Hype Cycle for Agile and DevOps, 2021

Published 12 July 2021 - ID G00747574 - 111 min read

By Analyst(s): George Spafford, Joachim Herschmann

Initiatives: Infrastructure and Operations Leaders

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.

Additional Perspectives

- アジャイルとDevOpsのハイプ・サイクル：2021年
  (15 November 2021)

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 — such as information security and risk management — to improve the delivery of value at the release cadence, scale and level of risk that the customer requires. 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. DevOps has moved from being an 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, more reliably and/or more predictably.

Scaling DevOps efforts will require management of the overarching value stream and optimizing aspects of human factors, organization, process, technology and information exchanges (see recommended research for related notes).

For more information about how peer infrastructure and operations (I&O) leaders view the technologies aligned with this Hype Cycle, see 2021-2023 Emerging Technology Roadmap for Large Enterprises.

The Hype Cycle

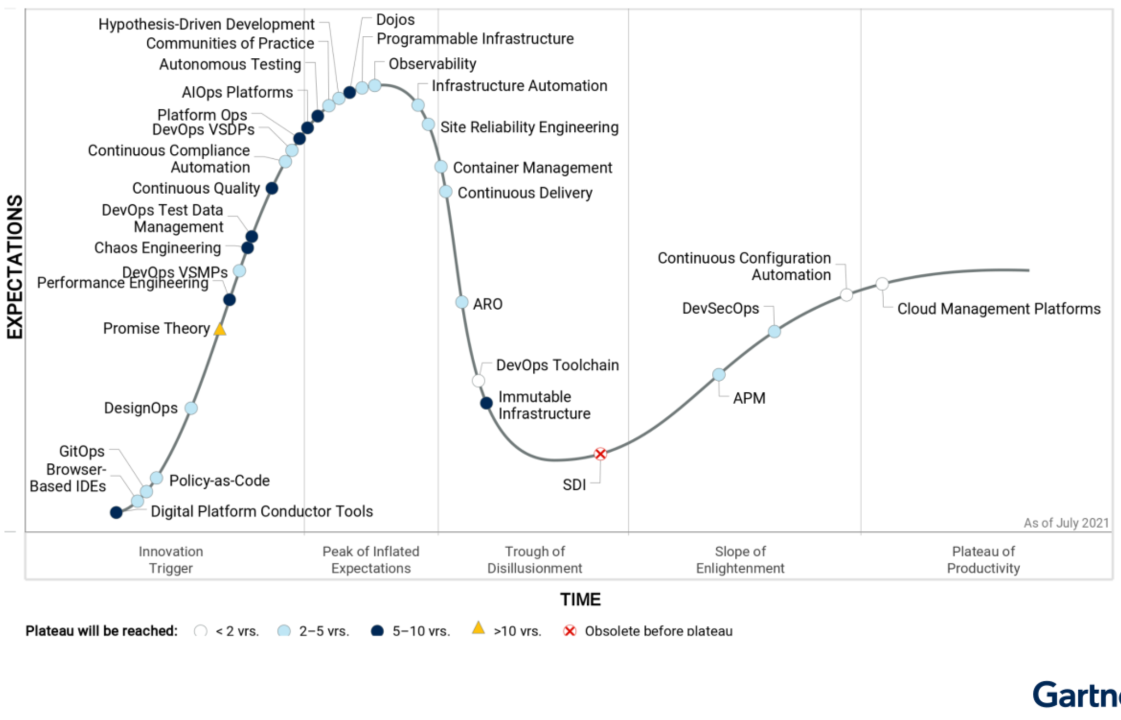
The DevOps Hype Cycle includes key approaches and technologies used to support a DevOps initiative. At this time, DevOps is a proven perspective to enable practitioners to optimize customer value, cost and risk. In the Achieve Business Agility With Automation, Continuous Quality and DevOps 2020 Survey, 90% of participants reported meeting or exceeding their top DevOps objectives. The attention has shifted from understanding the basics to determining how to scale efforts. DevOps leaders must continue to optimize the relevant value stream, requiring multiple IT disciplines to work together to optimize 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 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 unavoidable local optimization. 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 Recommended Research for more information).

Gartner clients emphasize 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. 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 Recommended Research).

Figure 1: Hype Cycle for Agile and DevOps, 2021



Source: Gartner (July 2021)

Downloadable graphic: Hype Cycle for Agile and DevOps, 2021

The Priority Matrix

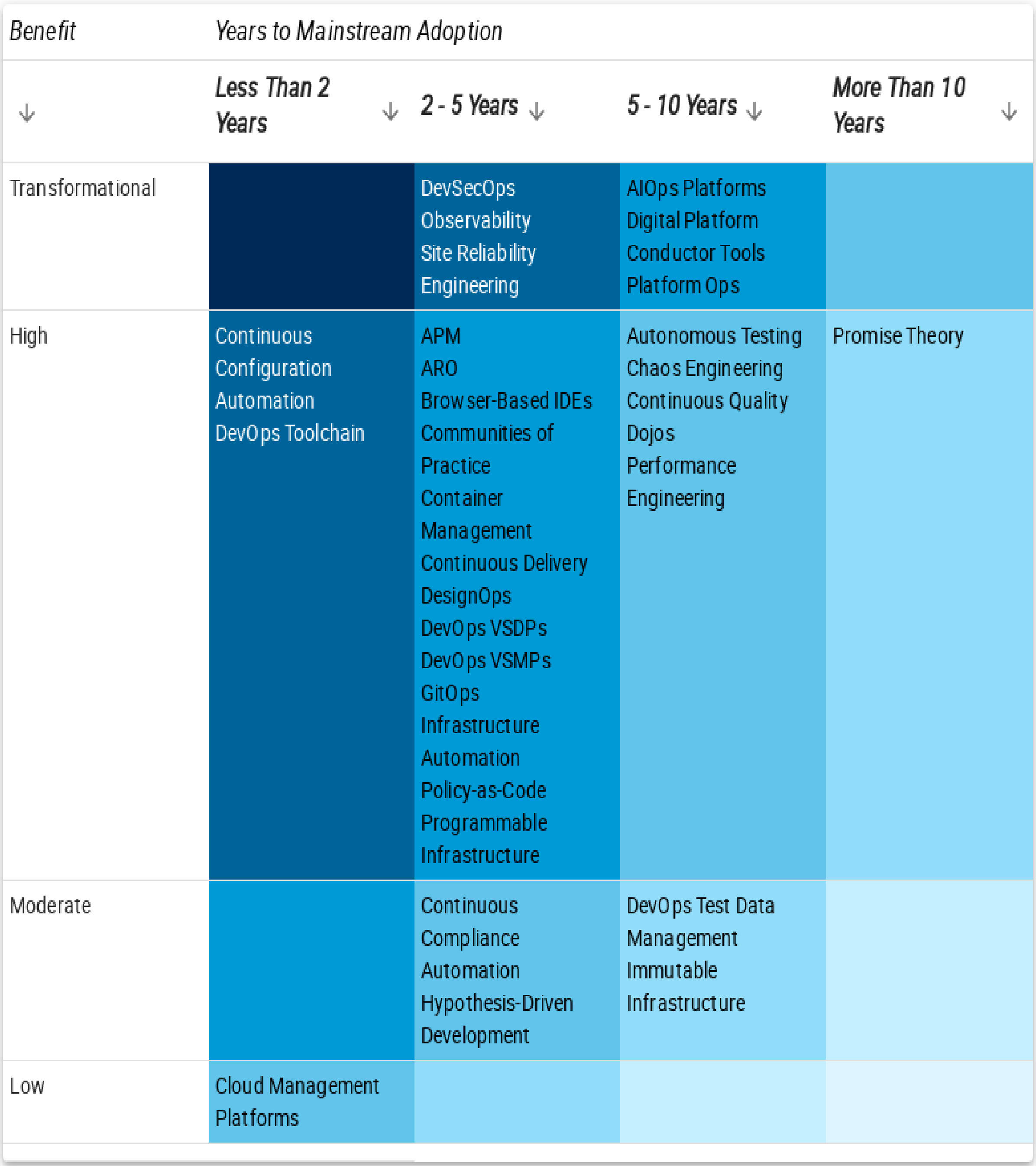
The Priority Matrix maps the time to maturity of technologies and frameworks in an easy-to-read grid format. It answers two high-priority questions:

- 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.

Table 1: Priority Matrix for Agile and DevOps, 2021

(Enlarged table in Appendix)



Benefit	*Years to Mainstream Adoption*			
↓	*Less Than 2 Years* ↓	*2 - 5 Years* ↓	*5 - 10 Years* ↓	*More Than 10 Years* ↓
Transformational		DevSecOps Observability Site Reliability Engineering	AIOps Platforms Digital Platform Conductor Tools Platform Ops	
High	Continuous Configuration Automation DevOps Toolchain	APM ARO Browser-Based IDEs Communities of Practice Container Management Continuous Delivery DesignOps DevOps VSDPs DevOps VSMPs GitOps Infrastructure Automation Policy-as-Code Programmable Infrastructure	Autonomous Testing Chaos Engineering Continuous Quality Dojos Performance Engineering	Promise Theory
Moderate		Continuous Compliance Automation Hypothesis-Driven Development	DevOps Test Data Management Immutable Infrastructure	
Low	Cloud Management Platforms			

Source: Gartner (July 2021)

Off the Hype Cycle

Agile Ops has completed its journey through the Hype Cycle because agile practices have been proven to deliver value.

Bimodal IT Operations was retired last year and Operating Models are used to discuss the evolution of approaches instead.

Please note that for brevity, common acronyms are used in place of the complete name this year (see Acronym Key and Glossary Terms at the end of this report).

On the Rise

Digital Platform Conductor Tools

Analysis By: Roger Williams, David Cappuccio, Dennis Smith

Benefit Rating: Transformational

Market Penetration: Less than 1% of target audience

Maturity: Embryonic

Definition:

Digital platform conductor (DPC) tools coordinate the various infrastructure tools used to plan, implement, operate and monitor underpinning technology and services for applications and digital products. They support digital business, regardless of the environments used or who owns them. DPC tools provide a unified view of underpinning technologies and their connection to applications. This augments strategic decision making and improves the value obtained from technology investments.

Why This Is Important

Traditional and hybrid infrastructure management tools fall short of the requirements of "anywhere operations." Moreover, as infrastructure and operations leaders struggle to manage their portfolio of investments to enable composable business, while optimizing costs and reducing risks, they need help to fill the gaps in visibility, assurance and coordination. These pressures are fueling the rise of DPC tools, which help organizations close these gaps in functionality.

Business Impact

DPC tools address the following gaps in infrastructure management toolsets:

- Providing visualizations of digital platform performance across all life cycle stages — planning, implementing, operating and monitoring.
- Enabling continual optimal performance and placement of workloads across all environments — on-premises, in the cloud or at the edge.
- Ensuring that improvement initiatives show tangible business value across all technology architectures — compute, storage, middleware and network layers.

Drivers

- Difficulty in maintaining an inventory of all technology infrastructure resources and their dependencies, aligned with changes to services, applications, and components and configurations of their promised performance levels.
- Lack of transparency into spending for hybrid digital infrastructure and how resource capacity aligns to actual application workload demand.
- Need to guide where workloads are processed (data center, public cloud, colocation facility, etc.) based on requirements, including capacity, cost and dependency dynamics.
- Challenges with estimating the value, efficiency, quality and compliance delivered by hybrid digital infrastructure based on aggregated data from performance analysis tools and other hybrid digital infrastructure management (HDIM) toolset data feeds.
- Desire for a single point of entry and reporting for digital platform resource requests, and routing them to appropriate HDIM tooling for fulfillment.
- Gaps, duplication and conflicts in data to support application workload migration and business continuity goals, as well as protection of data from accidental deletion or malicious activities.
- Inability to confirm compliance of application workloads and digital platforms to identity requirements and security baselines as part of the organization's cybersecurity mesh approach.
- Poor credibility of business cases for digital platform improvements, including: assessing business impact; measuring gaps between current and desired performance; providing oversight of improvement efforts; and validating benefits delivered.

Obstacles

- Lack of interoperability: Tool sprawl and difficulties in integration will limit DPC tool adoption. The technology landscape is littered with failed approaches that were intended to support data sharing between vendors.
- Lack of data credibility: The desire for a complete, accurate view of all technology as a precondition for decision making has been around for decades, yet is no closer to being realized. Customers that require perfect data before they act, and vendors that design their DPC tools to only work with complete and accurate data, will continue to co-create expectations that will not be met.

- Lack of budget: DPC tools may be viewed as “overhead” that does not have a compelling business case. No one likes paying for something that does not address specific pain points felt today.
- Lack of vendor commitment: Many vendors will be tempted to “DPC wash” their existing offerings and claim that these capabilities are already addressed or can be added for very little cost.

User Recommendations

- Build a DPC tooling strategy that supports digital business ambitions by defining the management elements, environments and technology layers required to meet the organization’s infrastructure needs now and in the future.
- Address measurement and coordination gaps by working with key stakeholders to identify infrastructure value, risk and cost objectives, and making targeted investments in integration, dependency mapping and continuous improvement capabilities.
- Plan for DPC tooling investments by determining which DPC capability aspects are needed in the short, medium and long term. Compare these capabilities to current and future vendor offerings for infrastructure management tooling that can provide initial DPC tool functionality.
- Ensure that DPC tooling investments can deliver sustained value by requiring that DPC tool marketers show how the tool will address current organizational pain points and how it will adapt to future needs as organizational requirements evolve.

Sample Vendors

Amazon Web Services (AWS); Cloudsoft; Flexera; LeanIX; Microsoft; OpsRamp; ServiceNow; Snow Software

Gartner Recommended Reading

Innovation Insight for Digital Platform Conductor Tools

Rethink Your Infrastructure Management Tool Selection Strategy

Browser-Based IDEs

Analysis By: Manjunath Bhat

Benefit Rating: High

Market Penetration: 1% to 5% of target audience

Maturity: Emerging

Definition:

Browser-based integrated development environments (IDEs) are consumed "as a service." They enable browser-based remote access to a complete development environment, which obviates the need for local installation/configuration. This decouples the development workspace from the physical workstation, enabling a consistent experience across devices. The workspace comprises code editors, debuggers, code review, collaboration tools, source control repositories and integrated CI/CD pipelines.

Why This Is Important

Browser-based IDEs provide consistent, secure access to preconfigured development workflows to developers. This frees them from setting up their own environments, eliminating the need to install and maintain prerequisites, software development kits (SDKs), security updates and workstation plug-ins. Browser-based IDEs are prepackaged with language tooling for multiple programming languages, enabling teams to write code for different application stacks with standardized and templatized workflows.

Business Impact

Browser-based IDEs are becoming popular, due to their native integration with Git-based repositories and continuous integration/continuous delivery (CI/CD) tools, accelerating DevOps adoption. Browser-based IDEs enable IT to centrally manage and secure access to development environments, even from personal devices, minimizing code exfiltration from user machines. Browser-based IDEs are important to self-service platforms that enhance developer productivity, ensuring consistency and governance.

Drivers

Gartner predicts that, by 2026, 60% of cloud workloads will be built and deployed using browser-based IDEs. Five factors are driving their increased adoption:

1. Remote work and remote onboarding of software developers create a need for a frictionless onboarding experience. The ability to share the development environment among team members makes remote debugging and pair programming easier.

2. Faster time to market requires consistency in development workflows and reduced toil to bootstrap environments. Environment setup issues can impede productivity and hurt the onboarding experience.

3. Cloud-native (e.g., Kubernetes) deployments require new tooling that either isn't available or is inconvenient to set up on-premises. Likewise, low-code development tools are web-first — 89% of low-code development vendors support a web-based IDE, compared with 24% for desktop-based IDEs.

4. The ability to centrally manage, govern and secure development environments becomes especially important with the increasing threat of software supply chain attacks. In addition, browser-based IDEs make it easier to support and secure bring your own PC (BYOPC) use cases.

5. Automating DevOps workflows introduces more plug-ins, extensions and API integrations, which make it cumbersome to manage on local machines.

Browser-based IDE providers are innovating rapidly. Representative providers include Amazon Web Services Cloud9, Codeanywhere, GitHub Codespaces, Gitpod, Red Hat CodeReady Workspaces and Replit.

Obstacles

Browser-based IDEs incur costs in addition to what an organization may already be paying for DevOps tooling. The cost can be prohibitive for teams that rely only on open-source tools for application development and delivery needs on local machines. The pay-per-use subscription costs add up when you have a large team, and developers may leave the IDE open with misconfigured settings for autohibernation or timeout.

Connectivity presents another obstacle. Poor or inconsistent internet speeds adversely affect developer productivity. Developers cannot write, debug and test code without a stable internet connection.

Security and compliance policies can prevent the use of cloud for development needs in some organizations. This can rule out the use of browser-based IDEs. Browser-based IDEs often depend on specific source control repositories or a specific Kubernetes implementation. This requires organizations to change their development practices before adopting browser-based IDEs.

User Recommendations

Software engineering leaders leading product and platform engineering teams should:

- Pilot the use of browser-based IDEs when their teams are developing with cloud-hosted source repositories. Even when developers prefer local environments on their machines, browser-based IDEs can provide a baseline as a "reference" environment.
- Ensure that browser-based IDEs are part of an overall developer self-service platform with central governance. The benefit of centralized governance and developer agility can be a win-win for platform and product teams.
- Use browser-based IDEs as one of the "quick wins" to improve the onboarding experience for new developers and reduce ramp-up time.
- Work with security leaders and enforce strong authentication and authorization policies to mitigate security risks. Browser-based IDEs present an additional, high-value attack vector, as they become the pipeline through which intellectual property flows.

Sample Vendors

Amazon Web Services; Codeanywhere; GitHub; Gitpod; Red Hat; Replit

Gartner Recommended Reading

Quick Answer: How to Create a Frictionless Onboarding Experience for Software Engineers

Expert Insight Video: Improve Remote Worker Productivity by Measuring Performance Fairly

Accelerating Agile and DevOps Adoption Post-COVID-19

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

GitOps

Analysis By: Paul Delory, Arun Chandrasekaran

Benefit Rating: High

Market Penetration: Less than 1% of target audience

Maturity: Embryonic

Definition:

GitOps is a technique for operating a cloud-native application using only declarative constructs stored in Git. It is the latest name for extending CI/CD to software deployment and infrastructure management. Beyond that, GitOps remains ill-defined and the subject of many debates. The CNCF's GitOps Working Group is formulating what it hopes will become the industry standard. The ideas behind GitOps are not new, but emerging tools and practices now promise to make this longtime dream a reality.

Why This Is Important

GitOps would be transformative for the I&O organization. GitOps workflows convert a Git pull request into a verified container image ready for production. Infrastructure is code. All changes flow through Git, where they are version-controlled, immutable, auditable and reversible. Developers interact only with Git, using abstract, declarative logic. I&O uses it to deploy infrastructure. It extends a common control plane across K8s clusters, which is increasingly important as clusters proliferate.

Business Impact

- **Agility and speed:** GitOps can be used to introduce version control, automation, collaboration and compliance. Artifacts are consistent and reusable. All of this translates directly to business agility and faster time to market.
- **Transparency and visibility:** GitOps artifacts are centralized and version-controlled, making them easy to verify and audit.
- **Resilience:** By implementing and centralizing infrastructure as code, GitOps enhances availability and resilience of services.

Drivers

- **Kubernetes adoption and maturity:** GitOps must be underpinned by an ecosystem of technologies, including tools for automation, infrastructure as code, CI/CD, observability and compliance. Kubernetes has emerged as a universal common substrate for cloud-native applications. This provides a ready-made foundation for GitOps. As Kubernetes adoption grows within the enterprise, so, too, can GitOps.

- **Need for increased speed and agility:** Speed and agility of software delivery is a critical metric that CIOs care about. As a result, IT organizations are pursuing tighter coupling of I&O and development teams to drive shorter development cycles, faster delivery and increased deployment frequency; enable organizations to respond immediately to market changes; handle workload failures better; and tap into new market opportunities. GitOps is the latest way to drive this type of cross-team collaboration.

- **Need for increased reliability:** Speed without reliability is useless. The key to increased software quality is effective governance, accountability, collaboration and automation. GitOps can enable this with process transparency and workflow commonality across development and I&O teams. Automated change management helps to avoid costly human errors that can result in poor software quality and downtime.

- **Talent retention:** Organizations adopting GitOps have an opportunity to upskill existing staff for more automation- and code-oriented I&O roles. This opens up opportunities for staff to learn new skills and technologies, resulting in better employee satisfaction and retention.

- **Cultural change:** By breaking down organizational silos, development and operations leaders can build cross-functional knowledge and collaboration skills across their teams to enable them to work effectively across boundaries.

- **Cost reduction:** Automation of infrastructure eliminates manual tasks and rework, improving productivity and reducing downtimes, both of which can contribute to cost reduction.

Obstacles

- **Prerequisites:** GitOps is only for cloud-native applications. The Working Group's draft architecture expressly requires Kubernetes and implicitly assumes the presence of a host of other technologies built on top of K8s. GitOps is necessarily limited in scope.
- **Lack of consensus:** For now, GitOps means different things to different people. This situation is likely to persist unless the GitOps Working Group succeeds in promulgating a definition that gains industrywide acceptance.
- **Cultural change:** GitOps is a cultural change that organizations need to invest in. IT leaders need to embrace process change. This requires discipline and commitment from all participants to doing things in a new way. Technology is not magic.
- **Skills gaps:** GitOps requires automation and software development skills, which many I&O teams lack.
- **Organizational inertia:** GitOps requires collaboration among different teams. This requires trust among these teams for GitOps to be successful.

User Recommendations

- **Use GitOps for cloud-native workloads:** Your initial target for GitOps must be a containerized, cloud-native application that is already using both Kubernetes and a continuous delivery platform such as Flux.
- **Embed security into GitOps workflows:** GitOps practices can and should embed security into the workflow. Security teams need to "shift left" so that the organization can build holistic CI/CD pipelines that impact software delivery and infrastructure configuration, with security embedded in every layer of that workflow.
- **Be wary of vendors trying to sell you GitOps:** GitOps isn't a product you can buy; it is a workflow and a mindset shift that needs to be part of your overall DevOps culture.
- **Use GitOps to sell automation initiatives:** I&O leaders can use the GitOps buzzword to get buy-in for adopting general automation best practices that the organization should already be doing. In the absence of a standard, you can define what GitOps means to your organization, then deploy it iteratively.

Sample Vendors

CloudBees; GitLab; Harness; Red Hat; Weaveworks

Policy-as-Code

Analysis By: Paul Delory

Benefit Rating: High

Market Penetration: 1% to 5% of target audience

Maturity: Emerging

Definition:

Policy-as-code languages express governance and compliance rules as code, so they can be enforced programmatically by automation tools. PaC languages are often domain-specific and declarative. With PaC, policies are treated as software, making them subject to version control, code review and functional testing. The most mature PaC tools can render any business logic in code. You can use them to enforce architectural standards, corporate policies, regulatory requirements, budgets and more.

Why This Is Important

In the most mature automation pipelines, I&O engineers mostly spend time on optimization, governance and compliance. They no longer build infrastructure; that work has been automated and turned over to end users. Now, I&O builds the guardrails around the infrastructure services that their end-users consume. I&O must align with security and compliance teams. PaC brings policy enforcement into their automation pipelines, while preserving a separation of duties that mirrors a typical IT org chart.

Business Impact

- **Security, compliance and automation:** PaC combined with infrastructure automation provides direct enforcement of policies with implicit compliance guarantees.
- **Alignment of security and Ops teams:** PaC allows security and compliance teams to interface directly with automation pipelines to ensure conformance.
- **Visibility and auditability:** PaC provides both unambiguous documentation of policies and evidence they are being enforced.
- **Less toil:** PaC reduces the overhead of creating and enforcing policies.

Drivers

- **New PaC tools:** Several dedicated PaC tools are now on the market, many of them open source. Most prominent is the Open Policy Agent, a Cloud Native Computing Foundation project. But other options abound, including: InSpec (part of the Chef suite), Sentinel (part of the HashiCorp suite) and Bridgecrew (recently acquired by Palo Alto). All of the major hyperscale public clouds also have their own native policy engines.

- **Increasing regulation:** The advent of new regulations such as GDPR has increased both the difficulty of compliance and the pressure on compliance teams. PaC allows compliance teams and auditors to document their policies in detail, and to verify they are being enforced.

- **Security breaches:** Similarly, a spate of newsworthy security breaches at public companies have put every IT organization's security and compliance practices under increased scrutiny. No I&O team wants their security failures to be the reason their company ended up in the headlines.

- **Growth of DevOps and DevSecOps:** More and more companies are embracing DevOps and DevSecOps — which means more and more companies are encountering the hard governance problems of automation, which PaC can help solve.

- **Cloud optimization and cost control:** Beside their benefits for security and compliance, PaC tools can also be used to enforce the I&O department's build standards for infrastructure — including enforcing budgets. In the public cloud, where oversized or unnecessary infrastructure incurs direct out-of-pocket costs, programmatically enforced policies can help to control spending.

Obstacles

- **Immaturity of tools:** PaC tools are not yet fully ripe. They will not gain real traction until they have an extensive library of community-generated content. Ideally, users would simply download the policies they need from a free, public repository — rather than having to write their own policies. Over time, as the user base expands and commercial offerings see increased adoption, PaC tools will reach a critical mass of downloadable content that enables real-world use cases.

- **Skill set:** Many I&O professionals lack the skills to operate automation and PaC tools effectively. Gartner clients routinely report that their automation and policy management are hindered primarily by people, not tools. PaC will magnify these existing skills challenges.

- **Organizational inertia:** PaC promises improved collaboration between I&O and security/compliance teams. But in some organizations, this change would be unwanted. Internal resistance of this kind will slow the rate, scope and scale of PaC initiatives.

User Recommendations

- **Start small:** Choose a pilot use case where PaC is likely to provide real business benefits. Once PaC has proven its value, expand to other use cases over time.

- **Prioritize existing templates:** Focus your PaC efforts on use cases that have ready-made implementation templates — ideally, publicly-available downloadable content. For example, almost every PaC tool on the market has a canned implementation of the CIS benchmarks.

- **Break down team silos:** Use PaC to build a common workflow for automation and policy enforcement that spans I&O, security and compliance teams.

- **Integrate PaC into automation pipelines:** To automate a process, you must know *what* to do before you can determine *how* to do it. PaC creates specific and enforceable guidelines for automation tools to follow, solving this problem.

- **Measure before and after:** Use observability tools and value stream mapping to define your starting state, then compare it to the end state. Collect real data to quantify the value of PaC.

Gartner Recommended Reading

Using Cloud-Native 'Policy as Code' to Secure Deployments at Scale

Innovation Insight for Continuous Compliance Automation

Market Guide for DevOps Value Stream Delivery Platforms

2021 Planning Guide for Security and Risk Management

2021 Planning Guide for IT Operations and Cloud Management

DesignOps

Analysis By: Brent Stewart

Benefit Rating: High

Market Penetration: 5% to 20% of target audience

Maturity: Adolescent

Definition:

DesignOps is a set of operational practices that enables design team management and product-level delivery of design assets. The team management side focuses on strategic alignment to the business, operations for the central design function and career development. The product delivery side 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.

Why This Is Important

DesignOps introduces formalized approaches to governance, operations and people management. As a set of easy-to-use operational standards, DesignOps continues to gain in popularity as digital product companies (for example, 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.

Business Impact

The growth of DesignOps is due, primarily, to the value it creates during the delivery of design assets. 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. DesignOps represents the first widespread implementation of operational methods and techniques created not only for designers, but also for developers.

Drivers

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. It also allows them to progressively support early development activities such as tech design and story creation.

There are three key drivers behind DesignOps:

- **Innovation:** When coupled with DevOps, DesignOps leads to more innovative solutions. As a practice, DesignOps employs dual-track agile that 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.
- **Speed:** 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.
- **Collaboration:** 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 crucially important groups.

Obstacles

To a large extent, the growth of DesignOps is inhibited by key gaps in planning, estimation and tracking knowledge:

- Few UX practitioners are educated in detailed planning and estimation using a common work breakdown structure (WBS).
- Few product managers are trained in UX planning, estimating and tracking.

- Popular enterprise agile planning (EAP) tools are not designed with UX practitioners, activities and deliverables in mind (though this is changing).

User Recommendations

Software engineering leaders should:

- Educate themselves about the practice of DesignOps
- Train their UX teams in the basics of agile
- Pilot the approach with a high-performing, multidisciplinary feature team

Following a successful pilot, application leaders and the pilot team members should

- Engage in a productwide rollout that involves training, updated product plans and the allocation of one or more persons 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.

Gartner Recommended Reading

DesignOps: Organize, Collaborate and Innovate Product UX at Speed

Technology Insight for Digital Product Design Platforms

3 Key Practices to Enable Your Multiexperience Development Strategy

Software Engineering Technologies Primer for 2021

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

Strategic Roadmap for Becoming a Digital Product Delivery Organization

Promise Theory

Analysis By: Roger Williams

Benefit Rating: High

Market Penetration: Less than 1% of target audience

Maturity: Embryonic

Definition:

Promise theory is an approach to modeling the interactions among entities to understand uncertainty, improve coordination and solve problems. Agents (entities in a system that can independently change) make explicit, voluntary promises about unverified behaviors, which enables them to examine a system and assess the likelihood of a desired outcome. Examples include a promise that work will finish by a certain time, or a system's response time for requests will be less than 50 ms.

Why This Is Important

Promise theory can provide a common language for coordination among various entities, due to the bottom-up nature of its approach. Promise theory provides infrastructure and operations (I&O) leaders with a scalable approach to enforce consistent, declarative and repeatable methods for deploying and managing systems.

Business Impact

Promise theory's open structure, simple concepts and acknowledgment of the uncertainty inherent in promises can enhance trust in digital business. Given the importance of building trust to succeed with agile and DevOps, promise theory can be a catalyst for these efforts. Promises can improve performance management and coordination among product team subject matter experts. The result is nimble, resilient, decentralized and distributed systems.

Drivers

The use of promises differs from a command-and-control approach to uncertainty, which is to require obligations, and to use punishments and rewards to enforce them. Command and control often fails, because it can't guarantee results (only consequences), and it is prone to passive resistance that results in actions, rather than desired outcomes. For instance, for reporting purposes, we may prohibit an incident record from being saved unless a category is selected. However, the report will be worthless if it leads to the default or first available option being selected for convenience, rather than the correct value. Similarly, we may insist on 99.999% uptime from a server, yet that, in and of itself, will not make it so.

Allowing entities to interact through a public promise offers multiple benefits:

- A declarative model (as opposed to a procedural model)
- Idempotency — repeatable actions that are safe to execute any number of times without causing a change in the state of the system
- Autonomy and scale — e.g., agents that manage the state of the system locally through delegation of control, instead of command and control

Obstacles

Promise theory has faced many hurdles in broader adoption. For example:

- Lack of awareness of the concept
- Rejection due to "not invented here" syndrome and/or an unwillingness to accept the uncertainty required for digital business and DevOps success
- An incorrect perception that the topic is merely theoretical and "ivory tower," rather than practical
- Push-back from experts on configuration management in various domains who are unwilling or unable to understand why new terminology is needed, rather than other domains simply adopting their approach to configuration management
- "Promise washing" of work already done, without recasting in terms of business outcomes
- Reluctance to set clear targets and provide transparency into performance versus those targets
- Conflict with other behaviors that undermine trust, such as command-and-control management techniques

User Recommendations

Product and DevOps teams looking 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
- An assessment of the level of commitment and intensity
- What benefit of value other agents can expect to obtain when the promise is kept

Once an initial set of promises is 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.

Sample Vendors

Apache Software Foundation; Cisco; Cloudsoft; Northern.tech (CFEngine)

Gartner Recommended Reading

Innovation Insight for Promise Theory

Don't Let the Cloud Ruin Your CMDB

Innovation Insight for Digital Platform Conductor Tools

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

Benefit Rating: High

Market Penetration: 5% to 20% of target audience

Maturity: Adolescent

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 affect application performance and encompasses the practices that help mitigate performance risks before progressing to subsequent phases.

Why This Is Important

The ability to consistently deliver products that satisfy end-user expectations of scalability, stability, 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.

Business Impact

Performance engineering includes both “shift left” and “shift right” testing practices and 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.

Drivers

- Raised end-user expectations for application quality, specifically nonfunctional characteristics, such as performance efficiency.
- The need to ensure business continuity under changing usage patterns, network topologies and data volumes.
- The need to optimize the use of modern hyperdynamic microservices-based architectures, multicloud deployments, and automation and integration capabilities of modern application platforms.
- Support for different migration scenarios, such as lift and shift, replatforming or refactoring the architecture of packaged or on-premises apps.
- The need to manage performance and scalability across different cloud providers, such as AWS, Google or Azure, to ensure the ability to shift from one operator to another without a change in user experience.
- Cost optimization for SaaS/PaaS services that makes use of dynamic infrastructure to spin up (and down) testing resources as needed.

Obstacles

- **Focusing only on tools and technology:** Performance engineering requires a change in organizational culture. Tools enable quality but won't solve problems on their own.
- **Organizational inertia:** Departmental silos, traditional top-down management structures and a lack of experience with managing quality continuously can impede adoption.
- **Lack of clear goals:** Successful performance engineering requires clear goals that are aligned with the priorities of the business.
- **Internal pushback:** Performance engineering requires engaging stakeholders throughout 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 across all team members.
- **Limitation to performance testing only:** Performance engineering includes designing with performance in mind, building the product with clear performance objectives and facilitating the discovery of issues early in development.

User Recommendations

- Create awareness for nonfunctional characteristics such as performance efficiency. ISO/IEC 25010 provides a template for understanding quality characteristics and includes performance efficiency as one of the top-level nonfunctional domains.
- Foster a proactive performance quality strategy that makes performance an explicit requirement and verifies that performance goals are met and user satisfaction meets expectations.
- Allocate ownership and appoint staff with the required skills needed for performance engineering by identifying the required roles, technologies and practices.
- Establish relevant performance quality metrics based on the joint objectives that the business and IT are trying to accomplish.
- Collaborate with I&O leaders and establish a feedback loop by leveraging performance information from production and real users.

Sample Vendors

AppDynamics; Broadcom; Dynatrace; Eggplant; Micro Focus; Tricentis

Gartner Recommended Reading

Adopt a Performance Engineering Approach for DevOps

Innovation Insight for Continuous Quality

Improve Software Quality by Building Digital Immunity

Innovation Insight for Autonomous Testing

DevOps VSMPs

Analysis By: Hassan Ennaciri, Akis Sklavounakis

Benefit Rating: High

Market Penetration: 5% to 20% of target audience

Maturity: Early mainstream

Definition:

DevOps value stream management platforms (VSMPs) enable organizations to manage their software delivery process as a value stream to maximize the delivery of 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 flow, surface constraints and non-value-added work, to improve customer value delivery.

Why This Is Important

As organizations scale their agile and DevOps practices, higher-level metrics that assess performance and efficiency of their product delivery is essential. DevOps VSMPs integrate with multiple data sources to provide DevOps-related telemetry. These insights enable stakeholders to make data-driven decisions in an agile manner and to correct course as needed. The visualization capabilities of DevOps VSMPs help product teams analyze customer value metrics against the cost required to deliver that value.

Business Impact

DevOps VSMPs help organizations bridge the gap between business and IT by enabling 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. These platforms also provide product teams with end-to-end visibility and insight into the flow of work to help them address constraints and improve delivery.

Drivers

- Scaling agile and DevOps initiatives.
- Need for visibility into business, development and operational metrics.
- Optimization of delivery flow by mapping end-to-end processes involved in software delivery to reduce waste and bottlenecks due to cross-team dependencies.
- Need to improve quality and velocity of product deployments.
- Visibility to security and compliance status of products.
- Need for orchestration capabilities to have better traceability.

Obstacles

- DevOps VSMPs do not include native continuous integration/continuous delivery (CI/CD) capabilities; execution of the delivery pipeline requires use of a custom toolchain or a value stream delivery platform (VSDP).
- VSMPs may lack native plugins for some existing tools and that will require custom integration.
- Rationalization of acquiring another tool with more dashboards acts as an obstacle.
- VSMPs are still evolving and not all vendors have all the critical capabilities.

User Recommendations

- Ensure all stakeholders in the value streams are involved in the selection of a VSMP: business leaders, owners, application leaders and I&O leaders.
- Examine your existing EAP tools or VSDPs as some vendors are expanding their capabilities to include VSMP functionality.
- Pilot the VSMP with a product that has mature DevOps practices to best evaluate what insight can be generated.

Sample Vendors

CloudBees; Digital.ai; Opsera; Plandek; Plutora; Tasktop

Gartner 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

Predicts 2021: Value Streams Will Define the Future of DevOps

Infographic: Top 10 Technology Trends Impacting DevOps

Chaos Engineering

Analysis By: Jim Scheibmeir, Dennis Smith

Benefit Rating: High

Market Penetration: 5% to 20% of target audience

Maturity: Adolescent

Definition:

Chaos engineering is the use of experimental and potentially destructive failure testing 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 before and after implementation. Chaos engineering is often utilized by site reliability engineering teams to proactively prove and improve resilience during fault conditions.

Why This Is Important

Many organization's stake success on test plans that overemphasize software functionality and underemphasize the importance of validating the system's reliability. Chaos engineering (CE) moves the focus of testing a system from the "happy path" — instead, testing how the system can gracefully degrade or even continue to be useful while under various levels of impact. CE can also help identify where product documentation is less than sufficient or understanding of a system is lacking or siloed.

Business Impact

With chaos engineering, we minimize time to recovery and change failure rate, while maximizing uptime and availability. Addressing these elements helps improve customer experience, customer satisfaction, customer retention and new customer acquisition.

Drivers

- Gartner's 2020 Achieving Business Agility with Automation, Continuous Quality and DevOps Survey found that 18% of respondents were currently using or planning to use chaos engineering to improve software quality.

- Complexity in systems and increasing customer expectations are the two largest drivers of this practice and the associated tools. As systems become more rich in features, they also become more complex in their composition, and more critical to business success.

- Overall, chaos engineering helps organizations create more reliable systems and prove that systems are reliable.

Obstacles

- The first obstacle to chaos engineering is perception. Within many organizations, the predominant view is that the practice is random, done first in production, and due to these leads to more risk than less.
- Another obstacle to chaos engineering is organizational culture and attitude toward quality and testing. When quality and testing are only viewed as overhead, then there will be a focus on feature development over application reliability.
- Another common obstacle is simply gaining the time and budget to invest in learning the practice and associated technologies. Organizations must reach minimum levels of expertise where value is then returned.

User Recommendations

- Utilize a test-first approach by practicing chaos engineering in preproduction environments. Then move findings into production environments.
- Incorporate chaos engineering into your system development or testing process.
- Build-out incident response protocols and procedures, as well as monitoring, alerting, and observability capabilities in tandem with advancement of the 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.
- Investigate opportunities to use chaos engineering in production to facilitate learning and improvement at scale as the practice matures. However, be warned that Gartner believes that there are still very few organizations purposely using chaos engineering in their production environments.
- Formalize the practice by adopting a platform or tool to track the activities and create metrics to build feedback for continuous improvements.

Sample Vendors

Alibaba Cloud; Amazon Web Services; ChaosIQ; Gremlin; steadybit; Verica

Gartner Recommended Reading

Improve Software Quality by Building Digital Immunity

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

DevOps Test Data Management

Analysis By: Dale Gardner

Benefit Rating: Moderate

Market Penetration: 20% to 50% of target audience

Maturity: Early mainstream

Definition:

DevOps test data management is the process of providing DevOps teams with data to evaluate the performance, functionality and security of applications. It typically includes copying production data, anonymization or masking and virtualization. In some use cases, specialized techniques, such as synthetic data generation, are appropriate. Given potential compliance and privacy issues, the process more frequently involves members of application and data security teams.

Why This Is Important

Test data management is inconsistently adopted across organizations, with many teams copying production data for use in test environments. As organizations shift to DevOps and the pace of development increases, this traditional approach is increasingly at odds with requirements for efficiency, privacy and security, and even the increased complexity of modern applications. This opens organizations to a variety of legal, security and operational risks.

Business Impact

Quick provisioning of test data helps ensure the pace of development isn't slowed. It's also increasingly important to remain compliant with the growing number of privacy mandates to which organizations are subject. This helps avoid fines and remediation and mitigation costs, along with the inevitable delays associated with audits and investigations. Finally, by providing application teams with anonymized or synthetic data, the risk of data breaches is reduced.

Drivers

- Test data management is generally viewed as a mature, relatively uncomplicated, technology. However, the reality is the combination of the increased pace of development from DevOps and a growing number of privacy mandates and constraints have strained traditional approaches — yielding new approaches and technology.

- More traditional test data management has been inconsistently adopted, with many organizations either simply using copies of production data or generating "dummy data" (distinct from emerging synthetic data generation techniques). The data privacy requirements and complexity issues noted have prompted organizations to revisit and update their processes with an eye toward scalability and automation. Updated technologies may also be a requirement. For example, requirements for speed and agility have created a need for data virtualization tools.

- Data protection is cited by a growing number of clients in inquiries regarding test data management. Privacy and data protection requirements mean it's no longer safe to simply provide development teams with a copy of production data. The practice leaves organizations open to increased risk of regulatory violations, data breaches and other security issues.

- With modern applications relying on an increasing number of interconnected data stores (many of which themselves are technologically vastly more complex), applications, and APIs to function, testing has become more complex. Such complexity demands tools support the ability to coordinate and synchronize changes across different data stores to ensure relational consistency while still addressing security and speed mandates.

Obstacles

- In the absence of a strong culture of security, processes and technologies to protect sensitive information during development and testing will encounter friction. Conflicting needs for rapid development and privacy require attention to a mix of organizational and cultural issues to strike a balance across groups.
- Responsibility for test data management in organizations has been shared by application development and database administration. New technologies and processes may shift those responsibilities to include security, complicating organizational dynamics and potentially creating the need for additional staffing.
- Implementation can be a burden, especially where little or no data sensitivity classification has been done. This must be accomplished before teams can proceed with required data transformation and masking. These efforts are typically combined with an analysis of data relationships, so that relational integrity can be assured.

User Recommendations

- Involve stakeholders — application development and test teams (to understand consumption patterns and needs), data management, privacy and security teams, compliance teams, other security teams, and legal counsel — as appropriate.
- Document existing test data management practices so tools and processes can be evaluated against data protection mandates.
- Coordinate to avoid duplication of effort and tooling since data masking tools may also be used by analytics teams (e.g., to provide data for machine learning or other purposes).
- Evaluate data masking tooling by considering support for databases and other stores, data discovery capabilities, types of masking supported and the ability to coordinate change to ensure consistency (e.g., key fields) across multiple sources.
- For DevOps use cases where frequent updates to test data are required, data virtualization may also be useful. Virtualization can speed the process of providing copies of safe data.

Sample Vendors

Actifio; BMC Compuware; Broadcom (CA Technologies); Delphix; Hazy; Informatica; MENTIS; Micro Focus; Solix Technologies

Gartner Recommended Reading

Market Guide for Data Masking

Elevating Test Data Management for DevOps

Continuous Quality

Analysis By: Joachim Herschmann, Jim Scheibmeir

Benefit Rating: High

Market Penetration: 5% to 20% of target audience

Maturity: Adolescent

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.

Why This Is Important

Many DevOps organizations are 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.

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.

Drivers

- Raised end-user expectations for application quality, which require a shift to a more holistic view of what constitutes superior quality that delights users.
- The pressure to innovate rapidly in order to launch differentiated products in the market quickly without compromising on quality.
- The ability to consistently deliver business value with consistently high quality to mature DevOps processes.
- The need to ensure that teams are equipped to create a superior user experience, build features that fit the market's timing, and enable the characteristics of an application that deliver value faster than they create technical debt.

Obstacles

- Lack of clear goals: Successful continuous quality requires clear goals that are aligned with the priorities of the business.
- Internal pushback: Continuous quality requires engaging stakeholders across the organization and empowering them to be more accountable. Such a holistic approach can be seen as restrictive and requires consensus on usage across all team members.
- Loss of productivity: Changing organizational culture and engaging in new practices require significant investment and time. This will impact current timelines and can cause a decrease in productivity prior to reaching steady productivity.
- Limitation to testing only: Continuous quality includes designing a product with quality in mind, building it with clear quality objectives and facilitating the discovery of issues early in development.
- Focusing only on tools: Continuous quality requires a change in organizational culture. Tools are enablers of quality but tools on their own won't solve problems.

User Recommendations

- Move away from the traditional application- or project-centric model of quality to a holistic quality approach by adopting an ecosystem-centric view of quality and a focus on business outcomes.
- Accelerate product delivery by championing a continuous quality mindset and 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 user experience (UX) designers and customer experience (CX) teams to infuse quality right from the inception of an idea.
- Establish relevant quality metrics based on the joint objectives that the business and IT are trying to accomplish.
- Task teams with developing continuous quality practices before choosing tools.

Gartner Recommended Reading

Innovation Insight for Continuous Quality

Adopt a Performance Engineering Approach for DevOps

Improve Software Quality by Building Digital Immunity

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

Innovation Insight for Autonomous Testing

Continuous Compliance Automation

Analysis By: Daniel Betts, Hassan Ennaciri

Benefit Rating: Moderate

Market Penetration: 5% to 20% of target audience

Maturity: Adolescent

Definition:

Continuous compliance automation (CCA) integrates compliance and security policy enforcement into DevOps delivery pipelines. 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 auditability.

Why This Is Important

Continuing DevOps initiative success drives enterprise investments in compliance automation. CCA improves release velocity and reliability while simplifying compliance enforcement via policy-driven, automated controls, improving 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, expensive remediation work.

Business Impact

Organizations evolving DevOps practices can minimize risks and penalties by embedding automated compliance into their delivery pipelines. CCA enables organizations to integrate compliance into all phases of the delivery pipeline and consistently enforces compliance policies without sacrificing operational agility.

Drivers

- Organizations are facing an increasing number of regulatory obligations and more stringent enforcement, so automating compliance will become even more valuable to I&O leaders as they strive to maximize flow through their DevOps value streams by: needing to scale to meet additional compliance requirements with limited delay; demonstrating compliance through automated testing; reducing the risk of compliance audit failures; and reducing the time spent in compliance steps and unexpected remediation work.

- As product teams adopt cloud-native application architectures and development models, there is a need to integrate compliance into the DevOps toolchain that supports those applications. For example, because containers are fundamentally immutable, the need to scan container images upfront requires specialized container-scanning tools for vulnerabilities. Comprehensive container security starts in development with an assessment of the contents of the container, secrets management and should extend into production with runtime container threat protection and access control.

Obstacles

- Most CCA tools only target one development or delivery activity. No vendor provides capabilities across all elements of the delivery value stream. DevOps teams must integrate multiple tools to provide compliance coverage across development and delivery activities.
- Failure to engage with compliance and security SMEs early in the development life cycle can lead to problems. Early input from compliance and security SMEs will help I&O leaders account for security and compliance requirements and audit failures.
- The lack of rule set understanding and consistent implementation can be an impediment to CCA. While it is important to leverage the acceleration that vendor rule sets can provide, it is vital that they are understood by organizational compliance teams and implemented consistently to provide maximum value.
- Poorly implemented CCA presents a business risk. If it is assumed that by implementing CCA delivered software becomes compliant without additional effort, organizations will face increased risk of compliance failure.

User Recommendations

- Adhere to compliance, governance and security requirements while creating a leaner operating environment. CCA tools enable DevOps teams to achieve both goals: improving value stream delivery and mitigating risks.
- Implement a shift-left approach to ensure compliance controls are understood earlier in the development process. Implement automated compliance checks at every phase of the pipeline, demonstrating a “shift-secure” approach.
- Invest in tools that enable CCA at scale and can provide a continuous approach to prevent, detect and correct audit failures.
- Enforce security and compliance across all domains, including databases, application code, infrastructure and open-source software. Since there is no single vendor tool that covers all those domains, DevOps teams must use multiple tools and integrate across all phases of the delivery pipeline.

Sample Vendors

Anitian; JFrog; Rapid7; Redgate; Snyk; Sonatype; Styra; WhiteSource

Gartner Recommended Reading

Innovation Insight for Continuous Compliance Automation

Market Guide for Compliance Automation Tools in DevOps

3 Steps to Ensure Compliance and Audit Success With DevOps

3 Steps to Integrate Security Into DevOps

How to Build and Evolve Your DevOps Toolchains

DevOps VSDPs

Analysis By: Manjunath Bhat

Benefit Rating: High

Market Penetration: 5% to 20% of target audience

Maturity: Early mainstream

Definition:

DevOps value stream delivery platforms (VSDPs) provide fully integrated capabilities that enable continuous delivery of software. These capabilities include planning, version control, continuous integration, test automation, release orchestration, continuous deployment (and rollback), monitoring, security testing and analyzing value stream metrics. DevOps VSDPs integrate with infrastructure and compliance automation tools to automate infrastructure deployment and policy enforcement.

Why This Is Important

Value stream delivery platforms enable organizations to simplify building and managing DevOps pipelines. They reduce complexity involved in orchestrating, integrating and governing pipeline activities. In addition, the prebuilt integration between different components of the platform leads to improved visibility, traceability and observability into the complete application development value stream. The benefits extend beyond IT.

Business Impact

VSDPs constitute the pipeline for continuous delivery of digital value. The seamless integration and shared visibility between development and operations workflows helps bridge the silos that exist between development and operations teams. Using a common platform for development, security and operations accelerates agile transformation and helps organizations adopt a product and platform team operating model. VSDPs are a foundational component of self-service platforms for development teams.

Drivers

- Continued cloud adoption and cloud-native architectures.
- Agile and DevOps ways of working with automation and collaboration being the focal points.
- Need for improved developer experience, satisfaction and productivity.
- Need for deeper collaboration and shared visibility across all IT teams (dev, sec and ops).
- Emergence of digital product and platform engineering teams to drive digital transformation initiatives.
- Competitive pressure that demands faster time to market for digital products and services.
- Proliferation of tools and the ensuing complexity of integrating those tools as well as the lack of skills to do so at scale.
- Industry traction around value stream management — VSDPs will increasingly bundle capabilities to analyze value stream metrics and extend the usability of the platforms to business stakeholders.
- Rise of vendors providing a managed open-source software (OSS)-based VSDP. Customers get the best of innovation from the OSS community and the guaranteed SLAs from a technology provider.
- Increased maturity of “as code” approaches to automate security and compliance (via policy as code) and infrastructure (via infrastructure as code).

Obstacles

- Organizations that want to unlock the full benefits of VSDPs must be willing to replace an existing toolchain — either completely or in-part. However, this is an impediment for teams that are resistant to change or view the change as a disruption to their ways of working.
- Organizations accrue technical and skills debt over time due to legacy automation workflows. That hinders teams from adopting new tools.
- The potential for vendor lock-in presents a constraint as well. Dependency on a single provider for a majority of their software development needs increases concentration risk and lowers bargaining leverage. Gartner cautions VSDP buyers that none of the vendors currently provide the full set of application delivery capabilities.
- VSDPs also fall short of providing the required value stream analytics that organizations can use to improve "flow" in the software delivery life cycle.

User Recommendations

- Scale DevOps initiatives by providing VSDPs as a self-service platform to reduce overhead, lower complexity, and ensure consistent and templatized workflows across multiple teams.
- Improve the flow of work by streamlining the software delivery life cycle with VSDPs that provide enhanced visibility, traceability, auditability and observability across the DevOps pipeline.
- Support remote development teams by adopting cloud-based VSDPs and using their integration and collaboration capabilities for code reviews, code sharing and issue tracking.
- Adopt an "as code" approach to managing IT workflows to improve operational efficiency and consistency — pipelines as code, infrastructure as code and policy as code.

Sample Vendors

Atlassian; Calculi; CloudBees; CodeNOW; Copado; Digital.ai; GitHub; GitLab; Harness; JFrog

Gartner Recommended Reading

Market Guide for DevOps Value Stream Delivery Platforms

Market Guide for DevOps Value Stream Management Platforms

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

Why DevOps Success Requires Platform Teams

Platform Teams and AIOps Will Redefine DevOps Approaches by 2025

Platform Ops

Analysis By: Daniel Betts, George Spafford

Benefit Rating: Transformational

Market Penetration: 5% to 20% of target audience

Maturity: Adolescent

Definition:

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

Why This Is Important

Platform ops practices are adopted by organizations needing to scale DevOps initiatives in order 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) practices.

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 in enabling the product teams. This makes it well-suited to support digital business initiatives such as analytics, mobile applications, portals and so forth.

Drivers

- Many large organizations are adopting platform models to scale DevOps, where platform teams build standardized capabilities that enable product teams to deliver customer value.
- Difficulty in providing enough operations expertise in DevOps product teams as they scale, resulting in slower delivery cycles, software defects and frustration.
- Full-stack DevOps team structures are not scalable and often lead to duplication and competing demands that thwart long-term success.
- Challenges to ensure high standards of governance and production efficiency when product teams recreate platforms' capabilities inconsistently from team to team.

Obstacles

- Before embarking on platform ops — picking technologies, hiring a team — you should determine 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 may not be the right approach. Efforts should be focused on an iterative learning approach and scale when ready.
- Having sufficient resources to fulfill the roles required to deliver a shared platform is a challenge for many organizations.
- There can be considerable ramp-up time to enable platform team members with the software engineering, testing and programming skills needed to effectively deliver a platform.
- Organizational size and maturity to adopt platform ops can impact the decision to use platform ops.

User Recommendations

- Establish dedicated platform teams to maintain and continuously improve shared, self-service platforms. Product teams need infrastructure services that are available on-demand to deliver value faster.
- Create a cross-functional team of versatile subject matter experts to support the evolving needs of product teams. This platform team needs longevity and dedicated funding throughout the entire life cycle of the platform.
- Encourage platform team members to work closely with product teams. They must determine which tools the teams need to be more productive, provide education on how to use the platform to meet their needs and implement architectures that are conducive to shared platforms.
- Collaborate with teams other than the product teams — such as security, compliance, finance — to ensure that additional requirements are integrated into the capabilities that the platform offers. Marketing and championing the self-service platforms is critical to their success.

Gartner Recommended Reading

Why DevOps Success Requires Platform Teams

Platform Teams and AIOps Will Redefine DevOps Approaches by 2025

Using Platform Ops to Scale and Accelerate DevOps Adoption

At the Peak

AIOps Platforms

Analysis By: Gregory Murray, Pankaj Prasad

Benefit Rating: Transformational

Market Penetration: 5% to 20% of target audience

Maturity: Emerging

Definition:

AI for IT operations (AIOps) platforms analyze telemetry to improve IT operations and have five characteristics: data ingestion across telemetry domains; topology assembly from network flows, application traces, hypervisor, container and cloud; event correlation to reduce the number of redundant events associated with an incident; pattern recognition to predict incidents or probable root cause; association of probable remediation to augment, accelerate or automate resolution.

Why This Is Important

The combination of increasing application complexity, monitoring tool proliferation, and increasing volumes and varieties of telemetry has shifted complexity from gathering the data to interpreting the data. AIOps capitalizes on machine learning and analytics techniques to classify and cluster diverse telemetry signals in near-real time, at scale, and in ways that exceed human capacity. Even where automated remediation is not possible, this analysis can augment and accelerate human response.

Business Impact

AIOps platforms deliver value through:

- Agility and productivity — By reducing alert fatigue through correlation of related events, operators can focus on fewer, more critical events.
- Service availability and triage cost — By reducing the time and effort required to identify root causes and augmenting, accelerating or automating remediation.
- Compounded value of existing monitoring tools — By unifying the events from siloed monitoring tools and learning actionable event patterns across domains.

Drivers

Demand for AIOps platform capabilities is accelerating and is fueled by:

- Increasing complexity — Organizations use an increasingly complex mix of modern, hybrid IT that relies on a complex, highly integrated mix of on-premises assets, cloud IaaS/PaaS providers and SaaS solutions to deliver their missions.
- Increasing monitoring expectations — Investments and improvements in monitoring and the pursuit of observability are generating more data from more sources. Emerging monitoring platforms and practices, like application performance management (APM) and digital experience monitoring (DEM), present operators with extremely detailed views into their business applications and the end-user experience. Effective use of this additional data requires near-real-time analysis and rationalization against other adjacent telemetry.
- Demands for reliability — Shifts in roles and responsibilities from newer operating models, like DevOps and SRE, in the pursuit of greater availability and faster incident resolution. AIOps platforms enable agility by offloading some of the mechanical tasks of event triage, root cause analysis and solution identification. This both accelerates response for common issues and frees up human creative capacity for novel events and business priorities.

As these trends, themselves, are accelerating, expect the case for AIOps platforms to continue to accelerate.

Obstacles

AIOps platform adoption must overcome the following obstacles:

- Expectations mask real opportunities — Hype is a prolific obstacle to AIOps adoption. It is hard to separate claims of intelligence and magical automation from practical, achievable use cases. Success with AIOps platforms requires clarity on the capabilities and limitations of the available solutions.
- Time to value for high-value use cases — AIOps platforms learn through observation. They learn everything from normal ranges and patterns of data to which automation resolves which issue. Learning requires time because it can take months to develop accurate detection models for rare events.

- Market shifts and maturity — AIOps is evolving and the market is in flux. Monitoring vendors are moving up the stack, AIOps platform vendors are reaching into monitoring domains, and ITSM vendors see AIOps capabilities as a way to extend their reach. Expect both technology advancements and market shifts to change what we currently understand by “state of the art” and “go to market.”

User Recommendations

- Establish clear, realistic use cases for an AIOps platform pilot. This fundamental step underpins an eventual strategy, while scoping the vendor landscape, clarifying the telemetry requirements and separating hype from reality.
- Integrate single-domain AIOps features with an AIOps platform. This approach enables efficient data ingestion and analysis, and the surfacing of insights across domains.
- Mature toward automation integration, ensuring the core benefits of AIOps are realized before full automation. High-risk cases are not ideal for automation but benefit from AIOps accelerating or augmenting human response.
- Shift mechanical operations to analytics and automated execution, while developing creative skills in integration and automation development. Prepare for disruption and adapt to a highly integrated environment. Tackle silos that can create blind spots in your AIOps platform coverage.

Sample Vendors

BigPanda; IBM; Moogsoft; ServiceNow; Splunk

Gartner Recommended Reading

Market Guide for AIOps Platforms

Infographic: Artificial Intelligence Use-Case Prism for AIOps

Solution Path for Adopting AIOps

Understanding the Application of AIOps Disciplines Within IT Operations

Autonomous Testing

Analysis By: Joachim Herschmann, Jim Scheibmeir

Benefit Rating: High

Market Penetration: 1% to 5% of target audience

Maturity: Adolescent

Definition:

Autonomous testing comprises AI- and ML-based technologies and practices to make software testing activities independent from human intervention. It continuously improves testing outcomes by learning from the data collected 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.

Why This Is Important

Software engineering leaders seeking to release faster without degrading quality are looking for more efficient ways of testing which includes all phases of the testing life cycle. As such, autonomous testing is a driver for a holistic approach to automating the broader set of quality and testing activities related to requirements quality, design quality, code quality, release quality, and operational resilience in an integrated way. This increases the degree of autonomy for those activities.

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.

Drivers

- A high dependency on human expertise and interaction limits how quickly modern digital businesses can design, build, and test new software.
- Where automated testing is already in place, current levels of automation often remain below expectations due to a continued dependency on human intervention in maintaining the automation as applications under test (AUT) evolve.
- The pressure to innovate quickly in order to differentiate in the market without compromising on quality relies both on a higher velocity and a higher degree of autonomy of the related activities.
- While delivery cycle time is decreasing, the technical complexity required to deliver a positive user experience and maintain a competitive edge is increasing. The answer is not more testing but more intelligent testing enabled by AI technologies.

Obstacles

- Risk of doing nothing: Waiting until feature-complete autonomous testing solutions are available leads to a loss in competitive advantage, and reduced agility and innovation.
- Unrealistic goals: Underestimating the time required to acquire new skills and setting wrong expectations about the time required to become successful.
- Managing data quality: Gathering, cleaning and processing of data and training of the model are not trivial tasks and require adequate skills. Moreover, they are not yet autonomous processes.
- Internal pushback: Autonomous testing requires significant investment in new areas such as data science and analytics. While this will motivate some team members, others may see the approach as a threat to their known way of working and they may be afraid to adjust.
- Immaturity of tools: Currently available tools are still relatively new, have a narrow scope of technology coverage and still need to prove their value.

User Recommendations

- 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.
- Familiarize yourself with advanced analytics and ML. Invest in augmented analytics tools that employ ML algorithms to mine existing data for current and future projects.
- Allocate ownership and appoint staff with the required skills such as data analytics by identifying the required roles, technologies and practices.
- Select tools that best match available skills and development style by assessing application profiles and different teams' needs.

Sample Vendors

Applitools; Diffblue; Functionize; mabl; ProdPerfect; Testim

Gartner Recommended Reading

Innovation Insight for Autonomous Testing

Innovation Insight for Continuous Quality

Improve Software Quality by Building Digital Immunity

Infographic: Artificial Intelligence Use-Case Prism for Software Development and Testing

Communities of Practice

Analysis By: Thomas Murphy

Benefit Rating: High

Market Penetration: 20% to 50% of target audience

Maturity: Early mainstream

Definition:

A community of practice (CoP) is a community- or stakeholder-owned forum, often initiated by an organization's leaders, to enable collaborative problem solving, knowledge sharing and organizational learning. Agile and DevOps teams require a shift in culture to support greater collaboration and cross-functional learning and to break down traditional silos and their associated centers of excellence (COEs).

Why This Is Important

As more organizations shift to a product model and adopt DevOps practices, CoPs are becoming increasingly important. While CoPs are widely used by agile and DevOps and cloud leaders, we are seeing growing adoption by mainstream clients. Many IT organizations continue to rely on top-down, functionally aligned COEs that focus on policy adherence and governance instead of problem solving. In contrast, CoPs are motivational if they provide a clear vision and space for the community to collaborate, learn and evolve.

Business Impact

CoPs provide organizations with a community-driven, cross-functional approach for learning new skills, behaviors and models to successfully transition to an agile, DevOps and product-based culture.

Our research finds that CoPs:

- Shorten the learning curve for employees
- Provide higher levels of employee satisfaction, leading to higher motivation and innovation
- Respond more rapidly to customer needs and inquiries
- Reduce duplication of effort
- Spawn new ideas for products and services
- Help members develop capabilities that align with organizational needs

Drivers

- DevOps product and platform team members must gain new skills and behaviors and improve the pace of delivery. CoPs provide a community-focused approach for knowledge sharing and organizational learning.
- CoPs are a continued source of strength through interactions and learning that should be captured and shared digitally.
- CoPs are voluntary groups that enable practitioners to create common guidelines and practices based on proven practice.
- Transversal knowledge management through CoPs prevents unintentional functional and product silos from forming.

Obstacles

- Challenges associated with hybrid work models will disrupt organizations that are initiating new CoPs.
- Hierarchical structures and existing reporting chains can feel threatened by the self-directed nature of COPs wanting to lean on COEs and business as usual.
- While COPs are a powerful concept, organizations will also need other structures and practices such as DevOps Dojos or InnerSourcing to drive the best long-term benefits.
- CoPs require management support and active coaching to be effective, continuously create value and prevent rapid obsolescence.

User Recommendations

I&O leaders and applications and software engineering leaders should:

- Empower cross-functional teams to learn and discover through experimentation, collaboration and knowledge sharing by creating and communicating a clear vision for the community with time and space.
- Encourage experimentation and risk taking by dedicating time to the community to build a culture where failure and new approaches are celebrated as learning opportunities.
- Complement the CoP by organizing job rotations, hackathons and technology conferences.

Sample Vendors

Atlassian; GitHub; Microsoft; Slack; Zoom

Gartner Recommended Reading

Foster Communities of Practice to Ensure Successful DevOps

Ignition Guide to Creating Agile Communities of Practice

Hypothesis-Driven Development

Analysis By: Jim Scheibmeir

Benefit Rating: Moderate

Market Penetration: 5% to 20% of target audience

Maturity: Adolescent

Definition:

Hypothesis-driven development (HDD) is software development based on running a series of tests 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.

Why This Is Important

When HDD is combined with agile and DevOps practices, it allows for the opportunity to accelerate the rate of innovation while reducing waste and managing risk. HDD enables an organization to define requirements and create software solutions in problem spaces where little is known about the best solution. It can prevent developing a solution that has no customers. HDD allows teams to test the market to see if the offering and its features are valued.

Business Impact

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

Drivers

Popularized by the book “The Lean Startup” by Eric Ries in 2011, HDD has been widely adopted in software product and game development. It is less common in IT development projects, and its position on the Hype Cycle reflects IT adoption. Gartner's 2020 Agile in the Enterprise survey, conducted online from 8 April to 22 May 2020 (n = 353 IT and IT-business professionals globally), found that 31% of respondents would be utilizing HDD by the end of 2020. HDD adoption is being driven by:

- Need to ensure business growth with changing user patterns, behaviors and new integration opportunities
- Increasing user expectations for application design, quality and experience
- Need to optimize development resources and minimize user fatigue due to application and process changes

Obstacles

- Experimental nature of HDD initiatives, as many IT organizations refuse to start a project where the solution, time and cost, and the user base are not well-understood.
- Frequent changes to user interfaces as well as supporting database and business logic code.
- Lack of talent proficiency in agile and DevOps practices that are required to support fast feature development and deployment to enable HDD.

User Recommendations

- 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.
- Delay using HDD until teams have reached agile and DevOps proficiency.
- Adopt a hypothesis pattern for stories. Replace the traditional story format, i.e., "As an <identify user>, I need to do <task> in order to accomplish <outcome>" or the BDD/Gherkin format, i.e., "given <initial state>, when <event>, then <result>" with 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>."
- Use lean startup principles to develop solutions with a four-stage HDD cycle: *think it, build it, ship it, tweak it.*

Sample Vendors

CloudBees; GitLab; LaunchDarkly; Optimizely; Split

Gartner Recommended Reading

Solution Path for Agile Transformation

Extend Agile With DevOps for Continuous Delivery

Quick Answer: What Are the Essential Shift-Right Practices for Testing in Production?

Guidance Framework for Adopting Lean UX

Learn From the Spotify Model for Better Enterprise Agile Scaling

Dojos

Analysis By: George Spafford

Benefit Rating: High

Market Penetration: 1% to 5% of target audience

Maturity: Adolescent

Definition:

A dojo is a gathering place (real or virtual) where students learn and practice together in an experiential manner in accordance with their skill level. Students can expand their skill set by learning about new methods and tools, while those with more experience can share and refine skills.

Why This Is Important

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 tend to fail. The use of a dojo can be layered with other methods to improve organizational learning for people ranging from newcomers to experienced ones.

Business Impact

Dojos are experiential learning opportunities 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.

Drivers

- Digital business initiatives are requiring new approaches and technologies in I&O.
- These approaches include moving systems to the cloud, DevOps, infrastructure as code and platforms — all of which require the staff to learn and change.
- I&O leaders are seeking organizational learning methods that can scale to help large numbers of employees learn.

Obstacles

- General misconceptions around how to approach dojos include huge investment being required to start dojos.
- People wanting to start dojos tend to start their initiative in an expanded manner requiring too many approvals, which leads to unwanted delays.
- Finding skilled practitioners who can also be effective instructors for the dojo is challenging.

User Recommendations

- Start small with the resources you have available and demonstrate value.
- Attract people to the dojos by understanding and delivering value that resonates with them.
- Use practitioners with real-world knowledge to teach in an experiential style so that students can learn by doing.
- Integrate the dojo into the DevOps strategy and understand which classes need to be held to help prepare people to move in the required direction.
- Co-create a dojo charter with all stakeholders by establishing a shared vision for the dojo, and using metrics that track progress and success.
- The dojos are a method of organizational learning. Clients tend to use three or more approaches. Investigate others such as communities of practice, job rotations, mentoring, internal technical conferences and so forth.

Gartner Recommended Reading

Agile Learning Manifesto

Rebalanced Technical Skills Portfolio (Nationwide)

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

Programmable Infrastructure

Analysis By: Nathan Hill, Philip Dawson

Benefit Rating: High

Market Penetration: 1% to 5% of target audience

Maturity: Emerging

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.

Why This Is Important

A continuous-delivery approach requires continuous insight and the ability to automate application responses. Programmable infrastructure ensures optimal resource utilization while driving cost-efficiencies. Moving to an API-driven infrastructure is the key first step to enabling anti-fragile and sustainable automation through programmatic techniques.

Business Impact

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.

Drivers

- Programmable infrastructure strategies can be applied to private cloud, hybrid cloud and infrastructure platforms, as well as public cloud. Demand for programmable infrastructure grows as heterogeneous infrastructure strategies are embraced.
- Programmable infrastructure is needed to manage the life cycle of infrastructure delivery from provisioning, resizing and reallocation to reclamation, or in the case of elastic external resources, and the termination of consumption.
- Programmable infrastructure is needed to optimize the infrastructure life cycle and more importantly enable the desired (performance, cost, speed) infrastructure provisioning in line with business demands.

Obstacles

- Cost of refreshing API-enabled infrastructure components on-premises. Applying automation to existing monolithic infrastructure components will fail due to lack of platform agility.
- Lack of maturity in APIs that enable integration across different infrastructure platforms.
- Lack of open APIs/API compatibility across vendor platforms.
- The scarcity of programmatic skills within I&O, especially in web technologies (such as HTTP and JSON) to develop these APIs.

User Recommendations

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 organization wide 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.

Sample Vendors

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

Gartner Recommended Reading

Programmable Infrastructure Is Foundational to Infrastructure-Led Disruption

How to Lead Digital Disruption With Programmable Infrastructure

Understand the Hype, Hope and Reality of Composable Infrastructure

How to Lead Digital Disruption With Programmable Infrastructure

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

Observability

Analysis By: Padraig Byrne, Gregg Siegfried

Benefit Rating: Transformational

Market Penetration: 5% to 20% of target audience

Maturity: Adolescent

Definition:

Observability is the characteristic of software and systems that allows them to be understood based on their outputs, and allows questions about their behavior to be answered. Tools that facilitate software observability enable observers to collect and explore high cardinality telemetry using techniques that iteratively narrow the possible explanations for errant behavior. The insights that observability provides are useful to operations, platform and DevOps teams.

Why This Is Important

The inherent complexity of modern applications and distributed systems, and the rise of practices such as DevOps, has left many organizations frustrated with legacy monitoring. These tools and techniques are unable to do more than collect and display external signals, resulting in monitoring that is only partially effective. Observability tools enable a skilled observer — often a software developer or site reliability engineer — to more effectively explain unexpected system behavior.

Business Impact

Observability tools have the potential to reduce both the number of service outages and their impact and severity. Their use by organizations can improve the quality of software, because previously invisible (unknown) defects and anomalies are identified and corrected. By allowing product owners to better understand how their products are used, observability supports more accurate and usable software, and a reduction in the number and severity of events affecting service.

Drivers

The message of observability is resonating powerfully in the industry:

- The term observability is everywhere (e.g., in monitoring products, automation tools and enterprise suites) and 2021 is shaping up to be the year of observability. Although the tools are getting better, the message is being obscured by its ubiquity.
- OpenTelemetry's progress in 2021 as the "observability framework for cloud-native software" has further heightened the profile of observability and its toolchain.

- Traditional monitoring systems capture and examine (possibly adaptive) signals in relative isolation, with alerts tied to threshold or rate-of-change violations. To achieve this, one must be aware of possible issues beforehand, and instrument accordingly. Given the complexity of modern applications, it is unfeasible to do this for all potential issues.

- Observability tools enable a skilled observer — a software developer or a site reliability engineer — to more effectively explain unexpected system behavior, provided enough instrumentation is available. Integrating software observability with artificial intelligence for IT operations (AIOps) to automate subsequent determinations is a potential future development. However, such AIOps-based observability systems do not yet exist.

- Observability is an evolution of longstanding technologies and methods, and monitoring vendors are starting to include observability ideas inside their products, while new companies are building around the idea. Hence, time to Plateau of Productivity will be shorter than normal innovation profiles.

Obstacles

There are a number of obstacles to implementation of observability:

- Conservative nature of IT operations — In many large enterprises, the role of IT operations has been to keep the lights on, despite constant change. This, combined with the longevity of existing monitoring tools, means that new technology is slow to be adopted.

- Many vendors, including those in the network and security domains, are using the term “observability” to differentiate their products. However, little consensus exists on the definition of observability and the benefits it provides, causing confusion among enterprises purchasing tools.

- Hyperscale cloud vendors — for example, Amazon Web Services (AWS), Microsoft Azure and Google Cloud Platform (GCP) — continue to improve their observability tools. These services may reduce or eliminate the future need for third-party tools.

User Recommendations

Users starting a DevOps journey should assess software observability tools to integrate into their continuous integration/continuous delivery (CI/CD) pipelines, feedback loops and retrospectives. They should also:

- Investigate problems that can't be framed by traditional monitoring by using observability to create time and site reliability engineering (SRE) error budgeting, adding flexibility to incident investigations.
- Enable observability by selecting vendors that use open standards for collection, such as OpenTelemetry.
- Tie service-level objectives (SLOs) to desired business outcomes, using specific metrics and leverage observability tools to understand variations.
- Avoid the implication that observability is synonymous with monitoring. At a minimum, observability represents the internal, rather than external, perspective. Ideally, observability solutions make identifying unknown unknowns less onerous by identifying correlations and patterns in streams of telemetry that may escape notice.

Sample Vendors

Amazon Web Services (AWS); Chronosphere; Google; Honeycomb; Lightstep; Microsoft

Gartner Recommended Reading

Monitoring and Observability for Modern Services and Infrastructure

Magic Quadrant for Application Performance Monitoring

Cool Vendors in Monitoring, Observability and Cloud Operations

Innovation Insight for Observability

Infrastructure Automation

Analysis By: Chris Saunderson

Benefit Rating: High

Market Penetration: 20% to 50% of target audience

Maturity: Mature mainstream

Definition:

Infrastructure automation (IA) allows DevOps and I&O teams to design and implement self-service, automated delivery services across on-premises and cloud environments. IA enables 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 a self-service catalog.

Why This Is Important

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.

Business Impact

By enabling engineers and developers to automate the provisioning and configuration of infrastructure, organizations will realize:

- Agility improvements — continuous integration and delivery of infrastructure.
- Productivity gains — version controlled, faster, repeatable deployment.
- Cost improvements — reductions in manual effort through increased automation.
- Risk mitigation — standardized configurations across all elements driving compliance.
- Efficiency — reducing toil and enabling value-added work.

Drivers

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
- Enable consumption of programmable infrastructure

- Self-service and on-demand environment creation
- Support DevOps initiatives (continuous integration/delivery/deployment)
- Resource provisioning
- Operational configuration management efficiencies
- Policy-based delivery and assessment/enforcement of deployments
- Enterprise-level framework to enable tooling strategy maturation

Obstacles

- Combination of tools needed to deliver IA capability can lead to an increase in tool count.
- Software engineering skills and practices are required to get maximum value from tool investments.
- Migration from point activities to infrastructure capability releases is a steep learning curve.
- IA vendor capability overlaps muddies tool landscape.
- Steep learning curves can lead to developers and administrators riveting to known scripting methods to deliver needed capabilities.

User Recommendations

- Identify existing IA tools in use to catalog capabilities and identify use cases.
- Assess existing internal IT skills to incorporate training needs to enable IA more fully.
- Baseline how managed systems and tooling will be consumed (engineer, self-service catalog, API or on-demand).
- Integrate security and compliance requirements into evaluation criteria.
- Develop an IA tooling strategy that incorporates current needs and near-term roadmap evolution (cloud adoption/proliferation).

Sample Vendors

Amazon Web Services; Chef; HashiCorp; Inedo; Microsoft; Pulumi; Puppet; Quali; VMware

Gartner Recommended Reading

Market Guide for Infrastructure Automation Tools

Innovation Insight for Continuous Infrastructure Automation

How to Build Agile Infrastructure Platforms That Enable Rapid Product Innovation

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

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

How to Lead Digital Disruption With Programmable Infrastructure

Assessing HashiCorp Terraform for Provisioning Cloud Infrastructure

Site Reliability Engineering

Analysis By: George Spafford, Daniel Betts

Benefit Rating: Transformational

Market Penetration: 5% to 20% of target audience

Maturity: Adolescent

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). Site reliability engineers work with product or platform teams to design and continuously improve systems that meet defined SLOs.

Why This Is Important

SRE emphasizes the engineering disciplines that lead to resilience; but individual organizations implement SRE in widely varying ways. 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 to enable organizational learning, continual improvement and reductions in unplanned work.

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 create new products or platforms that warrant the investment in reliability.

Drivers

- SRE is intended to help manage the risks of rapid change, through the use of 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 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), and usage of infrastructure-as-code capabilities (which usually requires a cloud platform).
- SRE also uses automation to reduce manual processes, and leverages resilient system engineering principles and an agile development process that employs continuous integration/continuous deployment (CI/CD).

Obstacles

- Among the biggest roadblocks toward getting started with SRE is the conceptual foundation necessary to obtain buy-in from both management and agile or DevOps teams. SRE is a significant investment, and organizations need to organize implementation based on the mission and on the adequate budget.
- Implementing SRE without an experienced site reliability engineer requires considerable learning and improvement in order to be effective. By nature, SRE embraces risk, and this conceptual mindset is difficult for many traditional employees to completely understand.
- SRE calls for blameless analysis of understanding how teams can learn from a problem, rather than assigning blame. This means understanding what made the problem happen and how to mitigate, and publishing the results.

User Recommendations

- Engage an executive sponsor to support the initiative.
- Demonstrate sufficient value to improve credibility and support; have an acceptable level of risk.
- Build an initial SRE team with a collaborative engineering mindset and a desire to learn and improve, and automate tasks to reduce repetitive manual work.
- Define clear SLOs and service-level indicators (SLIs) to be monitored and reported against.
- Instill collaborative working between site reliability engineers and developers to help them learn how to design and build their product to meet SLOs.
- Drive an iterative approach to start and evolve SRE practices.

Sample Vendors

BigPanda; Blameless; Datadog; New Relic; OpsRamp; PagerDuty; Splunk

Sliding into the Trough

Container Management

Analysis By: Dennis Smith

Benefit Rating: High

Market Penetration: 20% to 50% of target audience

Maturity: Early mainstream

Definition:

To manage containers at scale, container management provides capabilities such as container runtimes, container orchestration and scheduling, and resource management. 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.

Why This Is Important

Container runtimes simplify use of container functionality and enable integration with DevOps tooling and workflows. Productivity and/or agility benefits of containers include accelerating and simplifying the application life cycle, enabling workload portability between different environments and improving resource utilization efficiency. Container management makes it easier to achieve scalability and production readiness. It also optimizes the environment to meet business SLAs.

Business Impact

Gartner surveys and client interactions show that the demand for containers continues to rise. This is due to application developers' and DevOps teams' preference for container runtimes, which have introduced container packaging formats. Developers have quickly progressed from leveraging containers on their desktops to needing environments that can run and operate containers at scale, introducing the need for container management.

Drivers

- Container runtimes, frameworks and other management software provide capabilities such as packaging, placement and deployment, and fault tolerance (for example, clusters of nodes running the application).

- The emergence of de facto standards (for example, Kubernetes) and offerings from the public cloud providers have simplified deploying containers at scale. Many vendors enable 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.

- Container-related edge computing use cases have increased in industries that need to get compute and data closer to the activity (for example, telcos, manufacturing plants, etc.).

- Data analytics use cases have emerged over the past few years, as have operational control planes that enable the management of container nodes and clusters.

- All major public cloud service providers now offer on-premises container solutions. Independent software vendors (ISVs) are starting to package their software for container management systems.

- Some enterprises have scaled sophisticated deployments, and many more have recently commenced container deployments or are planning to. This is expected to increase as enterprises restart application modernization projects postpandemic.

Obstacles

- Third-party container management software faces huge competition in the container offerings from the public cloud providers, both with public cloud deployments and the extension of their software to on-premises environments. These offerings are also challenged by ISVs that choose to craft open-source components with their software during the distribution process.

- More abstracted, serverless offerings may enable enterprises to forgo container management. Among these services are Knative, AWS Lambda and Fargate, Azure Functions, and Google's Cloud Run. These services embed container management in a manner that is transparent to the user.

- Organizations that perform relatively little app development or make limited use of DevOps principles are served by SaaS, ISV and/or traditional application development packaging methods.

User Recommendations

- Determine if your organization is a good candidate for container management software adoption by weighing organizational goals of increased software velocity and immutable infrastructure, and its hybrid cloud requirements, against the effort required to operate third-party container management software.
- Leverage container management capabilities integrated into cloud IaaS and PaaS providers' service offerings by experimenting with process and workflow changes that accommodate the incorporation of containers.
- Avoid open-source deployments unless the organization has ample in-house expertise to support.

Sample Vendors

Amazon Web Services (AWS); Google; IBM; Microsoft; Mirantis; SUSE (Rancher Labs); Red Hat; VMware

Gartner Recommended Reading

Market Guide for Container Management

Continuous Delivery

Analysis By: Hassan Ennaciri

Benefit Rating: High

Market Penetration: 20% to 50% of target audience

Maturity: Adolescent

Definition:

Continuous delivery (CD) is a software engineering approach that enables teams to produce valuable software in short cycles while ensuring that the software can be reliably released at any time. Through dependable, low-risk releases, CD makes it possible to continuously adapt software to incorporate user feedback, shifts in the market and changes to business strategy. This approach requires engineering discipline to facilitate the complete automation of the software delivery pipeline.

Why This Is Important

Growing DevOps initiative success continues to drive enterprise investments in CD capabilities. CD improves release velocity and reliability while simplifying compliance enforcement via automation. It is a prerequisite and a first step to continuous deployments for organizations that aspire to push changes with zero downtime.

Business Impact

CD is a key practice for 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.

Drivers

- Agile and DevOps adoption to deliver solutions
- Need to improve release velocity and reliability
- Need to shift left and simplify compliance enforcement via automation
- Need to improve software development life cycle (SDLC) to more consistently deploy application builds and updates, buy extending the benefits of Continuous integration (CI) and automated testing to continuously build deployable software
- CD is a prerequisite and first step to continuous deployments for organizations aspiring to push changes with zero downtime

Obstacles

- Organizational culture and collaboration between teams with different roles and skills is a major barrier to CD success. Agile practices that helped bridge the gap between business and development need to be extended to deployment, environment configuration, monitoring and support activities.
- Lack of value stream mapping of product delivery hinders visibility and quick feedback loops needed for continuous improvements. Teams struggle to improve and focus on value work as they don't have insights to the critical steps in the process, the time each step takes handoffs and wait states.
- Other challenges that impact success of CD include application architecture and lack of automation in all areas of testing, environment provisioning, configuration security and compliance.

User Recommendations

- When starting a CD initiative, enterprises must consider all associated technologies and take an iterative 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 value stream metrics.
- DevOps teams must evaluate and invest in associated tooling, such as application release orchestration tools, containers and infrastructure automation tools. These tools provide some degree of environment modeling and management, which can prove invaluable for scaling CD capabilities across multiple applications.
- DevOps teams need to consider DevOps Value Stream Delivery Platforms (VSDPs) to provide fully integrated capabilities that enable continuous delivery of software.

Sample Vendors

Broadcom; Calculi; CloudBees; GitLab; Harness; JFrog

Gartner Recommended Reading

How to Build and Evolve Your DevOps Toolchains

Market Guide for DevOps Value Stream Delivery Platforms

ARO

Analysis By: Daniel Betts

Benefit Rating: High

Market Penetration: 20% to 50% of target audience

Maturity: Early mainstream

Definition:

Application release orchestration (ARO) tools combine 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.

Why This Is Important

Demand for new applications and features delivered faster to support business agility will continue growing for the foreseeable future. As a result, the resulting tumultuous and transformative activity (often in the form of DevOps initiatives) 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.

Business Impact

ARO tools provide transparency improvements to the release management process by making visible bottlenecks and wait states in areas such as infrastructure provisioning 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 more quickly and reliably delivered.

Drivers

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 — faster delivery of new applications and updates in response to changing market demands
- Cost reduction — significant reduction of manual interactions by high-skill and high-cost staff, freeing them to work on higher-value activities
- Risk mitigation — consistent use of standardized documented processes and configurations across multiple technology domains
- Improvement and remediation — use of dashboard views over metrics outlining and predicting release quality and throughput
- Improved visibility and traceability to the release process

Obstacles

- 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.
- Market has changed — the current landscape of vendors has built feature-comparable products, with some incremental differences.
- 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).

User Recommendations

- Organize activities into three categories that align with ARO critical capabilities: deployment automation, pipeline and environment management, and release orchestration.
- Prioritize capabilities for current and future needs prior to evaluating vendors. The better insight you have of your current release activities, the faster you will see value from ARO.
- When evaluating ARO tools, features should be the No. 1 driver for selection. Requirements should be mapped to capabilities as part of the evaluation. Where legacy environments exist, those should be weighted more heavily.
- Simplify and speed up the transition to automated workflows by documenting current application release procedures, activities and artifacts performed by both traditional and DevOps teams.
- Ensure the ARO platform dashboard with release performance and underlying platforms metrics, code deployment cycles, lead time from commit to deploy, incident recovery, change failure rate and the factors that drive them.

Sample Vendors

Broadcom; CitLab; CloudBees; Digital.ai; IBM; Microsoft; Plutora

Gartner Recommended Reading

Market Guide for DevOps Value Stream Delivery Platforms

Market Guide for DevOps Value Stream Management Platforms

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

Why DevOps Success Requires Platform Teams

Platform Teams and AIOps Will Redefine DevOps Approaches by 2025

DevOps Toolchain

Analysis By: Thomas Murphy

Benefit Rating: High

Market Penetration: More than 50% of target audience

Maturity: Mature mainstream

Definition:

A DevOps toolchain comprises tools that support DevOps pipeline activities and provide fast feedback. It primarily focuses on the code to build/test/deploy sequences. While some tools support specific pipeline activities, vendors are increasingly delivering integrated solutions and supporting container-centric delivery. The center point of these solutions is the CI/CD orchestration system.

Why This Is Important

DevOps toolchains enable development and operations teams to deliver new and updated applications faster. They can include dozens of unintegrated tools, which makes automation a complex and arduous task. Our 2019 DevOps survey found that organizations have, on average, 28 toolchains, which represents a huge undertaking to create and maintain. Thus, DevOps toolchains demand greater collaboration between development and operations. DevOps cannot simply be achieved by adopting tools.

Business Impact

Delivering business value is central to DevOps. A well-designed, integrated and automated DevOps toolchain enables development and operations team members to collaborate in achieving common objectives and metrics to ensure quality, timely application delivery. DevOps teams deliver the greatest business impact when they adopt an agile mindset and have the right technology to support the activity needs of individual team members.

Drivers

- Core tooling around CI/CD 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 core pipelines evolve, a new wave of broader “toolchains” is emerging around value stream delivery. We expect organizations will have multiple toolchains of the pipeline variety, and these pipelines will feed data into value stream management tools.
- The market continues to evolve via acquisitions, the emergence of open source and new commercial products, and the continued adoption of cloud architecture.

Obstacles

- Existing investments in current tools and lack of migration support dampens ROI for shift to single pipeline, and developer familiarity with existing tools slows toolchain adoption.
- Hesitancy to adopt toolchains until all capabilities are available hinders toolchain maturation. Vendors provide 80% solutions, but still require best-of-breed integrations, which leads to capability overlap or double investment.
- Shifting pricing models as vendors move from per seat to consumption meters and the shift from on-premises to cloud-based solutions, which may occur at a pace organizations aren't able to accommodate.

User Recommendations

I&O leaders and applications and software engineering leaders should:

- Develop a toolchain strategy by establishing business objectives, identifying practices to achieve those objectives and then selecting tools to support those practices. Tool selection should be the last step. When evaluating tools — even open-source options — account for the costs attached to learning, integrating and replacing them.
- Focus on removing the greatest constraint by automating development and continuous delivery processes. Each DevOps product or platform team member should understand the capabilities and contribution of each tool and work to minimize gaps and overlaps between tools. The teams should also use software engineering practices such as version control, code management and managed distribution.
- Manage and evolve DevOps toolchains securely by establishing a toolchain community of practice, using pipeline integrated security and compliance and developing effective test automation.

Sample Vendors

Amazon Web Services; Atlassian; CloudBees; Codefresh; GitLab; Harness; HashiCorp; Microsoft

Gartner Recommended Reading

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

How to Build and Evolve Your DevOps Toolchains

Ignition Guide to Managing a DevOps Toolchain

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

Case Study: Manage the DevOps Toolchain as a Product (LexisNexis)

Immutable Infrastructure

Analysis By: Neil MacDonald, Tony Harvey

Benefit Rating: Moderate

Market Penetration: 5% to 20% of target audience

Maturity: Adolescent

Definition:

Immutable infrastructure is a process pattern (not a technology) in which the system and application infrastructure, once deployed into production, is never updated in place. Instead, when changes are required, the infrastructure and applications are simply replaced from the development pipeline.

Why This Is Important

Immutable infrastructure ensures 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. This approach is easier to adopt and often applied with cloud-native applications.

Business Impact

Taking an immutable approach to workload and application management simplifies automated problem resolution by reducing the options for corrective action to, essentially, one — repair the application or image in the development pipeline and re-release into production. The result is an improved security posture with fewer vulnerabilities and faster time to remediate when new issues are identified.

Drivers

- Linux containers and Kubernetes are being widely adopted. Containers improve the practicality of implementing immutable infrastructure and will drive greater adoption.
- Interest in zero trust and other advanced security postures where immutable infrastructure can be used to proactively regenerate workloads in production from a known good state (assuming compromise), a concept referred to as “systematic workload reprovisioning.”
- For cloud native application development projects, immutable infrastructure simplifies change management, supports faster and safer upgrades, reduces operational errors, improves security and simplifies troubleshooting.

Obstacles

- The use of immutable infrastructure requires a strict operational discipline that many organizations haven’t yet achieved, or have achieved for only a subset of applications.
- IT administrators are reluctant to give up the ability to modify or patch runtime systems.
- 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.
- Many application stacks have elements that are deployed in the form of virtual machine images. VM replacement is slower and requires greater coordination than other workload components such as containers.

User Recommendations

- Reduce or eliminate configuration drift by establishing a policy that no software, including the OS, is ever patched in production. Updates must be made to the individual components, versioned in a source-code-control repository, then redeployed for consistency.
- Prevent unauthorized change by turning off all normal administrative access to production compute resources — for example, by not permitting SSH or RDP access.
- Adopt immutable infrastructure principles with cloud-native applications first. Cloud-native workloads are more suitable for immutable infrastructure architecture than traditional on-premises workloads.
- Treat scripts, recipes and other code used for infrastructure automation similarly to the application source code itself, which mandates good software engineering discipline.

Sample Vendors

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

Gartner Recommended Reading

How to Make Cloud More Secure Than Your Own Data Center

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

SDI

Analysis By: Philip Dawson

Benefit Rating: High

Market Penetration: 20% to 50% of target audience

Maturity: Obsolete

Definition:

Software-defined infrastructure (SDI) includes the broad set of software-defined infrastructure components and the software-defined data center, network, storage and compute. SDI also includes non-data-center infrastructure deployed in Internet of Things (IoT) applications and an SD edge infrastructure.

Why This Is Important

Software defined is the further abstraction of software from hardware. It enables business to be more agile and flexible by enabling programmatic control of the infrastructure through software interfaces. SDI combines 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.

Business Impact

While data center SDI is embedded in other data center initiatives like cloud and hyperconverged infrastructure, SDI is now focused in key verticals operating in multiple, geographically-distributed locations (such as retail, manufacturing, retail banking, distribution and utilities). And it continues to extend IoT and non-data-center SDI initiatives for new IT operations and functions.

Drivers

- SDI data center infrastructure is well-covered with compute (SDC), network (SDN & SDWAN) and storage (SDS), but SDI also extends to non-data-center infrastructure with the use of monitoring devices or machines that are software-defined.
- SDI reaches beyond and between SDDCs, and leverages SDI benefits and features for new multimode applications and edge IoT endpoints.
- In 2021, SDIs' lingering presence of hype 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.
- 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.

Obstacles

- SDI is now tied to vendor technology, not interoperability.
- SDI has now overlapped other integrated systems taxonomy like hyperconvergence, driving cloud adoption.
- In 2021, we are seeing SDI continue to release vendor-specific silo technology (not heterogeneous service driven) and, hence, is obsolete as multivendor interoperability standards and technology silos.

User Recommendations

- Extend the program to include the integration and measurement of non-data-center edge infrastructure, as SDI initiatives roll out.
- Focus on core IT SDI for compute, network, storage and facilities, but expand the impact of SDI on IoT, edge computing, remote office/branch office (ROBO) and other operational technologies.
- Anticipate SDI to be tied to a specific vendor or technology silo such as storage and network hardware or virtualization software. Be cautious not to commit to a vendor's SDI without realizing the specific area of lock-in.

Sample Vendors

IBM; Intel; Microsoft; Red Hat; VMware; Wipro

Gartner Recommended Reading

The Road to Intelligent Infrastructure and Beyond

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

Climbing the Slope

APM

Analysis By: Padraig Byrne, Mrudula Bangera, Josh Chessman

Benefit Rating: High

Market Penetration: 20% to 50% of target audience

Maturity: Mature mainstream

Definition:

Application performance monitoring (APM) enables the observation of application behavior and its dependencies, users and business key performance indicators (KPIs) throughout the application's life cycle. The applications being observed may be developed internally, as packaged applications or as software as a service (SaaS).

Why This Is Important

The APM market continues to evolve beyond its core root of server-side application monitoring as organizations seek to optimize business outcomes, enhance user experience and improve application performance. It is no longer sufficient to monitor one aspect of the technology stack; nor is it enough to deploy proprietary technologies to collect performance data. The exponential growth in mobile, cloud-native applications continues to fuel the APM market.

Business Impact

APM solutions enable businesses to examine modern applications' end-to-end performance, coupled with detailed inspections to quickly identify service-impacting outages. As organizations continue to embrace digital transformation, their need for agility in order to succeed with these transformation initiatives increases. Therefore, APM solutions must support that need for agility, aid in its acceleration and effectiveness, and not be perceived as just another monitoring tool.

Drivers

- Unified monitoring: New application monitoring tools are becoming more unified. This approach requires platforms that share common data models to conduct correlation analysis and other critical functions of APM.

- Holistic monitoring: Modern monitoring platforms are becoming more holistic in terms of the types of data they can ingest, analyze and integrate. The continued adoption of new application development and operations technologies requires monitoring teams to constantly test the limits of their monitoring products.
- Shift-left monitoring: Testing in preproduction and integration with continuous integration/continuous delivery (CI/CD) tools have become the new monitoring norm, increasing the quality and robustness of the finished product.
- Intelligent monitoring: The use of logs, traces, metrics and multiple other types of telemetry is enabling operations and monitoring teams to find unexpected patterns in high-volume, multidimensional datasets using artificial intelligence for IT operations (AIOps) technologies.
- Business monitoring: Digital transformation, along with the effects of remote work due to the COVID-19 pandemic, has highlighted the need to monitor technology as well as user experience and its impact on business outcomes.

Obstacles

- APM software often fails to provide a complete solution, requiring organizations to pivot between tools, wasting time and resources while struggling to find the root cause of the problem.
- Containers/microservices, Internet of Things (IoT), cloud and software-defined anything (SDx) changes are coming to IT operations environments faster than monitoring strategies are evolving to handle them. This is leading to visibility gaps and performance challenges.
- IT monitoring teams still rely on manually invoked runbooks or ad hoc scripts to remediate problems, hindering infrastructure and operations (I&O) leaders’ ability to deploy and monitor new technologies.
- Vendors promise a lot when it comes to AIOps, but the reality is that most uses of AI/ML in APM are still evolving and remain immature.
- There is a major disconnect between what I&O monitors and what the business cares about, which can have significant negative implications for the business.

User Recommendations

- Choose vendors that assist in relating application performance to business objectives and serve not only ITOps, but also DevOps, application owners and lines of business, providing value throughout the application life cycle. 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: mapping and monitoring of customer and business journeys; bidirectional integration with the DevOps toolchain; new emerging standards in instrumentation, such as OpenTelemetry; cloud-native monitoring with an API-first approach; application security; and integrations with your existing or planned IT service management (ITSM) and configuration management database (CMDB) tools.

Sample Vendors

Cisco Systems; Datadog; Dynatrace; IBM Instana; New Relic; Splunk

Gartner Recommended Reading

Magic Quadrant for Application Performance Monitoring

Critical Capabilities for Application Performance Monitoring

DevSecOps

Analysis By: Neil MacDonald, Mark Horvath

Benefit Rating: Transformational

Market Penetration: 20% to 50% of target audience

Maturity: Early mainstream

Definition:

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

Why This Is Important

As IT development processes become more agile (including shifts to DevOps operating models), security should be proactively and seamlessly integrated into them — DevSecOps. DevSecOps offers a means of effectively integrating security into the development process, eliminating or reducing friction between security and development, and pragmatically achieving a secure, workable software development life cycle (SDLC).

Business Impact

The goal is to enable development to move faster without compromising on security and compliance. 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.

Drivers

- Adoption of DevOps and other rapid development practices requires security and compliance testing that can keep up with the pace of development.
- DevSecOps offerings are applied as early as possible in the development process whereas traditional application security testing (AST) tools associated with older development models are applied late in the development cycle, frustrating developers and business stakeholders.
- Testing results need to be integrated into the development process in ways that complement developer’s existing workflows, not require them to learn skills seemingly unrelated to their goals.
- The use of open source has greatly increased the risk to the inadvertent use of known vulnerable components and frameworks by developers.

Obstacles

- Developers view security testing tools as slowing them down in getting new releases out.
- Implemented incorrectly, siloed and cumbersome security testing is the antithesis of DevOps.
- Developers don't want to leave their development (continuous integration/continuous delivery [CI/CD]) pipeline to view results of security and compliance testing tools.
- Historically, static application security testing (SAST) and dynamic application security testing (DAST) tools have been plagued with false positives or vague information, frustrating developers.
- The diversity of developer tools used in a modern CI/CD pipeline will complicate seamless integration of DevSecOps offerings.

User Recommendations

- 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.
- Prioritize the identification of open-source software (OSS) components and vulnerabilities in development (referred to as software composition analysis).
- 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 apply security policies in container- and Kubernetes-based environments.
- Favor offerings that can link scanning in development (including containers) to correct configuration and protection at runtime.

Sample Vendors

Apiiro; Aqua Security; Contrast Security; IBM (Red Hat), NowSecure; Palo Alto Networks; ShiftLeft; Snyk; Sonatype; Veracode

Gartner 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

Market Guide for Software Composition Analysis

How to Make Cloud More Secure Than Your Own Data Center

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

Continuous Configuration Automation

Analysis By: Chris Saunderson

Benefit Rating: High

Market Penetration: 20% to 50% of target audience

Maturity: Mature mainstream

Definition:

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

Why This Is Important

CCA tools have proven critical to operational efficiency and enabling DevOps initiatives by their ability to manage and deliver infrastructure and associated software and configuration changes as code. CCA tools have continued to expand their reach into networking, containers, patching, compliance and security use cases, enabling programmatic assessment and the remediation of configuration deltas. They also offer a framework for the automation of Day No. 2 operations of delivered systems.

Business Impact

By enabling automation of the deployment and configuration of settings and software programmatically, organizations realize:

- **Agility improvements** — continuous integration/delivery for infrastructure and operations (I&O) infrastructure management
- **Productivity gains** — repeatable, version-controlled infrastructure deployment
- **Cost optimization** — reductions in manual interventions by skilled staff
- **Risk mitigation** — automated compliance using standardized, documented processes and configurations

Drivers

- Organizations need a broader set of deployment and automation functions beyond configuration management, including: infrastructure-as-code (IaC); patching; application release orchestration (ARO); configuration auditing (e.g., for regulatory or internal policy compliance); and orchestration of operational tasks.
- Enabling an automation framework that enables deployment automation and Day No. 2 operational automation of changes, compliance and response to incidents or problems
- Imperative for infrastructure administrators to mature from task-based scripting to a more-structured approach to automation and delivery
- Need for repeatable, standardized deployments available to an end-user or I&O administrator that enables quick delivery of infrastructure to meet end-user needs and I&O policies and baselines.

Obstacles

- Confusion in the capabilities in automation tools leads to assumptions and misunderstandings of capabilities.
- Developers and administrators may use CCA on an insular basis, further inhibiting enterprisewide adoption.
- IT skill sets hinder adoption of these tools, requiring source code management and software engineering skills to make full use of capabilities.
- The growing use of immutable infrastructure has created confusion about the role of CCA tools; however, CCA tools and containers can be used in a highly complementary manner.

User Recommendations

- Clarify the role CCA tools fulfill in their toolchain and make selections based on the tasks that are in scope.
- Evaluate the availability of content against organizational use cases. Prioritize CCA tools that provide out-of-the-box content that addresses current pain points and accelerates time to value.
- Costs associated with CCA tools extend beyond just the licensing cost: Both professional services for enablement and training requirements should be included in cost evaluations.
- Expect to invest in training because not all infrastructure administrators have the skills needed to use these tools successfully.
- Guard against developers and administrators reverting to known scripting methods to complete specific tasks in place of using CCA capabilities.
- DevOps and IT operations leaders maximize the value of CCA tool investments by ensuring that their organizations' culture can embrace CCA tools strategically.

Sample Vendors

CFEngine; Chef; Inedo; Puppet; Red Hat; SaltStack

Gartner Recommended Reading

Market Guide for Infrastructure Automation Tools

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

Innovation Insight for Continuous Infrastructure Automation

Entering the Plateau

Cloud Management Platforms

Analysis By: Dennis Smith

Benefit Rating: Low

Market Penetration: 5% to 20% of target audience

Maturity: Mature mainstream

Definition:

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

Why This Is Important

Managing a workload after it has migrated to the cloud is as important as the initial cloud migration. Enterprises will deploy CMPs (often 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.

Business Impact

- CMPs provide the functionality to address initial provisioning of cloud resources, as well as ongoing management.
- The recent CMP market focus has been on preventing enterprises from overspending or leaving themselves vulnerable to security issues — two key items to avoid when adopting cloud services.

Drivers

- Organizations need to address multicloud requirements. In some cases, they want to manage public services that were often acquired by lines of business (LOBs) outside the infrastructure and operations (I&O) organization and have become difficult to manage.

- Vendors are addressing the key issues enterprises face. Many vendors are looking to combine cost management and security functionality into governance tooling. A few vendors are also looking to provide infrastructure as code (IaC) assistance by overlaying cloud management functionality to this capability.

- In addition, many vendors have added container management to their CMP offerings. The ability to serve both application developers and I&O personas is key. This requires CMPs to be linked into the application development process to enable I&O teams to enforce provisioning standards, without imposing a workflow that inhibits agility.

- Desirable IT outcomes include: policy enforcement; reduced lock-in to public cloud providers, although at the cost of CMP vendor lock-in, which can slow innovation; enhanced ability to broker services from various cloud providers and make informed business decisions as to the 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); and movement of complex workloads into the cloud that require best-in-class capabilities from multiple public cloud providers resulting in a multicloud adoption model.

Obstacles

- **Market evolution.** The CMP market continues to change, with vendors and enterprise customers getting a better feel for where such tooling can and cannot be used.

- **Evolving customer requirements.** Challenges vendors face include interfacing with multiple public clouds, cost transparency with workload optimization to remediate cost overruns, handling newer functions (e.g., containers and serverless deployments), and edge computing.

- **Market consolidation.** Many CMP vendors are acquiring cost management vendors and incorporating their functionality into their products. Vendors in adjacent markets are acquiring CMP vendors and combining this functionality with asset management and SaaS operational management. Cloud service providers (CSPs) and management service providers (MSPs) are entering the market, and many long-standing vendors have introduced next-generation products that target gaps in their previous products.

User Recommendations

- Weigh your full vertical (infrastructure as a service [IaaS], platform as a service [PaaS] and SaaS) and horizontal (on-premises, public cloud and edge) needs.
- Choose between native cloud services and CMPs by identifying a preferred provider model. Favor native cloud services if you value depth with a cloud provider; choose CMPs you want breadth across cloud providers.
- Determine the utility of functionally focused tools by defining the organization's functionality needs. Integrate cloud management or traditional management tools, because no vendor has a total cloud management solution.
- Ensure staffing to operate CMP platforms by planning new roles (e.g., cloud architects).
- Develop skills in financial and capacity management.
- Enable aggregation, integration, customization and governance for multicloud adoption models and environments by establishing a unified consumption layer via the Multicloud Management Platform (MCMP) or the Material Cost Management System (MCMS).

Sample Vendors

CloudBolt; Flexera; Morpheus Data; Scalr; Snow Software; VMware

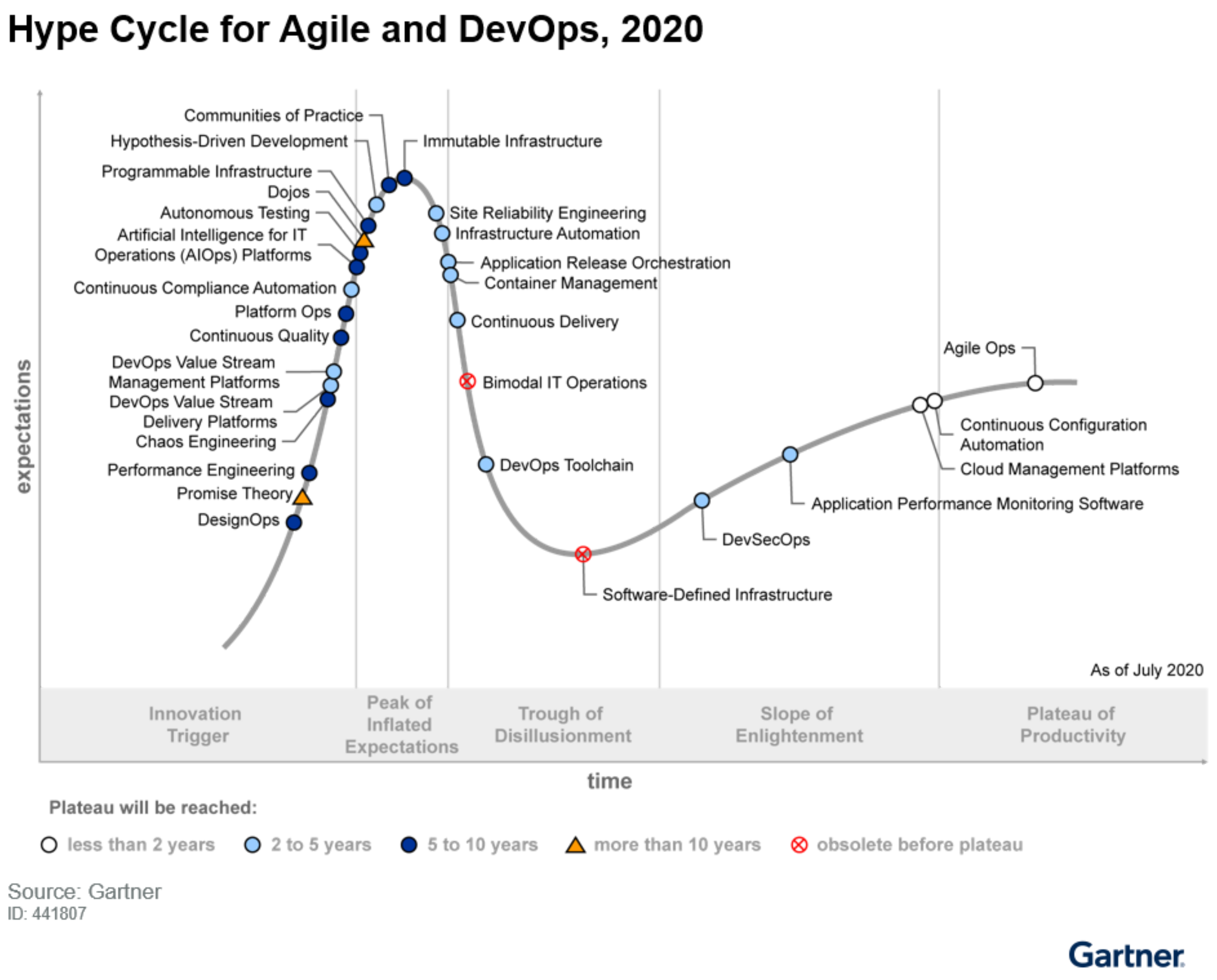
Gartner Recommended Reading

Market Guide for Cloud Management Tooling

Market Guide for Container Management

Appendixes

Figure 2: Hype Cycle for Agile and DevOps, 2020



Hype Cycle Phases, Benefit Ratings and Maturity Levels

Table 2: Hype Cycle Phases

(Enlarged table in Appendix)

Phase ↓	*Definition ↓*
Innovation Trigger	A breakthrough, public demonstration, product launch or other event generates significant media 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 innovation is pushed to its limits. The only enterprises making money are conference organizers and content publishers.
Trough of Disillusionment	Because the innovation 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 innovation'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 innovation 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 innovation to reach the Plateau of Productivity.

Source: Gartner (July 2021)

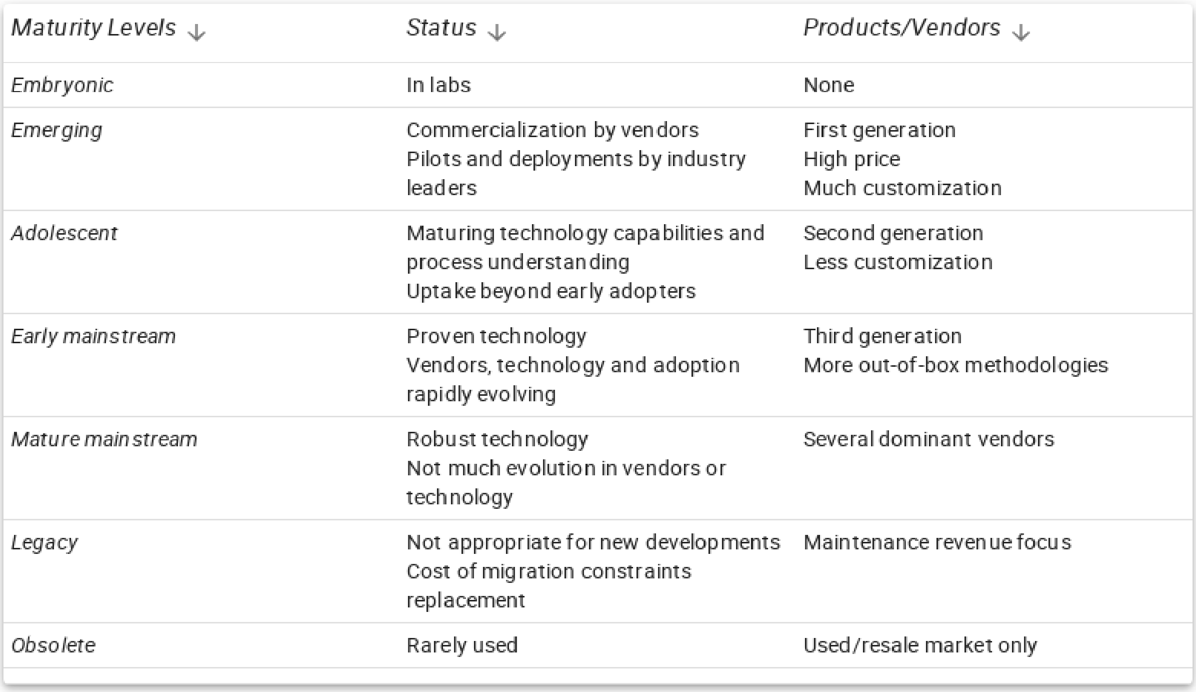
Table 3: Benefit Ratings

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 (e.g., improved user experience) that will be difficult to translate into increased revenue or cost savings.

Source: Gartner (July 2021)

Table 4: Maturity Levels

(Enlarged table in Appendix)



Maturity Levels ↓	*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 constraints replacement	Maintenance revenue focus
Obsolete	Rarely used	Used/resale market only

Source: Gartner (July 2021)

Acronym Key and Glossary Terms

AIOps	Artificial Intelligence for IT Operations (AIOps)
APM	Application Performance Monitoring
ARO	Application Release Orchestration
SDI	Software Defined Infrastructure
VSDP	Value Stream Delivery Platform
VSMP	Value Stream Management Platform

Document Revision History

Hype Cycle for Agile and DevOps, 2020 - 15 July 2020

Hype Cycle for DevOps, 2019 - 17 July 2019

Hype Cycle for DevOps, 2018 - 24 July 2018

Hype Cycle for DevOps, 2017 - 14 July 2017

Hype Cycle for DevOps, 2016 - 26 July 2016

Recommended by the Authors

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

Why DevOps Success Requires Platform Teams

2021-2023 Emerging Technology Roadmap for Large Enterprises

Market Guide for DevOps Value Stream Management Platforms

Market Guide for DevOps Value Stream Delivery Platforms

How to Build a Collaborative DevOps Culture

Table 1: Priority Matrix for Agile and DevOps, 2021

Benefit	*Years to Mainstream Adoption*			
↓	***Less Than 2 Years*** ↓	***2 - 5 Years*** ↓	***5 - 10 Years*** ↓	***More Than 10 Years*** ↓
Transformational		DevSecOps Observability Site Reliability Engineering	AIOps Platforms Digital Platform Conductor Tools Platform Ops	
High	Continuous Configuration Automation DevOps Toolchain	APM ARO Browser-Based IDEs Communities of Practice Container Management Continuous Delivery DesignOps DevOps VSDPs DevOps VSMPs GitOps Infrastructure Automation Policy-as-Code Programmable Infrastructure	Autonomous Testing Chaos Engineering Continuous Quality Dojos Performance Engineering	Promise Theory

Benefit	*Years to Mainstream Adoption*			
↓	***Less Than 2 Years*** ↓	***2 - 5 Years*** ↓	***5 - 10 Years*** ↓	***More Than 10 Years*** ↓
Moderate		Continuous Compliance Automation Hypothesis-Driven Development	DevOps Test Data Management Immutable Infrastructure	
Low	Cloud Management Platforms			

Source: Gartner (July 2021)

Table 2: Hype Cycle Phases

Phase ↓	*Definition* ↓
Innovation Trigger	A breakthrough, public demonstration, product launch or other event generates significant media 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 innovation is pushed to its limits. The only enterprises making money are conference organizers and content publishers.
Trough of Disillusionment	Because the innovation 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 innovation'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 innovation 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 innovation to reach the Plateau of Productivity.

Phase ↓	*Definition* ↓

Source: Gartner (July 2021)

Table 3: 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 (e.g., improved user experience) that will be difficult to translate into increased revenue or cost savings.

Source: Gartner (July 2021)

Table 4: Maturity Levels

Maturity Levels ↓	*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 constraints replacement	Maintenance revenue focus
Obsolete	Rarely used	Used/resale market only

Source: Gartner (July 2021)