includes the broad set of software-defined anything (SDx) infrastructure components and the software-defined data center (SDDC). SDI also includes non-data-center infrastructure deployed in Internet of Things (IoT) applications and an SD edge of edge-based adapters, monitoring devices, gateways, appliances and machines.

Position and Adoption Speed Justification: Data center infrastructure is well-covered with compute (SDC), network (SDN) and storage (SDS), but SDI also extends to non-data-center infrastructure with the use of monitoring devices or machines that are software-defined. This is enabled through the use of sensors and adapters that are abstracted through software, becoming SDI in edge, IoT and operational technology (e.g., retail POS), rather than traditional, IT-driven SDI through data center or cloud. In 2020, we are seeing SDI move to vendor-specific silo technology (not heterogeneous service drive) and, hence, obsolete as multivendor interoperable standards.

User Advice: As SDI initiatives roll out, consider the integration and measurement of non-data-center edge infrastructure. Focus on core IT SDI for compute, network, storage and facilities, but consider the impact of SDI on IoT, edge computing, remote office/branch office (ROBO) and other operational technologies. Key verticals operating in multiple, geographically distributed locations (such as retail, manufacturing, retail banking, distribution and utilities) are extending IoT and non-data-center SDI initiatives for new IT operations and functions. Expect SDI to be tied to a specific vendor or technology silo.

Business Impact: With the increase of IoT touching edge-based operational technology, SDI reaches beyond and between SDDCs, and leverages SDI benefits and features for new multimode applications and edge IoT endpoints. However, SDI is now tied to vendor technology not interoperability.

Benefit Rating: High

Market Penetration: 20% to 50% of target audience

Maturity: Obsolete

Sample Vendors: IBM; Intel; Microsoft; Red Hat; VMware; Wipro

Recommended Reading: “Simplify Intelligent Infrastructure by Using Workload Architectures”

“Drive Administration, Application and Automation Capabilities of Infrastructure-Led Disruption”

Climbing the Slope

DevSecOps

Analysis By: Neil MacDonald; Mark Horvath

Definition: DevSecOps is the integration of security and compliance testing into emerging agile IT and DevOps development pipelines as seamlessly and transparently as possible, ideally without reducing the agility or speed of developers or requiring them to leave their development environment. Ideally, offerings provide security protection at runtime as well.

Position and Adoption Speed Justification: Originally proposed by Gartner in 2012, adoption of DevSecOps takes time, but interest is high. Several security vendors directly target DevSecOps use cases, and mainstream adoption is less than five years away. Slow rates of adoption and slow movement on the Hype Cycle are primarily due to the process and cultural changes required across IT organizational silos to adopt agile “DevOps like” models and to include security in this to deliver DevSecOps. However, industry initiatives around “secure DevOps,” DevSecOps, DevOpsSec, and “rugged DevOps” have gained significant traction. Leading security vendors are evolving their solutions to become more programmable, laying the foundation for higher levels of automation and orchestration from testing into deployment.

DevOps (often combined with container/Kubernetes adoption and programmatic cloud infrastructure) is being driven by developers in the name of speed and agility. Security must be a part of this shift, but in a way that respects the collaborative nature of DevOps. Security cannot be siloed, which forces developers outside of their toolchain to perform security testing. DevSecOps offerings need to programmatically integrate with common CI/CD testing tools to support automation and without requiring a security professional to be involved — other than setting policy guiderails to be applied. In addition, the declarative nature of scripts and cloud automation tools used in development can be tied automatically to programmable security infrastructure for protection at runtime. Whether you drive security from container layers, scripts, templates or toolchains, the desired outcomes are the same — the automated and compliant configuration of the underlying security infrastructure based on policy, reflecting the intended state of the workloads.

User Advice: Investigate the adoption of a DevSecOps operating model for next-generation data center infrastructure or public-cloud-based computing environments to automatically link security policy enforcement mechanisms with the deployment of new workloads. As your organization investigates DevOps operating models or moves toward more-agile IT development and operations processes, consider these actions:

- Prepare security and risk management teams for automated integration with DevOps initiatives, and identify the primary skills and technology gaps.
- “Shift left” and make security testing tools and processes available earlier in the development process, ideally as the developers are writing code.
- As zero vulnerability applications aren’t possible, favor automated tools with fast turnaround times with a focus on reducing false positives and allowing developers to concentrate on the most critical vulnerabilities first.
- Start identifying OSS components and vulnerabilities in development as a high-priority project (referred to as software composition analysis), as the biggest risk comes from known vulnerabilities and misconfigurations.
- Invest in programmable security infrastructure capable of supporting security policy toolchains, which facilitates speed through automation and flexibility via open APIs (typically, REST-based). Require your security vendors to support “out of the box” integration with common development toolchain vendors and also support full API enablement of their offerings for automation.
- Require security controls to understand and be capable of applying security policies in container and Kubernetes-based development and deployment environments.
- Experiment with DevSecOps workflows using public cloud infrastructure and programmatic ways that security policies can be integrated into templates, blueprints and recipes to avoid manual security policy configuration.
- Favor offerings that can link scanning in development (including containers) to correct configuration and protection at runtime.

Business Impact: As IT development and operations processes become more agile (including shifts to DevOps operating models), security must not be an afterthought and should be seamlessly integrated into agile development processes — DevSecOps. Furthermore, the externalization of security policy enables business units and security organizations, not developers, to define appropriate policies. Policy-driven automation of security infrastructure improves compliance, the quality of security enforcement and developer efficiency, as well as overall IT effectiveness.

Benefit Rating: Transformational

Market Penetration: 20% to 50% of target audience

Maturity: Early mainstream

Sample Vendors: Aqua Security; Contrast Security; Data Theorem; NowSecure; Palo Alto Networks; ShiftLeft; Snyk; Sonatype; Synopsys; Veracode

Recommended Reading: "12 Things to Get Right for Successful DevSecOps"

"Integrating Security Into the DevSecOps Toolchain"

"Market Guide for Cloud Workload Protection Platforms"

"Magic Quadrant for Application Security Testing"

"Critical Capabilities for Application Security Testing"

"How to Make Cloud More Secure Than Your Own Data Center"

"DevSecOps Security Metrics: Use Your Code Repository to Start a Virtuous Cycle"

Application Performance Monitoring Software

Analysis By: Charley Rich; Federico De Silva

Definition: APM software is composed of:

- Front-end monitoring to observe and analyze the performance and behavior of end-user interactions with applications.
- Application discovery, tracing and diagnostics (ADTD), used to discover and detect the relationships between web/application servers, microservices and infrastructure using bytecode instrumentation and/or distributed tracing.
- Analytics uses domain-centric artificial intelligence for IT operations (AIOps) to detect patterns, anomalies and causality.

Position and Adoption Speed Justification: Gartner continues to see increasing growth in the market size (over 15.3% growth rate on revenue of $4.3B, see "Market Share: All Software Markets, Worldwide, 2019") driven by the continued acceleration of the digitalization of business processes and the realization that greater observability is needed to monitor the applications that are essential to digital business transformation.

APM vendors have expanded with integrated monitoring and analysis of infrastructure, including network, servers, databases, logs, containers, orchestration, service mesh, microservices, cloud services, the relation of performance metrics with business KPIs and business processes.

Increased emphasis on customer experience is stimulating the need for greater insight into the customer journey as users interact with digitalized business processes. This need goes beyond traditional front-end monitoring, as part of APM measuring latency and encompasses a broader set of capabilities including measuring end-user sentiment, as well as accomplishment of business objectives such as orders, claims and trades (see "Market Guide for Digital Experience Monitoring").

APM is playing a critical role in providing rapid feedback to developers about the impact of the most recent production deployments. They display performance analysis within the context of a release with some solutions providing bidirectional deployment tool integration and the sharing of tags between IT ops and DevOps.

APM vendors' adoption of open standards continues to expand. A number of APM vendors have become active in the group defining a future standard for distributed tracing, collection of metrics and logs, OpenTelemetry. (See "Magic Quadrant for Application Performance Monitoring.") This approach is developer-intensive but, with automation it may replace or expand the standard model of application monitoring, the agent. The proprietary nature of APM agents are also evolving, pushing more of the APM value proposition into the analytics realm, reducing vendor "stickiness." The mainstream adoption of OpenTelemetry could disrupt vendors that are slow to adopt these new technologies and could impact the rate of adoption of microservices. This is part of an ongoing trend toward self-reporting infrastructure.

APM solutions are increasingly adding machine learning (ML), a technology used in AIOps solutions, into their products. ML is being used to reduce the noise operators deal with, predicting and detecting anomalies and determining causality.

User Advice: As organizations continue to embrace digital transformation, their need for agility in order to deliver on these transformation initiatives increases. Therefore, APM solutions must support that need for agility, aid in its acceleration and effectiveness, and avoid being perceived as just another performance monitoring tool.

Choose vendors that assist in relating application performance to business objectives and serve not only IT operations, but also DevOps, application owners and lines of business, providing value throughout the life cycle of an application. Select a vendor that provides actionable answers and not just endless drill-downs to more data.

Choose APM vendors based on their abilities to support, the following:

- The mapping and monitoring of customer and business journeys
- Bidirectional integration with the DevOps tool chain
- New emerging standards in instrumentation such as OpenTelemetry
- Cloud-native monitoring with an API-first approach

Bidirectional integration with ITSM tools to bridge the gap between APM and ITSM tools (see “Avoid the Unexpected Consequences of IT Change Management With AIOps and CMDB”).

Business Impact: APM is critical for isolating problems, shortening MTTR, improving service availability and customer experience. Advanced capabilities can be used to better understand business processes by mapping instrumentation data to a business process, broadening APM’s value in performance monitoring to delivering insights into business operations. improving business relevance, communicating insight to business personas and in resolving the most important problems first. As a result, APM continues to be of use to a growing set of stakeholders including IT Ops, development/DevOps, CloudOps and application owners or line of business (see “Critical Capabilities for Application Performance Monitoring”).

Benefit Rating: High

Market Penetration: 20% to 50% of target audience

Maturity: Mature mainstream

Sample Vendors: Aternity; Broadcom (CA Technologies); Cisco (AppDynamics); Datadog; Dynatrace; IBM; Instana; New Relic; SolarWinds; Splunk

Recommended Reading: “Magic Quadrant for Application Performance Monitoring”

“Critical Capabilities for Application Performance Monitoring”

“Market Guide for Digital Experience Monitoring”

“Broaden Application Performance Monitoring to Support Digital Business Transformation”

“Advance Your Application Performance Monitoring Strategy to Support Microservices”

“2019 Strategic Roadmap for IT Operations Monitoring”

“I&O Leaders Must Use Monitoring Metrics to Optimize Customer Experience”

Cloud Management Platforms

Analysis By: Dennis Smith

Definition: Cloud management platforms (CMPs) enable organizations to manage private, public and multicloud services and resources. Their specific functionality is a combination of provisioning and orchestration; service request management; inventory and classification; monitoring and analytics; cost management and resource optimization; cloud migration, backup and disaster recover; and identity, security and compliance. This functionality can be provided by a single product or a set of vendor offerings with some degree of integration.

Position and Adoption Speed Justification: While the CMP market is continually changing, vendors and enterprise customers are getting a better feel about where such tooling can and cannot be used. Vendors are still being challenged with evolving customer requirements (for example, interfacing with multiple public clouds, cost transparency with workload optimization to remediate cost overruns and handling newer functions like containers and serverless deployments). At the same time, major market consolidation will continue. For example, many vendors, that initially targeted cost management, have been acquired as this functionality is becoming a part of the basic CMP. Additionally many vendors in adjacent markets are acquiring CMP vendors and combining this functionality with asset management (software and hardware) and SaaS operational management. Cloud service providers (CSPs) and management service providers (MSPs) are also entering the market. Additionally, many long-standing vendors are introducing next-generation products, often targeting holes that their previous products had. Finally vendors in different markets (e.g., monitoring) are also entering the market. Some of the core CMP functionality is also being combined (for example, monitoring and analytics with cost management and resource optimization). The ability to serve both application developer and I&O personas is the key. This requires that CMPs be linked into the application development process without imposing a workflow that inhibits agility while also allowing infrastructure and operations (I&O) teams to enforce provisioning standards.

Organizations have an increasing need to address multicloud requirements. In some cases, they want to become internal cloud service brokers (CSBs) and manage public services that were previously acquired — often by lines of business (LOBs) outside the I&O organization — and have become difficult to manage operationally.

User Advice: As CMP market volatility increases, IT organizations must:

- Consider CMP vendor's viability along with evaluating features.
- First consider native cloud services as an alternative or option versus CMPs, particularly if you favor depth with an individual cloud provider versus breadth across different cloud providers.
- Consider functionally focused tools (e.g., cloud expense management tool) if you only require a limited set of functionalities.
- Augment, swap out or integrate additional cloud management or traditional management tools for many requirements, because no vendor provides a complete cloud management solution.
- Standardize, because deriving value from your CMP will depend heavily on the degree of standardization offered by the infrastructure, software and services.
- Set realistic expectations on deployment times, as mature organizations implement CMP in a relatively short period (one to two years); however, less mature organizations may require two or more years to design effective, repeatable, and automatable standards and processes.
- Plan for new roles, such as cloud architects and cloud service brokers (CSBs), including developing skills in the financial management and capacity management areas.

Business Impact: Enterprises will deploy CMPs (increasingly as a part of a larger product suite) to increase agility, reduce the cost of providing services and increase the likelihood of meeting service levels. Costs are reduced and service levels are met because CMP deployments require adherence to standards, as well as increased governance and accountability. Desirable IT outcomes include:

- Policy enforcement (e.g., on reusable standard infrastructure components).

- Reduced lock-in to public cloud providers, although at the cost of CMP vendor lock-in that can slow innovation.
- Enhanced ability to broker services from various cloud providers and to make informed business decisions on which providers to use.
- Ongoing optimization of SLAs and costs.
- Management of SLAs and enforcement of compliance requirements.
- Health and performance monitoring of cloud applications.
- Accelerated development, enabling setup/teardown of infrastructure that mimics production, resulting in lower overall infrastructure costs and higher quality. This can be in support of DevOps initiatives.

Benefit Rating: Low

Market Penetration: 5% to 20% of target audience

Maturity: Mature mainstream

Sample Vendors: CloudBolt; Flexera; HyperGrid; Morpheus Data; Scalr; Snow Software; VMware

Recommended Reading: "Magic Quadrant for Cloud Management Platforms"

"Critical Capabilities for Cloud Management Platforms"

Continuous Configuration Automation

Analysis By: Chris Saunderson

Definition: Continuous configuration automation (CCA) tools enable infrastructure administrators and developers to automate the deployment and configuration of systems and software programmatically. They support the description of configuration states and settings, as well as the deployment of software binaries and configuration data. Most CCA tools have open-source heritage, with some offering commercial support. Commercial CCA tools have vendor support, role-based administration and more advanced management capabilities than open-source versions.

Position and Adoption Speed Justification: CCA tools have proven critical to DevOps initiatives due to their ability to manage and deliver infrastructure and associated software and configuration changes as code. This enables inclusion of CCA-specified infrastructure as code into DevOps pipelines. CCA tools have continued to expand their reach into networking, containers, and into compliance and security roles. These tools are critical to DevOps initiatives, due to their:

- Programmatic access to infrastructure elements
- Ability to allow IT operations personnel to build an automation life cycle modeled on a software development life cycle

- Ease of experimentation, extensibility and access to active communities
- Potentially lower total cost of ownership (TCO) for significant configuration management capability

Enterprise adoption of these tools is hindered mainly by the IT skill sets needed to use them. Many I&O organizations lack individuals with basic scripting skills and source code repository management skills, let alone the skill to conceptualize and manage infrastructure as code. Developers and administrators may use them on a tribal basis, further inhibiting enterprisewide adoption. The growing use of containers has also created confusion about the role of CCA tools; however, Gartner believes that CCA tools and containers can be used in a highly complementary manner.

Organizations are increasingly using CCA tools for a broader set of deployment and automation functions beyond configuration management — for example, patching, application release orchestration, configuration auditing (e.g., for regulatory or internal policy compliance) and orchestration. As CCA tools are increasingly used in adjacent functions, organizations will experience the advantages of using these tools in new ways, but will also discover limitations relative to tools that are purpose-built for functions other than configuration management. An approach based on consistently applying a curated catalog of CCA tools will deliver effective productivity improvements to clients.

User Advice: The overlap between infrastructure automation (IA) and CCA tools adds to the confusion around these tool markets. Clients should be clear about the role that CCA tools fulfill in their toolchain to ensure that duplication is minimized. Because CCA tools provide a programmatic framework, the costs associated with them extend beyond just the licensing cost (or the lack thereof), so enterprises should include professional services and training requirements into cost evaluations. In particular, most infrastructure and operations (I&O) organizations should expect to invest in training because not all infrastructure administrators have the skills needed to use these tools successfully. CCA tools have a learning curve, and it is tempting for developers and administrators to revert to known scripting methods to complete specific tasks. DevOps and IT operations leaders who want to maximize the value of CCA tool investments must ensure that their organizations' culture can embrace CCA tools strategically.

Use the following criteria to determine which CCA vendor and product is appropriate: internal IT skills, security and compliance capabilities, authentication and authorization support, alignment to other tools within operating environment, orchestration functionality, scalability, and platform support.

Business Impact: By enabling infrastructure administrators and developers to automate the deployment and configuration of settings and software in a programmatic way, organizations across all verticals stand to realize:

- Agility improvements — By enabling continuous integration and delivery concepts to IT infrastructure management.
- Productivity gains — Via faster deployment and repeatable, version-controlled configuration of infrastructure.

- Cost reduction improvements — Via significant reductions in required manual interactions by highly skilled and high-cost staff by automating “day 2” operational tasks. Licensing cost reductions may also be achieved.
- Risk mitigation — Compliance improves via the consistent use of standardized, documented processes and configurations across physical and virtual infrastructures.

CCA tools can drive efficiencies in operational configuration management, as well as provide a flexible framework for managing the infrastructure of DevOps initiatives. They achieve this by integrating with other toolchain components — notably, infrastructure automation tools, continuous integration (CI) and application release orchestration in support of continuous delivery.

Benefit Rating: High

Market Penetration: 20% to 50% of target audience

Maturity: Early mainstream

Sample Vendors: Ansible; CFEngine; Chef; Inedo; Orca; Puppet; SaltStack

Recommended Reading: “Market Guide for Continuous Configuration Automation Tools”

“Best Practices to Enable Continuous Delivery With Containers and DevOps”

“Use a Bimodal Approach to Improve the Selection of DevOps Continuous Configuration Automation Tools”

“Solution Path for Infrastructure Automation”

“Market Guide for Infrastructure Automation Tools”

Entering the Plateau

Agile Ops

Analysis By: George Spafford

Definition: Agile ops is a philosophy and not a methodology. The term “agile ops” refers to the increased need for I&O to rapidly respond to changes in customer requirements. In situations where application development is involved, this leads to the development of DevOps capabilities. In cases where there is no custom application development activity, then I&O must pragmatically leverage concepts from agile application development, lean, DevOps and site reliability engineering (SRE).

Position and Adoption Speed Justification: Agile ops is not a “practice” in and of itself. Instead, it is a pointer to proven methods that I&O leaders need to learn about and pragmatically adopt to improve the agility of their organizations. Many concepts from DevOps are leveraged to improve responsiveness. DevOps has reached the plateau in 2020 and this means the concepts underpinning agile ops have as well.

User Advice: For I&O leaders who have been mandated to adopt agile, or have identified the need to "improve agility," identify which of the following four scenarios seems to best fit:

1. Is the requirement a general need to improve I&O's ability to respond quickly when changes are needed but does not directly include application development?
If so, investigate the use of Kanban, Gemba Kaizen, collaboration, small teams and extensive automation. A great deal can be learned by looking at the practices used in lean, Theory of Constraints, DevOps and SRE.
2. Is there a desire to adopt agile product management techniques for I&O?
If so, investigate the use of scrum and Kanban along with lean techniques, minimum viable product and continuous improvement.
3. Is there a desire to actually use agile development and proper software engineering principles in the increasingly programmable space of I&O?
If so, investigate both agile and DevOps for these situations along with product and platform organizational models. Start to regard the infrastructure as a set of software products to be properly designed, documented, tested, version controlled and placed in a shared repository with proper metatagging.
4. Is there a desire to improve I&O agility to work more closely with agile application development teams?

If so, investigate DevOps and collaborate with stakeholders to begin iterative improvements.

Business Impact: I&O groups seeking to improve their ability to respond to changes in customer requirements can obtain significant improvements in the realization of value.

Benefit Rating: High

Market Penetration: 20% to 50% of target audience

Maturity: Mature mainstream

Recommended Reading: "6 Steps to Increase DevOps Release Velocity by Removing Constraints"

"Use 8 Simple Steps to Get DevOps Right" "Agile and DevOps Primer for 2020"

"DevOps Teams Must Utilize Site Reliability Engineering to Maximize Customer Value"

"Five Steps to Increase Development Release Velocity"

"How to Scale DevOps by Building Platform Teams"

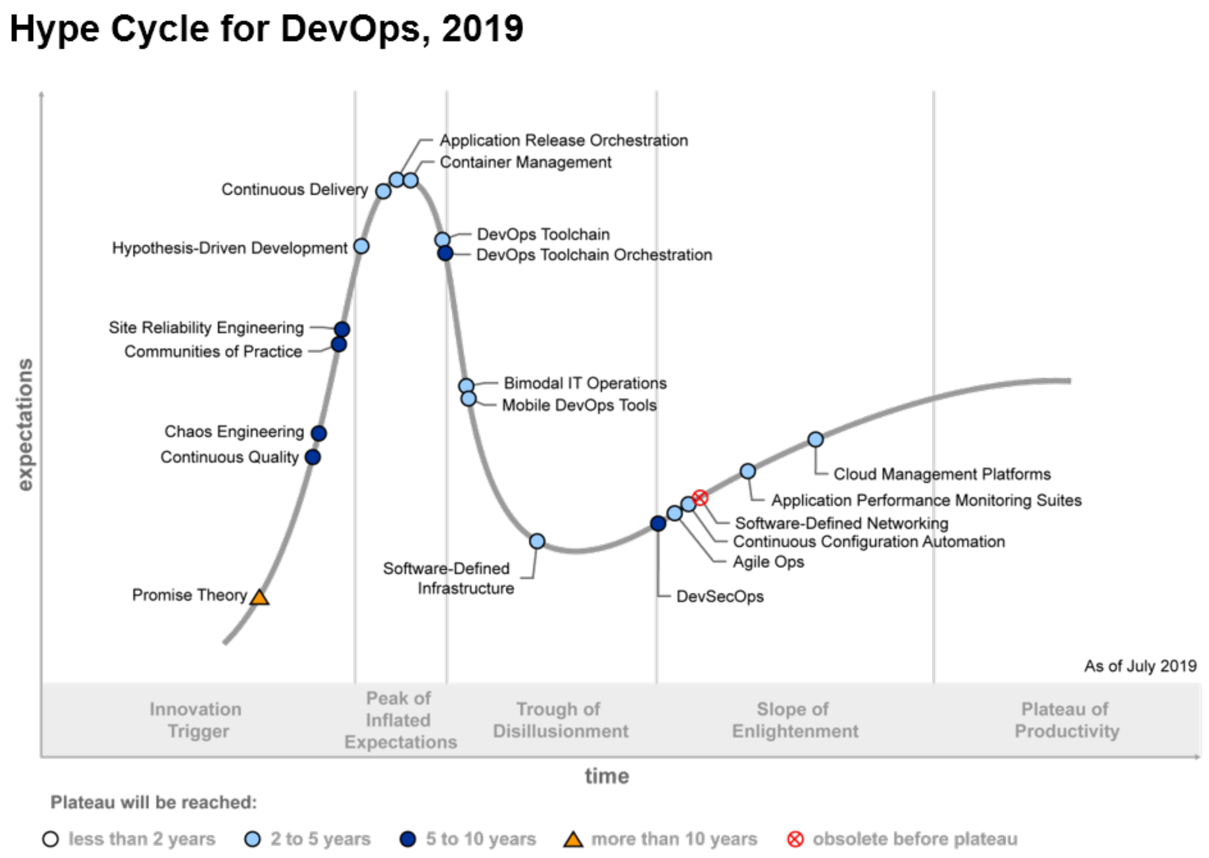
"How to Manage and Market Platforms as Products for DevOps Teams"

"Maximize the Success of Enterprise Agile: Utilize DevOps as the Accelerator"

"Avoid Agile Transformation Failure by Using Agile Coaches"

Appendixes

Figure 3. Hype Cycle for DevOps, 2019



Source: Gartner
ID: 369900

Hype Cycle Phases, Benefit Ratings and Maturity Levels

Table 1. Hype Cycle Phases

Phase	Definition
Innovation Trigger	A breakthrough, public demonstration, product launch or other event generates significant press and industry interest.
Peak of Inflated Expectations	During this phase of overenthusiasm and unrealistic projections, a flurry of well-publicized activity by technology leaders results in some successes, but more failures, as the technology is pushed to its limits. The only enterprises making money are conference organizers and magazine publishers.
Trough of Disillusionment	Because the technology does not live up to its overinflated expectations, it rapidly becomes unfashionable. Media interest wanes, except for a few cautionary tales.
Slope of Enlightenment	Focused experimentation and solid hard work by an increasingly diverse range of organizations lead to a true understanding of the technology's applicability, risks and benefits. Commercial off-the-shelf methodologies and tools ease the development process.
Plateau of Productivity	The real-world benefits of the technology are demonstrated and accepted. Tools and methodologies are increasingly stable as they enter their second and third generations. Growing numbers of organizations feel comfortable with the reduced level of risk; the rapid growth phase of adoption begins. Approximately 20% of the technology's target audience has adopted or is adopting the technology as it enters this phase.
Years to Mainstream Adoption	The time required for the technology to reach the Plateau of Productivity.

Source: Gartner (July 2020)

Table 2. Benefit Ratings

Benefit Rating	Definition
Transformational	Enables new ways of doing business across industries that will result in major shifts in industry dynamics
High	Enables new ways of performing horizontal or vertical processes that will result in significantly increased revenue or cost savings for an enterprise
Moderate	Provides incremental improvements to established processes that will result in increased revenue or cost savings for an enterprise
Low	Slightly improves processes (for example, improved user experience) that will be difficult to translate into increased revenue or cost savings

Source: Gartner (July 2020)

Table 3. Maturity Levels

Maturity Level	Status	Products/Vendors
Embryonic	In labs	None
Emerging	Commercialization by vendors Pilots and deployments by industry leaders	First generation High price Much customization
Adolescent	Maturing technology capabilities and process understanding Uptake beyond early adopters	Second generation Less customization
Early mainstream	Proven technology Vendors, technology and adoption rapidly evolving	Third generation More out-of-box methodologies
Mature mainstream	Robust technology Not much evolution in vendors or technology	Several dominant vendors
Legacy	Not appropriate for new developments Cost of migration constrains replacement	Maintenance revenue focus
Obsolete	Rarely used	Used/resale market only

Source: Gartner (July 2020)

Gartner Recommended Reading

Some documents may not be available as part of your current Gartner subscription.

Understanding Gartner’s Hype Cycles

Adopt an Iterative Approach to Drive DevOps Success in Large Organizations

Use 8 Simple Steps to Get DevOps Right

Agile and DevOps Primer for 2020

4 Steps to Create a DevOps Dojo That Accelerates Learning and Cultural Change

Innovation Insight for Continuous Quality

How DevOps Can Deliver Continual Customer Value Faster

Three Ways Midsize Enterprises Can Maximize Value From DevOps

DevOps Success Requires Shift-Right Testing in Production

Evidence

Gartner 2019 DevOps Survey. The results presented are based on a Gartner study conducted to assess the objectives, performance and challenges faced in DevOps initiatives. It also delves into the performance, drivers and challenges of scaling DevOps. The primary research was conducted online from 14 November through 18 December 2018 among 273 respondents in North America, Western Europe and the Asia/Pacific region.

Qualifying organizations span various industries, except services. Companies were screened for having a minimum annual revenue for fiscal year 2017 of $100 million and a minimum of 50 full-time IT employees. Companies were required to have DevOps to support systems/IT products in production. DevOps efforts were required to be completely in-house or a mix of in-house and outsourcing, with a minimum of five DevOps teams to support systems/IT products in production. The sample represents organizations in the U.S. (n = 83), Canada (n = 35), the U.K. (n = 44), Germany (n = 31), India (n = 48) and Australia/New Zealand (n = 32).

Respondents were required to have a role that is primarily IT-focused or a fairly even blend of business and IT. They were also required to be involved in decisions regarding DevOps efforts at their organizations. Quotas were applied for countries, industries and annual revenue.

GARTNER HEADQUARTERS

Corporate Headquarters
56 Top Gallant Road
Stamford, CT 06902-7700
USA
+1 203 964 0096

Regional Headquarters
AUSTRALIA
BRAZIL
JAPAN
UNITED KINGDOM

For a complete list of worldwide locations,
visit http://www.gartner.com/technology/about.jsp

It consists of the opinions of Gartner's research organization, which should not be construed as statements of fact. While the information contained in this publication has been obtained from sources believed to be reliable, Gartner disclaims all warranties as to the accuracy, completeness or adequacy of such information. Although Gartner research may address legal and financial issues, Gartner does not provide legal or investment advice and its research should not be construed or used as such. Your access and use of this publication are governed by Gartner Usage Policy. Gartner prides itself on its reputation for independence and objectivity. Its research is produced independently by its research organization without input or influence from any third party. For further information, see "Guiding Principles on Independence and Objectivity."