# Hype Cycle for Open-Source Software, 2021

Published 26 July 2021 - ID G00747504 - 60 min read

By Analyst(s): Arun Batchu, Anne Thomas, Mark Driver

Initiatives: Software Engineering Technologies; Applications and Software Engineering Leaders; Software Engineering Strategies

Much of the world's software infrastructure is now based on open-source software. Applications and software engineering leaders should use this Hype Cycle to track innovations that facilitate the use of, or are powered by, open-source software.

## Strategic Planning Assumptions

Through 2025, more than 70% of enterprises will increase their IT spending on open-source software, compared with their current IT spending.

By 2025, software as a service will become the preferred consumption model for OSS due to its ability to deliver better operational simplicity, security and scalability.

By 2025, 75% of application development teams will implement software composition analysis tools in their workflow, up from 40% today, in order to minimize the security and licensing risks associated with open-source software.

By 2025, 20% of software engineering teams will adopt innersource practices, up from 2% today.

# Analysis

## What You Need to Know

Open-source software (OSS) is big business. Large and small vendors sponsor the most successful OSS projects, although hoards of projects are still managed by individuals and small teams. Much of the world's software depends on OSS infrastructure, libraries and frameworks. OSS is also where innovation happens across all software segments. The preponderance of OSS makes an enterprise OSS strategy essential, leading many organizations to establish an open-source program office (OSPO).

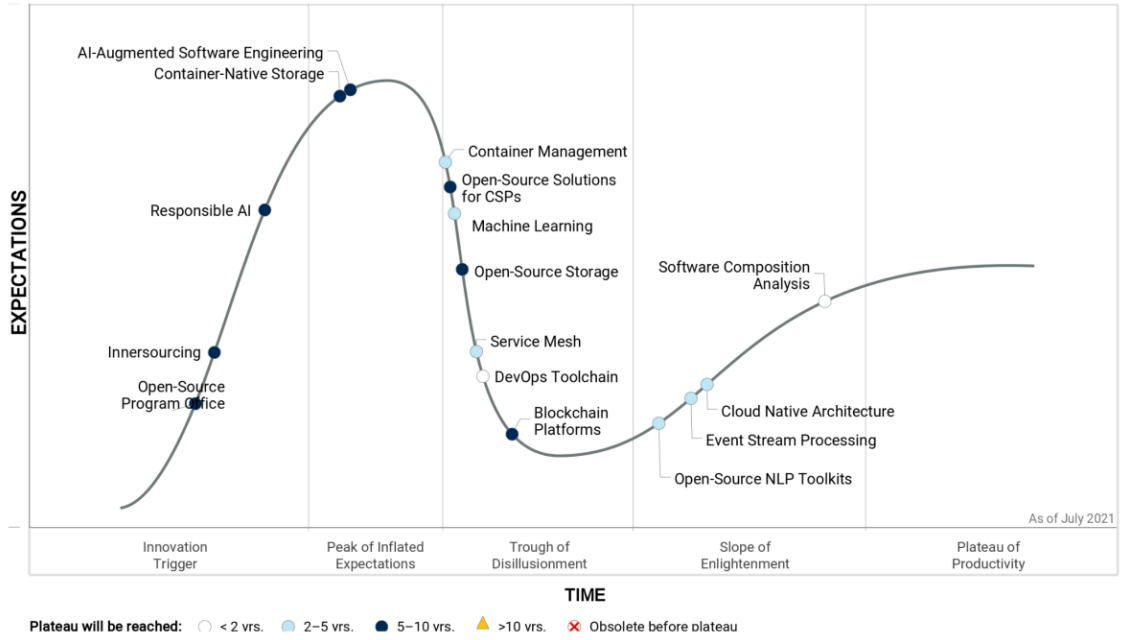
But the legal and management risks associated with OSS are increasing in quantity, severity and complexity. OSS maintainers are often overworked and underpaid, leading to increased risk of unmanaged OSS within the enterprise. Applications and software engineering leaders should champion their enterprises to consume OSS innovations, but to do so safely. Leaders must also figure out ways for their teams to contribute back software, and for their organizations to pay the maintainers, to avoid disruption to their enterprise's software supply chain.

## The Hype Cycle

OSS is software that is licensed under terms that grant everybody the rights to use, study, change and share the software and its source code in both modified and unmodified form. OSS usage has become nearly universal. A Gartner survey two years ago showed that OSS was used by over 90% of enterprises, and Gartner inquiry trends indicate that this percentage has grown in the years since then. [1] A 2020 Gartner survey showed that 75% of successful digital businesses used cloud-native OSS stacks and OSS-powered cloud services to build their digital platforms. [2] In 2020, 60 million new OSS repositories were created on GitHub by 56 million developers on the platform. [3] The number of OSS components in an average application has more than doubled in the last five years to 528; yet the proportion of vulnerabilities and license noncompliance have also increased at an alarming rate. [4] Applications and software engineering leaders are actively developing their OSS strategies — establishing an OSPO, crafting OSS governance policies, using OSS components in their software supply chains and adopting new OSS innovations. This is especially the case in areas such as cloud-native architecture, DevOps tooling, artificial intelligence, event-stream processing and blockchain. Some software engineering teams are adopting innersource strategies, bringing the principles of OSS development into in-house development.

The COVID-19 pandemic has driven renewed interest in OSS, primarily in search of cost-cutting strategies. While OSS reduces licenses fees, it doesn't always reduce total cost of ownership, because support subscriptions are often comparable in price to license subscriptions. Organizations that opt out of support subscriptions must bear the risks and costs of self-support. Software composition analysis (SCA) tools reduce the risks of OSS by providing visibility into the software bill of materials. These tools are expected to mature in two years or less.

## Figure 1: Hype Cycle for Open-Source Software, 2021




Source: Gartner (July 2021)

## Downloadable graphic: Hype Cycle for Open-Source Software, 2021

## The Priority Matrix

The Priority Matrix maps the benefit rating for each innovation against the amount of time each innovation requires to achieve mainstream adoption. The benefit rating provides an indicator of the potential of the innovation, but the rating may not apply to all industries and organizations. Identify which of the innovations offer significant potential benefits to your organization based on your own use cases.

Some of the technology categories listed on this Hype Cycle are not exclusively offered as open source, although much of the innovation in each of these categories is happening in open-source communities. Evaluate the value these innovations can bring to your organization based on the technology capabilities, rather than the licensing model.

Software engineering teams that adopt OSS should ensure that they do so safely. An OSPO can be a catalyst for open-source innovation and governance. SCA tools enable teams to manage open-source code more effectively. Innersource practices can help engineering teams improve the quality and security of their products.

### Table 1: Priority Matrix for Open-Source Software, 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 | | Event Stream Processing<br>Machine Learning | AI-Augmented Software Engineering<br>Blockchain Platforms<br>Responsible AI | |
| High | DevOps Toolchain<br>Software Composition Analysis | Container Management<br>Open-Source NLP Toolkits | Container-Native Storage<br>Open-Source Program Office<br>Open-Source Solutions for CSPs | |
| Moderate | | Cloud Native Architecture<br>Service Mesh | Innersourcing<br>Open-Source Storage | |
| Low | | | | |

Source: Gartner (July 2021)

## On the Rise

### Open-Source Program Office

**Analysis By:** Fintan Ryan, Arun Chandrasekaran, Arun Batchu, Anne Thomas, Mark Driver

**Benefit Rating:** High

**Market Penetration:** 1% to 5% of target audience

**Maturity:** Emerging

**Definition:**

An Open-Source Program Office (OSPO) is a central focus point for an organization's work with open source. The OSPO creates and evolves an open source strategy with input from various leaders that clearly and succinctly identifies the benefits, risks and policies governing it.

**Why This Is Important**

Gartner estimates that open-source software (OSS) is used in at least 90% of enterprises. OSS is a foundational technology component. Gartner's 2020 Building Digital Platforms Survey found that organizations building digital platforms heavily used OSS, including vendor-supported open source (see How to Build a Digital Business Technology Platform). Organizations such as Goldman Sachs, Toyota and Bloomberg have established OSPOs to manage the usage of OSS (see TODO's OSPO Landscape).

**Business Impact**

An OSPO is key to an enterprise open-source strategy. OSPOs recognize open source's strategic importance, enforce effective governance policies, facilitate contributions and communicate open source's value to stakeholders. Embracing open source has a positive effect on employee retention. Contributing to, or maintaining, an open-source project is seen as a positive career driver. An OSPO builds links with the open-source community at large, helping to build a pool for potential recruitment.

**Drivers**

- OSS is ubiquitous, but enterprises often have limited visibility into where and how OSS is used within their technology stack.

- Software developers wish to contribute to open-source projects that they use within their day-to-day work. Actively supporting contributions to OSS projects helps talent acquisition and retention.
- Poorly governed use of OSS exposes an enterprise to legal risks. For example, ungoverned use of OSS components may violate licensing terms or infringe upon intellectual property rights.
- Establishing the source and provenance of software components is essential as legal requirements for software bills of materials grow (see the Executive Order on Improving the Nation's Cybersecurity on the White House website).

### Obstacles

- Creating an effective OSPO requires sponsorship from senior leadership.
- Policies created by the OSPO risk becoming heavyweight. Software development teams will avoid heavyweight policies.
- A lack of self-service tooling and automation can delay the software development life cycle. Policies related to consumption and publication set by the OSPO must be automatically applied.

### User Recommendations

- Establish an OSPO with responsibility for governance, technology strategy and communication around the use of OSS.
- Create an open-source strategy document that identifies the organization's goals and objectives for using OSS.
- Define an open-source governance policy that delineates where and how OSS can be consumed, provides an overview of licenses and usage policies, and sets out associated risks.
- Become active members of open-source foundations, such as TODO.
- Make the processes for contributing to open-source projects as frictionless as possible.
- Use software composition analysis to establish accurate software bills of materials.
- Create an OSS advisory council consisting of the OSPO and representatives from security, legal and technical teams.

- Build governance tools into your automation pipelines to enforce OSPO policies (e.g., license compliance) on the consumption of open-source software.

### Sample Vendors

TODO

### Gartner Recommended Reading

A CTO's Guide to Top Practices for Open-Source Software

Ensure Safe and Successful Usage of Open-Source Software With a Comprehensive Governance Policy

Tool: Open-Source Software Governance Policy Template

How to Manage Open-Source Software Risks Using Software Composition Analysis

## Innersourcing

**Analysis By:** Arun Chandrasekaran, Mark Driver, Anne Thomas, Arun Batchu, Fintan Ryan, Mark O'Neill

**Benefit Rating:** Moderate

**Market Penetration:** 1% to 5% of target audience

**Maturity:** Emerging

### Definition:

Innersourcing is the practice of bringing the open-source software principles of collaboration and openness into the organization for any form of software development.

### Why This Is Important

Gartner has seen client inquiries on innersourcing increase 20% year over year. This coincides with a renewed interest in open source, in part driven by the COVID-19 pandemic. Organizations such as Comcast have been successful in adopting an innersourcing model (see the Comcast case study in A CTO's Guide to Top Practices for Open-Source Software). This has driven the interest of other organizations to similarly apply open-source practices internally in their organizations.

### Business Impact

- Innersourcing encourages strategic thinking of software usage rather than trying to solve tactical problems.
- Organizations can use innersourcing to more effectively align with external open-source communities and foundations to tap into their knowledge.
- Embracing open-source principles can encourage employee retention through recognition of contributions.
- Applications created in an innersourcing model are suitable for being released as publicly available open-source projects.

### Drivers

- Automated software deployment is a driver for innersourcing, meaning that DevOps adoption is a prerequisite for an innersouring program.
- The need for communication and collaboration between disparate software development teams within organizations drives the need for innersourcing.
- Software developers are now overwhelmingly familiar with the practices of open source, including through interacting with code repositories, using well-known open-source tools and even, in some cases, through contributing to open-source projects. This creates the foundation for the effective use of open-source principles inside organizations.
- Code quality can be improved through the usage of automated quality and consistency checking, which is often part of the technical infrastructure of an innersourcing effort, often via the work of an open-source program office (OSPO).
- Increased cross-team collaboration and reuse of functionality, as code libraries or via APIs, are also key drivers of innersourcing initiatives.

### Obstacles

- Creating an effective innersourcing program requires sponsorship from senior leadership.
- Governance of innersouring is a challenge, but can be met through establishing an open-source program office and communities of practice.
- Many organizations do not yet have self-service access to deploy code, or access APIs or shared libraries.

### User Recommendations

- Emphasize adequate documentation, cross-team collaboration and code contribution to other internal teams' repositories.
- Create smaller and flatter product teams with an emphasis on egalitarian decision making and constant communication on product direction.
- Establish and communicate best practices on coding standards, preferred tooling and common repositories, and code release process.
- Enable self-service access to infrastructure as a service (IaaS), and automate the build, test and release pipelines.
- Emphasize documentation and a simplified on-boarding experience for newcomers.
- Conduct hackathons to stimulate the creative process, and listen to developer feedback to remove bureaucracy.

### Gartner Recommended Reading

A CTO's Guide to Top Practices for Open-Source Software

Building a Platform for Product Team Productivity (adidas)

Create API Portals That Drive API Adoption Among Internal and External Developer Communities

### Responsible AI

**Analysis By:** Svetlana Sicular

**Benefit Rating:** Transformational

**Market Penetration:** 1% to 5% of target audience

**Maturity:** Emerging

**Definition:**

Responsible artificial intelligence is an umbrella term for aspects of making appropriate business and ethical choices when adopting AI that organizations often address independently. These include business and societal value, risk, trust, transparency, fairness, bias mitigation, explainability, accountability, safety, privacy and regulatory compliance. Responsible AI encompasses organizational responsibilities and practices that ensure positive, accountable AI development and exploitation.

**Why This Is Important**

Responsible AI has emerged as the top AI topic for Gartner clients. The more AI replaces human decisions at scale, the more it amplifies the positive and negative impacts of such decisions. Responsible AI pursues positive outcomes and prevents negative results by resolving dilemmas rooted in delivering value versus tolerating risks. Recently, many jurisdictions globally introduced new and pending AI regulations that challenge data and analytics leaders to respond in meaningful ways.

**Business Impact**

Responsible AI signifies the move toward accountability for AI development and use at the individual, organizational and societal levels. If AI governance is practiced by designated groups, responsible AI applies to everyone involved in the AI process. Responsible AI helps achieve fairness, even though biases are baked into the data; gain trust, although transparency and explainability methods are evolving; and ensure regulatory compliance, despite the AI's probabilistic nature.

**Drivers**

Responsible AI means a deliberate approach in many directions at once. Data science's responsibility to deliver unbiased, trusted and ethical AI is just the tip of the iceberg. Responsible AI helps AI participants develop, implement, exploit and resolve the dilemmas they face. Ideally, it enhances both sides at the following levels:

- **Organizational** — Resolving AI's business value versus risk in regulatory, business and ethical constraints. It could also include employee reskilling and intellectual property protection.

- **Societal** — Resolving AI effectiveness for societal well-being versus limiting human freedoms. Existing and pending legal guidelines and regulations, such the EU's Artificial Intelligence Act, make responsible AI a necessity.
- **Customer, citizen** — Resolving privacy versus convenience involves a thin line between customers' readiness to give their data in exchange for goods or benefits and customer/citizen concerns about their privacy. Fairness and ethics are the greatest drivers in this space. Regulations shed light on the necessary steps — for example, the U.S. Federal Trade Committee's "Using Artificial Intelligence and Algorithms" for consumer protection. However, this does not relieve organizations of deliberation specific to their constituents.
- **Workplace** — Resolving work efficiency versus employer "creepiness" includes concerns about AI's effect on jobs and employee morale, as well as change management.

AI affects all ways of life and touches all societal strata; hence, the responsible AI challenges are multifaceted and cannot be easily generalized. New problems constantly arise with rapidly evolving technologies and their uses, such as using generative AI for creating deepfakes. Most organizations combine some of the following drivers under the umbrella of responsible AI:

- Accountability
- Diversity
- Ethics
- Explainability
- Fairness
- Human centricity
- Operational responsibility
- Privacy
- Regulatory compliance
- Risk management
- Safety

- Transparency
- Trustworthiness

### Obstacles

- Unawareness of AI's unintended consequences prevails. Many organizations turn to responsible AI only after they hit AI's negative effects, whereas prevention is easier and less stressful.
- Legislative pace, uncertainty and complexity puts responsible AI on hold in many firms. It also leads to one-sided efforts for regulatory compliance, while ignoring other responsible AI drivers.
- Rapidly evolving AI technologies, including tools for explainability, bias detection, privacy protection, and some regulatory compliance, lull organizations into a false sense of responsibility, while mere technology is not enough. A disciplined AI ethics and governance approach that brings together multiple perspectives and diversity of opinions is necessary, in addition to technology.
- Poorly defined accountability and incentives for responsible AI practices make responsible AI look good on paper, but ineffective in reality.

### User Recommendations

Data and analytics leaders, take responsibility — it's not AI, it's you who are liable for the results and impacts, either intended or unintended.

- Combine the responsible AI aspects you currently address independently to promulgate consistent approaches across all focus areas. The most typical areas of responsible AI in the enterprise are fairness, bias mitigation, ethics, risk management, privacy and regulatory compliance.
- Designate a champion accountable for the responsible development of AI, for each use case.
- Raise awareness of AI differences from the familiar concepts continuously. Provide training and education on responsible AI, first to most critical personnel, and then to your entire AI audience.
- Establish an AI ethics board to resolve AI dilemmas. Ensure diversity of participants and the ease to voice AI concerns.

- Participate in industry or societal responsible AI groups. Learn best practices and contribute your own, because everybody will benefit from this.

### Sample Vendors

Google, H2O.ai, IBM, Microsoft, SAS, Tazi.ai

### Gartner Recommended Reading

Predicts 2021: Artificial Intelligence and Its Impact on People and Society

Top Trends in Data and Analytics for 2021: Smarter, More Responsible and Scalable AI

Cool Vendors in AI Governance and Ethical Response

Case Study: Ethical AI With an External Board (Axon)

What Non-Technology Executives Should Do in Support of Responsible AI Initiatives

Financial Services CIOs Must Focus AI Investments on 'Responsible AI' in 2021

## At the Peak

### AI-Augmented Software Engineering

**Analysis By:** Arun Batchu

**Benefit Rating:** Transformational

**Market Penetration:** 1% to 5% of target audience

**Maturity:** Emerging

#### Definition:

AI-augmented software engineering (AIASE) is the use of AI technologies such as machine learning (ML), natural language processing (NLP) and similar technologies to aid software engineering teams in creating and delivering applications faster, more consistently, and with higher quality. AIASE commonly integrates with an engineer's existing tools to provide them with real-time, intelligent feedback and suggestions.

#### Why This Is Important

Today's software engineering methods involve writing boilerplate code that saps expert engineers' creativity, limiting their productivity. Novice developers are challenged by the complexity of modern software they need to deliver. Moreover, engineers need to master multiple fit-for-purpose languages that are used to realize a complex software system. AIASE helps in resolving these issues, by acting as an intelligent assistant, a pair-programmer, an expert coach and quality control inspector.

#### Business Impact

AIASE is enabling creative business problem-solving by automating boilerplate software engineering tasks. It is increasing developer velocity by recommending highly relevant code and library recommendations in a fraction of the time it would take otherwise. It is augmenting quality and testing engineers by allowing tests to self-heal and by automatically creating tests. AIASE models continuously improve their utility by learning from its regular interactions with the engineers and environment.

#### Drivers

- Increasing complexity of software systems to be engineered.
- Increasing demand for developers to deliver high-quality code faster.
- Continuous modernization of existing systems via replacement or refactoring.

- Increasing impact of software development on business models.
- Development of language server protocols for easy integration into widely adopted integrated development environments.
- Application of deep learning language models to software code.
- Increase in usage of high-quality open source software as model-training data.
- Application of machine learning to models, used within model-driven development platforms.
- Increasing availability of GPUs and TPUs needed to run machine learning workloads in the cloud.
- Increasing computational power of developer machines, especially for running and training machine learning models.

### Obstacles

- Lack of deep understanding of generated solutions.
- Limited awareness of the existence of production-ready tools.
- Minimal evidence proving ROI of AIASE.
- Resistance from software engineers who fear job obsolescence
- Lack of transparency and provenance of data used for model training.
- Trade-off between relevance and latency of real-time suggestions.

### User Recommendations

- Make the engineering teams aware of, evaluate, and adopt, AIASE tools with a sense of urgency.
- Identify the programming languages within the enterprise best suited to start with.
- Evaluate funding needed to obtain necessary licenses for AIASE, and develop a business case to obtain them.
- Establish a method to measure productivity gains.

- Check the maintainability of AI-generated code, tests, and models ( in development platforms).
- Reassure their software engineers about AIASE augmenting — not replacing them.
- Track the rapidly evolving and highly impactful market to identify new products that minimize development toil and improve experience of software engineers.

**Sample Vendors**

Codota; Diffblue; IBM; Kite; Mendix; Microsoft; NerdVision; OutSystems; SeaLights; Tabnine

**Gartner Recommended Reading**

Emerging Technologies: Critical Insights Into AI-Augmented Software Development

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

**Container-Native Storage**

**Analysis By:** Julia Palmer, Arun Chandrasekaran

**Benefit Rating:** High

**Market Penetration:** 5% to 20% of target audience

**Maturity:** Emerging

**Definition:**

Container-native storage (CNS) is specifically designed to support container workloads and focus on addressing unique cloud-native scale, granularity and performance demands while providing deep integration with the container management systems. CNS is designed to be aligned with microservices architecture principles and adhere to the requirements of container-native data services, including being hardware-agnostic, API-driven, and based on distributed software architecture.

### Why This Is Important

Many container-based applications require support for stateful data. CNS solutions are being specifically designed to run cloud-native applications. The common foundation is typically based on a distributed, software-defined unified pool of storage and has container level granularity of data services. In addition, the entire stack is most often orchestrated with Kubernetes to manage container life cycle integration and enable self-service operations for developers.

### Business Impact

- CNS enables deployment of stateful cloud-native applications, which enhances elasticity, availability and hybrid cloud integration.
- I&O leaders require storage platforms that can adhere to principles of container-native infrastructure as they now support more stateful applications, require to share application data, and need to provide advanced data services.
- CNS eliminates the bottlenecks to achieving agility in the end-to-end process of building and deploying modern cloud native applications.

### Drivers

- As organizations devise a cloud strategy, they are building new workloads using cloud-native principles as well as rearchitecting traditional workloads on Kubernetes platforms, both of which are driving significant momentum in the adoption of CNS.
- Due to the increased popularity of deploying and operating container environments by orchestration platform, most IT leaders are now looking for a persistent storage solution that can be tightly integrated with container orchestrators, such as Kubernetes.
- I&O leaders require new tools and processes for data management in order to provide storage services accessed by stateful applications running in containers and orchestrated by Kubernetes.
- Kubernetes storage standardization is benefiting from the introduction and growing adoption of the Container Storage Interface, which is being promoted by the Cloud Native Computing Foundation.
- CNS solutions can be deployed on-premises or in the cloud, making them optimal for hybrid and multicloud deployment infrastructure. Since the CNS functions are based on software, they can be implemented in containers, enabling them to be managed with the same orchestration functions as containerized applications.

### Obstacles

- A CNS solution will not be adopted by every enterprise, as it remains most appropriate for new cloud-native applications, or for applications that will be revised with significant refactoring.
- Although embracing the CNS paradigm will yield agility benefits, adopting a CNS solution is likely to increase operational complexity in the short term for traditional enterprise environments.
- CNS vendor landscape and technology is constantly evolving and many vendors in this space are early-stage startups.
- Given the fragmented nature of the vendor ecosystem today, I&O leaders run the risk of creating a technology silo with CNS solutions, which is a common obstacle to large-scale adoption.

### User Recommendations

- Choose storage solutions aligned with microservices architecture principles and which adhere to the requirements of container-native data services, such as being hardware-agnostic, API-driven, based on distributed architecture, and capable of supporting edge, core or public cloud deployments.
- Select storage products closely aligned with the developer workflow tools that can be directly integrated with the application layer for portability, scaling and data protection.
- Validate your vendor's capability of continuous innovation delivery, quality customer support and a consistent pricing model, given that the container ecosystem is rapidly evolving with unproven vendor business models
- Find and review reference examples of using the storage orchestration system with commonly deployed stateful applications before you adopt a CNS solution for a particular application, such as Apache Cassandra, Apache Kafka, MySQL, MongoDB and PostgreSQL.

### Sample Vendors

Diamanti; MayaData; NetApp; Pure Storage; Red Hat; Robin.io; StorageOS; SUSE

### Gartner Recommended Reading

An I&O Leader's Guide to Storage for Containerized Workloads

Solution Path for Implementing Containers and Kubernetes

Market Guide for Container Management

## 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

**Open-Source Solutions for CSPs**

**Analysis By:** Mentor Cana

**Benefit Rating:** High

**Market Penetration:** 20% to 50% of target audience

**Maturity:** Early mainstream

**Definition:**

Open source is a model for the development, support and distribution of software that encourages and, in many ways, enforces community stewardship of the technology. Communications service provider (CSP) CIOs and CTOs are increasingly focused on deploying open-source solutions. The adoption of open-source software continues to disrupt CSPs' IT and network operations ecosystems, yet enables CSPs to achieve better interoperability and flexibility in the long term.

### Why This Is Important

Because open-source adoption can help reduce vendor lock-in and provide greater control over technology change management, CSPs are embracing open source in their network and IT operations. While the CSP IT operations domain has been embracing open-source software over many years, the adoption of open source in the CSP network domain has picked up among leading global CSPs. Gartner expects that it will further grow over the next few years.

### Business Impact

Open-source solutions provide CSPs with the following major benefits:

- Lower cost of acquisition of technologies and capabilities
- Increased agility with faster time to market
- Accelerated innovation cycle
- Diversifying sourcing mechanisms
- Reducing vendor dependence
- Developing openness in infrastructures and applications
- Easier partnering with ecosystems participants

### Drivers

- Open source presents the single biggest opportunity for CSPs to begin the transition from vendor-locked operations' ecosystems to vendor-agnostic operations' ecosystems. Both, use of open-source and contribution to open-source initiatives, are steps toward vendor independence for CSPs. This allows CSPs to leverage wider innovation, enhance vendor neutrality and gain better control over their technology change management.
- CSPs are increasingly looking to reduce their solution development-, implementation- and support-related costs. In the past, internal research and development has proved to be expensive, time-consuming and inefficient, yielding only mediocre results. However, because the most successful open-source software solution developments are driven by collaborative communities (with active participation by large CSPs) that have hundreds of thousands of contributors around the world, CSPs expect to reap the benefits of community-based innovation and development.
- Open source in the network function virtualization (NFV) management and orchestration (MANO) component is growing rapidly. Examples include open-source solutions, such as Open Network Automation Platform (ONAP), open-source MANO (OSM), and open radio access network and/or physical access. In addition, DevOps, continuous integration/continuous delivery (CI/CD) and cloud-native adoption has also fueled open-source tools adoption. As CSPs accelerate the effort to virtualize their network functions so that they can automate and orchestrate service delivery without specific vendor lockings, the adoption of open-source solutions enables this flexibility as well as speed for CSPs.
- This trend of adopting and operationalizing open-source solutions is expected to continue for the next three to five years.

### Obstacles

- Successful adoption and solution delivery using open-source requires building strong in-house architecture and development teams that can be sustained over the long term.
- General lack of maturity and performance (when compared to proprietary or vendor-led platforms), which can, on occasion, be inconsistent.
- Operational support for open source can prove challenging and is often underestimated by adopters.
- Development, engineering and operational resources can be hard to attain.

### User Recommendations

- Encourage the use of open-source solutions by making it mandatory to have open-source solutions in technology roadmaps for areas of the network where open source can provide the desired outcomes.
- Incorporate open-source solutions in architecture, roadmap and timeline by methodically replacing various vendor-locked proprietary operational software elements with open-source alternatives, where possible.
- Force the development of open-source alternatives by eliminating or severely restricting the funding for vendor-locked proprietary solutions.
- Prioritize the development of open-source software solutions by focusing on enhancing its reliability and implementing internal support processes.
- Promote innovation internally and in the CSP vendor landscape by taking advantage of the open-source ecosystem, especially with vendors that use open source. Invest in programs to instill the culture of innovation.
- Build open-source competencies by hiring new talent to address the needs of open-source solutions, both for development and support.

### Sample Vendors

Kubernetes; ONAP; Open Compute Project; OpenDaylight; Open Networking Foundation (ONF); OpenStack; O-RAN ALLIANCE; Open Source MANO; Xen

### Gartner Recommended Reading

Hype Cycle for Open-Source Software, 2020

Open-Source Change: Navigating Change in the Digital Era

Peer Connect Perspectives: Developing an Open-Source Strategy

OSS Solution Sourcing Criteria Changes for CSP Operations Transformation

## Machine Learning

**Analysis By:** Farhan Choudhary, Carlie Idoine, Shubhangi Vashisth

**Benefit Rating:** Transformational

**Market Penetration:** 5% to 20% of target audience

**Maturity:** Adolescent

**Definition:**

Machine learning is an AI discipline that solves business problems by utilizing statistical models to extract knowledge and patterns from data. There are three major approaches that relate to the types of observation provided. These are supervised learning, where observations contain input/output pairs (also known as “labeled data”); unsupervised learning (where labels are omitted); and reinforcement learning (where evaluations are given of how good or bad a situation is).

**Why This Is Important**

According to Gartner’s 2019 AI in Organizations survey, machine learning (ML) is the AI initiative for which more POCs and production systems are conducted. Over the past few years, ML has gained a lot of traction because it helps organizations to make better decisions at scale with the data they have. ML aims to eliminate traditional trial-and-error approaches based on static analysis of data, which is often inaccurate and unreliable, by generalizing knowledge from data.

**Business Impact**

Machine learning drives improvements and new solutions to business problems across a vast array of business, consumer and social scenarios like:

- Automation
- Price optimization
- Customer engagement
- Supply chain optimization
- Predictive maintenance
- Fraud detection

Machine learning impacts can be explicit or implicit. Explicit impacts result from machine learning initiatives. Implicit impacts result from products and solutions that you use without realizing they contain machine learning.

**Drivers**

- As organizations continue to adopt these technologies, we recently see focus on aspects that relate to ML explainability and operationalization. Augmentation and automation (of parts) of the ML development process improve productivity of data scientists and enable citizen data scientists in making ML pervasive across the enterprise.
- In addition, pretrained ML models are increasingly available through cloud service APIs, often focused on specific domains or industries.
- Data science and machine learning education is becoming a standard at many academic institutions, therefore fueling the supply of newer talent eager to venture into this space.
- There's always active research in the area of machine learning in different industries — manufacturing, healthcare, corporate legal, defense and intelligence. Thus, its applicability is far and wide.
- Newer learning techniques such as zero, one, few or end shot learning are emerging that take away the burden of having high volumes of quality training data for ML initiatives. This lowers the barrier to entry and experimentation for organizations.
- New frontiers are being explored in synthetic data, new algorithms (e.g., deep learning variations) and new types of learning. These include federated/collaborative, generative adversarial, transfer, adaptive and self-supervised learning, all aiming to broaden ML adoption.

### Obstacles

- The triggers of its massive growth and adoption have been growing volumes of data, advancements in compute infrastructure and the complexities that conventional engineering approaches are unable to handle.
- Even though ML is one of the particularly popular AI initiatives in the last few years, it is not the only one. Organizations also tend to rely on other AI techniques such as rule-based engines, optimization techniques, physical models to achieve decision augmentation or automation.
- A significant portion of ML models at an organization doesn't make it into production, therefore adding to technical debt and risks mistrust in the initiative, often delaying value realization from ML at organizations.
- The application of ML is often oversimplified as just model development but it's not so. Several dependencies which are overlooked, such as data quality, security, legal compliance, ethical and fair use of data, serving infrastructure, and so forth, have to be considered in ML initiatives.

### User Recommendations

- Build up and extend descriptive analysis toward predictive and prescriptive insights, which can be excellent candidates for machine learning.
- Assemble a (virtual) team that prioritizes machine learning use cases, and establish a governance process to progress the most valuable use cases through to production.
- Utilize packaged applications if you find one that suits your use case requirements. These often can provide superb cost-time-risk trade-offs and significantly lower the skills barrier.
- Explicitly manage MLOps and ModelOps for deploying, integrating and monitoring analytical, ML and AI models.
- Adjust your data management and information governance strategies to enable your ML team. Data is your unique competitive differentiator, and adequate data quality, such as the representativeness of historical data for current market conditions, is critical for the success of ML.

### Sample Vendors

Amazon Web Services (AWS); Databricks; Dataiku; DataRobot; Domino; Google Cloud Vertex AI; H2O.ai; Microsoft Azure; SAS; TIBCO Software

### Gartner Recommended Reading

Magic Quadrant for Data Science and Machine Learning Platforms

Critical Capabilities for Data Science and Machine Learning Platforms

Toolkit: RFP for Data Science and Machine Learning Platforms

3 Types of Machine Learning for the Enterprise

Understanding MLOps to Operationalize Machine Learning Projects

## Open-Source Storage

**Analysis By:** Julia Palmer, Arun Chandrasekaran

**Benefit Rating:** Moderate

**Market Penetration:** 5% to 20% of target audience

**Maturity:** Adolescent

### Definition:

Open-source storage (OSS) is a form of software-defined storage for which the source code is made available to the public through a distribution license that complies with open source definition. Open-source storage supports many of the same features as proprietary storage, including support of structured and unstructured data, as well as heterogeneous management.

### Why This Is Important

Although open-source storage has been around for over a decade, it has been adopted mainly by hyperscalers, technical service providers and large organizations. Recent innovations in x86 hardware and flash, combined with an innovative open-source ecosystem, are making open-source storage and its licensing models attractive for cloud and big data workloads and as a potential alternative to proprietary storage.

### Business Impact

- Gartner is seeing adoption among technology firms' service provider clients, as well as in research and academic environments. Big data, analytics, dev/test and private cloud use in enterprises are also promising use cases for open-source storage.
- As data continues to grow at a frantic pace, open-source storage will enable customers to store and maintain data, particularly unstructured data, at a lower acquisition cost, with "good enough" availability, performance and manageability.

### Drivers

- Open-source storage is playing an important role in enabling cost-effective, scalable platforms for new cloud and big data workloads. The COVID-19 pandemic has accelerated the consideration of more cost-effective storage products.
- Today, Gartner clients are actively evaluating open-source storage across block, file and object protocols. More than 90% of enterprises worldwide use OSS in support of their mission-critical IT workloads, whether they are aware of it or not. This has led to more acceptance of open-source options across the technology stack.
- Cloud computing, microservices application architectures, big data analytics and information archiving push the capacity, pricing and performance frontiers of traditional scale-up storage architectures. This has led to a renewed interest in open-source software as a means to achieve high scalability in capacity and performance at lower acquisition costs.
- The emergence of open-source platforms such as Kubernetes and TensorFlow are backed by large, innovative communities of developers and vendors, together with vendors such as Red Hat (Gluster Storage, Ceph Storage) and SUSE (Ceph) and DDN (Lustre). Collectively, they provide enterprises with a broad selection of options to consider for use cases such as cloud storage, big data, stateful microservices workloads and archiving.
- There is also an influx of open-source storage projects for container-based storage such as MinIO, OpenEBS, Longhorn and Rook.

### Obstacles

- Onboarding open-source storage software will require more resources as some IT leaders often overestimate the benefits and underestimate the costs and risks. Although open-source storage offers a less-expensive upfront alternative to proprietary storage, IT leaders need to weigh the benefits, risks and costs accurately.
- It is difficult to predict cost and ROI of ownership of open-source storage. Pragmatic long-term open source investment strategy must include a balance of all three factors (cost savings, flexibility and innovation) to yield successful results.
- For sufficiently mature OSS projects, I&O leaders may turn to the community as a knowledge base to augment their self-support efforts. Unfortunately, these communities do not come with a contracted service-level agreement, and there is no guarantee for quick and reliable support.

### User Recommendations

- With the emerging maturity of open-source storage solutions, enterprise IT buyers should not overlook the value proposition of these solutions. Allocate resources to invest in participating and contributing to OSS initiatives to support ecosystem activities.
- IT leaders should actively deploy pilot projects, identify internal champions, train storage teams and prepare the overall organization for this disruptive trend.
- Although source code can be downloaded for free, it is advisable to use a commercial distribution and to obtain support through a vendor, because OSS requires significant effort and expertise to install, maintain and support.
- IT leaders deploying “open core” or “freemium” storage products need to carefully evaluate any downsides of lock-in against the perceived benefits attained. The proprietary software version often comes in the form of add-on modules, retained features or management tools that function on top of OSS.

### Sample Vendors

DDN; iXsystems; MayaData; MinIO; Rancher Labs; Red Hat; SoftIron; SUSE

### Gartner Recommended Reading

A CTO’s Guide to Top Practices for Open-Source Software

Quick Answer: Is Open Source Software More or Less Secure Than Proprietary Software?

Quick Answer: Has Vendor Pressure Changed The Definition of Open Source In 2021?

Peer Connect Perspectives: Developing an Open-Source Strategy

### Service Mesh

**Analysis By:** Anne Thomas

**Benefit Rating:** Moderate

**Market Penetration:** 1% to 5% of target audience

**Maturity:** Adolescent

**Definition:**

A service mesh is a distributed computing middleware that optimizes communications between application services within managed container systems. It provides lightweight mediation for service-to-service communications, and supports functions such as authentication, authorization, encryption, service discovery, request routing, load balancing, self-healing recovery and service instrumentation.

**Why This Is Important**

A service mesh is lightweight middleware for managing and monitoring service-to-service (east-west) communications, especially among microservices running in ephemeral managed container systems, such as Kubernetes. It provides visibility into service interactions, enabling proactive operations and faster diagnostics. It automates complex communication concerns, thereby improving developer productivity and ensuring that certain standards and policies are enforced consistently across applications.

**Business Impact**

- A service mesh helps ensure resilient and secure request-response communication between services deployed in Kubernetes and other managed container systems.
- Service mesh middleware is one of many management technologies that provide software infrastructure for distributed applications deployed in managed container systems.
- This type of middleware, along with other management and security middleware, helps provide a stable environment that supports “Day 2” operations of containerized workloads.

### Drivers

- Service mesh adoption is closely aligned with microservices architectures and managed container systems like Kubernetes. Service mesh supports needed functionality in ephemeral environments, such as service discovery and mutual Transport Layer Security between services.
- As microservice deployments scale and grow more complex, DevOps teams need better ways to track operations, anticipate problems and trace errors. Service mesh automatically instruments the services and feeds logs to visualization dashboards.
- A service mesh implements the various communication stability patterns (including retries, circuit breakers and bulkheads) that enable applications to be more self-healing.
- Many managed container systems now include a service mesh, inspiring DevOps teams to use it. The hyperscale cloud vendors provide a service mesh that is also integrated with their other cloud-native services.
- Independent vendors, such as Buoyant, HashiCorp and Kong provide service meshes that support multiple environments.

### Obstacles

- Service mesh technology is immature and complex, and most development teams don’t need it. It can be useful when deploying microservices in Kubernetes, but it’s never required.
- Users are confused by the overlap in functionality among service meshes, ingress controllers, API gateways and other API proxies. Management and interoperability among these technologies hasn’t yet been addressed by the vendor community.
- Many people associate service mesh exclusively with Istio, even though it isn’t the most mature product in the market and has a reputation for complexity.
- Independent service mesh solutions face challenges from the availability of platform-integrated service meshes from the major cloud and platform providers.

### User Recommendations

- Delay adoption of service mesh until your teams start building applications that will get value from a mesh, such as applications deployed in managed container systems with a large number of service-to-service (east-west) interactions.
- Favor the service meshes that come integrated with your managed container system unless you have a requirement to support a federated model.
- Reduce cultural issues and turf wars by assigning service mesh ownership to a cross-functional PlatformOps team that solicits input and collaborates with networking, security and development teams.
- Accelerate knowledge transfer and consistent application of security policies by collaborating with I&O and security teams that manage existing API gateways and application delivery controllers.

### Sample Vendors

Amazon Web Services; Buoyant; Decipher Technology Studios; Envoy; F5; Google; HashiCorp; Istio; Kong; Microsoft; Red Hat; Solo.io; Tetrate; VMware

### Gartner Recommended Reading

How a Service Mesh Fits Into Your API Mediation Strategy

Assessing Service Mesh for Use in Microservices Architectures

Emerging Technology Analysis: Service Mesh

### 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)

### Blockchain Platforms

**Analysis By:** Rajesh Kandaswamy, David Furlonger

**Benefit Rating:** Transformational

**Market Penetration:** 5% to 20% of target audience

**Maturity:** Adolescent

#### Definition:

Blockchain platforms provide the foundation to create and run blockchain solutions and decentralized networks. This includes support for distributed ledgers, decentralized consensus, tokenization and smart contracts. They enable creation of blockchain solutions that provide immutability, transparency, decentralized contract execution, and tokenization of physical or digital assets.

#### Why This Is Important

Blockchain platforms are the foundation on which blockchain applications are built and managed. The key aspects of blockchain — such as a distributed ledger, immutability, transparency, tokenization and support for smart contracts — are implemented through use of a blockchain platform. They also provide supporting services, such as developer languages, performance tuning and monitoring, security services and support for blockchain interoperability.

#### Business Impact

Blockchain platforms can play a key role in creating blockchain solutions and networks that change the commercial terms and governance structures of business and society. This is achieved by decentralizing and automating digital business, offering capabilities that support and foster autonomous, peer-to-peer (P2P) transactions, diverse value-exchange scenarios, on-demand markets, smart assets, smart contracts and the operation of decentralized autonomous organizations (DAOs).

### Drivers

- Blockchain platforms continue to evolve to support the core elements of blockchain technologies identified in the Definition section.
- Platform developers are also attempting to build out traditional application stack capabilities. These include better modelling support, developer tools, frameworks, integration mechanisms and software development kits (SDKs).
- Feedback from early adopters along with enhanced design is leading to improved performance of the leading platforms.
- Development continues to progress in design, testing and piloting across different industries and has gained more traction with the digital acceleration fostered by addressing the challenges brought about by COVID-19.
- As digital acceleration pervades all industries and the public sector, more attention is being paid to specific use cases that blockchain platforms can support — such as the provenance of data or goods, portable identity, asset tokenization, payments, central bank digital currencies, among others.

### Obstacles

- Lack of clear success and proven use cases have been a challenge for promoting adoption.
- Most projects require cooperation among different entities, but achieving governance and cooperation across multiple enterprises is proving difficult.
- Adopting blockchain features and capabilities to provide business value requires that enterprises process adjustments which are disruptive to today's business processes that most enterprises are not willing to bear.
- For enterprises, the full scope of decentralization demands that the platform has to solve competing demands in terms of cost, performance and security. This is while trying to match or better traditional features expected of enterprise software, including ease of use, developer support, reporting, interoperability and integration.

### User Recommendations

- Evaluate your strategy for pursuing one of five solution archetypes (see Use 4 Business Currencies and 5 Archetypes to Evaluate Blockchain Initiatives). Once chosen, assess vendor offerings extremely carefully using a technology-agnostic function framework to ensure an apple-to-apple comparison (see Guidance for Assessing Blockchain Platforms).
- Prepare for continued evolution, a shifting competitive landscape and consolidation of offerings.
- Check the level of platform openness, decentralization and tokenization functionality throughout the platform stacks carefully, as well as upgrade paths — especially in a multiparty business setup and/or consortia.
- Assess vendor capabilities for protocols, interoperability, integration, security mechanisms, performance, manageability, tooling and support.

### Sample Vendors

Enterprise Ethereum; Hyperledger; Nexledger Universal; R3 Corda

### Gartner Recommended Reading

Guidance for Assessing Blockchain Platforms

Guidance for Blockchain Solution Adoption

Use 4 Business Currencies and 5 Archetypes to Evaluate Blockchain Initiatives

How to Operate and Support Blockchain Platforms

How to Position Blockchain Platforms to Increase Adoption

## Climbing the Slope

### Open-Source NLP Toolkits

**Analysis By:** Adrian Lee

**Benefit Rating:** High

**Market Penetration:** More than 50% of target audience

**Maturity:** Early mainstream

#### Definition:

Open-source natural language processing (NLP) toolkits enable end users and commercial companies to review, modify or design the source codes for their own purposes, mostly free of charge. Open-source NLP toolkits address the most common problems of analyzing and processing large natural language corpora into structured data to support applications such as text analytics, chatbots or virtual assistants.

#### Why This Is Important

Open-source NLP toolkits have been available for over 20 years. Current releases incorporate a new generation of natural language technologies that provide value by optimizing business processes and operations. Applicable use cases are in business intelligence (text analytics), customer service (improving customer satisfaction, increasing engagement) and employee support (productivity applications, knowledge base management).

#### Business Impact

Open-source NLP toolkits enable:

- Rapid prototyping of conversational agents or applications.
- A test bed environment for enterprises to internally test their NLP requirements.
- Partial NLP capabilities for some conversational AI platform vendors.
- Enterprises to gain significant control over the application in order to develop customized solutions.
- Application vendors with conversational elements to develop their solutions for resale with limited fees.

### Drivers

- Growing adoption of text analytics and other language-based solutions is driving strong enterprise interest in building solutions with open-source NLP toolkits.
- Some enterprises need to customize their integrations of NLT with their applications and use cases.
- There is a lack of availability of customization of custom-made and packaged NLT platforms without high professional services fees.
- The long-term viability and functionality of NLT platforms currently in the market are uncertain.
- NLP toolkits are mature, have broad user communities and are a NLP foundation widely accepted by enterprises and some conversational AI providers to enable the most common modes of linguistic analysis to help machines understand text.
- Core features of open-source NLP toolkits position them to be suitable, lower-cost platforms for prototyping and piloting conversational agents or applications.
- An increased proportion of open-source NLP toolkits will come with some pretrained intent models, languages and test suites to accelerate the project delivery for end users.

### Obstacles

- Open-source NLP toolkit use cases are often focused toward supporting experimental research and noncommercial projects. Prototypes may not make it into production or be scaled up for commercial use.
- Open-source toolkits require enterprises to develop their own advanced in-house skills.
- To realize shorter-term business objectives, IT leaders still consider managed service providers that can better fulfill domain-specific needs and come with prebuilt intent models to accelerate deployment.
- Limitations can exist, especially where end users do not augment toolkits with suitable ML algorithms to improve the performance of the NLP output.
- Using open-source NLP toolkits is iterative and requires additional customization and integrations with enterprise applications before it delivers business benefits.

- Differences exist between self-service, managed service providers and open-source platforms in ontologies, taxonomies and domain specificity to drive contextual natural language understanding.

### User Recommendations

Open-source NLP toolkits are relevant for organizations that:

- Want to build their own natural language technology stack by starting with a suitable toolkit to match their existing IT infrastructure.
- Possess their own in-house resources of data scientists and artificial intelligence (AI) technology engineers for NLP.
- Can bear a longer and more costly initial IT investment by using open-source NLP toolkits in order to benefit subsequently from lower total costs of ownership.

### Sample Vendors

Apache OpenNLP; Intel; NLTK Project; PyTorch; spaCy; Stanford NLP Group

### Gartner Recommended Reading

Architecture of Conversational AI Platforms

Cool Vendors in Conversational AI Platforms

Guidance Framework for Evaluating Conversational AI Platforms

Emerging Technologies: Tech Innovators in Conversational AI and Virtual Assistants

## Event Stream Processing

**Analysis By:** W. Roy Schulte, Pieter den Hamer

**Benefit Rating:** Transformational

**Market Penetration:** 20% to 50% of target audience

**Maturity:** Early mainstream

**Definition:**

Event stream processing (ESP) is computing that is performed on streaming data (sequences of event objects) for the purpose of stream analytics or stream data integration. ESP is typically applied to data as it arrives (data "in motion"). It enables situation awareness and near-real-time responses to threats and opportunities as they emerge, or it stores data streams for use in subsequent applications.

**Why This Is Important**

ESP is a key enabler of continuous intelligence and related real-time aspects of digital business. ESP's data-in-motion architecture is a radical departure from conventional data-at-rest approaches that historically dominated computing. ESP products have progressed from niche innovation to proven technology and now reach into the early majority of users. ESP will reach the Plateau of Productivity within several years and eventually be adopted by multiple departments within every large company.

**Business Impact**

ESP transformed financial markets and became essential to telecommunication networks, smart electrical grids and some IoT, supply chain, fleet management, and other transportation operations. Most of the growth in ESP during the next 10 years will come from areas where it is already established, especially IoT and customer experience management. Stream analytics from ESP platforms provides situation awareness through dashboards and alerts, and detects anomalies and other significant patterns.

**Drivers**

Five factors are driving ESP growth:

- Companies have ever-increasing amounts of streaming data from sensors, meters, digital control systems, corporate websites, transactional applications, social computing platforms, news and weather feeds, data brokers, government agencies and business partners.
- Business is demanding more real-time, continuous intelligence for better situation awareness and faster, more-precise and nuanced decisions.
- ESP products have become widely available, in part because open-source ESP technology has made it less expensive for more vendors to offer ESP. More than 40 ESP platforms or cloud ESP services are available. All software megavendors offer at least one ESP product and numerous small-to-midsize specialists also compete in this market.

- ESP products have matured into stable, well-rounded products with many thousands of applications (overall) in reliable production.
- Vendors are adding expressive, easy-to-use development interfaces that enable faster application development. Power users can build some kinds of ESP applications through the use of low-code techniques and off-the-shelf templates.

### Obstacles

- ESP platforms are overkill for most applications that process low or moderate volumes of streaming data (e.g., under 1000 events per second), or do not require fast response times (e.g., less than a minute).
- Many ESP products required low-level programming in Java, Scala or proprietary event processing languages until fairly recently. The spread of SQL as a popular ESP development language has ameliorated this concern for some applications, although SQL has limitations. A new generation of low-code development paradigms has emerged to further enhance developer productivity but is still limited to a minority of ESP products.
- Many architects and software engineers are still unfamiliar with the design techniques and products that enable ESP on data in motion. They are more familiar with processing data at rest in databases and other data stores, so they use those techniques by default unless business requirements force them to use ESP.

### User Recommendations

- Use ESP platforms when conventional data-at-rest architectures cannot process high-volume event streams fast enough to meet business requirements.
- Acquire ESP functionality by using a SaaS offering, IoT platform or an off-the-shelf application that has embedded CEP logic if a product that targets their specific business requirements is available.
- Use vendor-supported closed-source platforms or open-core products that mix open-source with value-added closed-source extensions for mainstream applications that require enterprise-level support and a full set of features. Use free, community-supported, open-source ESP platforms if their developers are familiar with open-source software and license fees are more important than staff costs.
- Use ESP products that are optimized for stream data integration to ingest, filter, enrich, transform and store event streams in a file or database for later use.

### Sample Vendors

Amazon; Confluent; Google; IBM; Informatica; Microsoft; Oracle; SAS; Software AG; TIBCO Software

### Gartner Recommended Reading

Market Guide for Event Stream Processing

Adopt Stream Data Integration to Meet Your Real-Time Data Integration and Analytics Requirements

Market Share Analysis: Event Stream Processing (ESP) Platforms, Worldwide, 2020

## Cloud Native Architecture

**Analysis By:** Anne Thomas

**Benefit Rating:** Moderate

**Market Penetration:** 5% to 20% of target audience

**Maturity:** Adolescent

### Definition:

Cloud native architecture is the set of application architecture principles and design patterns that enables applications to fully utilize the agility, scalability, resiliency, elasticity, on-demand and economies of scale benefits provided by cloud computing. Cloud native applications are architected to be latency-aware, instrumented, failure-aware, event-driven, secure, parallelizable, automated and resource-consumption-aware (LIFESPAR).

### Why This Is Important

Many organizations are moving to cloud native architecture as they shift their application workloads to cloud native application platforms. Cloud native principles and patterns enable applications to operate efficiently in a dynamic environment and make the most of cloud benefits. Organizations that simply “lift and shift” legacy applications to cloud native platforms often find that the applications perform poorly, consume excessive resources and aren’t able to fail and recover gracefully.

### Business Impact

- Cloud native architecture ensures that applications can take full advantage of a cloud platform's capabilities to deliver agility, scalability and resilience.
- It enables DevOps teams to more effectively use cloud self-service and automation capabilities to support continuous delivery of new features and capabilities.
- It can also improve system performance and business continuity, and it can lower costs by optimizing resource utilization.

### Drivers

- Organizations want to make the most of cloud computing to support their digital business initiatives, but they can't fully exploit cloud platform benefits without cloud native architecture.
- Software engineering teams are adopting cloud native architecture to support cloud native DevOps practices, including self-service and automated provisioning, blue/green deployments, and canary deployments. A basic set of rules known as the " twelve-factor app" ensures that applications can support these practices.
- Cloud native architecture includes practices such as application decomposition (following the mesh app and service architecture [MASA] structure), containerization, configuration as code, and stateless services.

### Obstacles

- Cloud native architecture adds a level of complexity to applications, and development teams require new skills, new frameworks, and new technology to be successful.
- Without proper education, architects and developers can apply the principles poorly and deliver applications that fail to deliver the expected benefits. This leads to developer frustration in adopting the new patterns and practices.
- Not every cloud-hosted application needs to be fully cloud-native and developers may be confused about when they need to use particular patterns to address their specific application requirements.

### User Recommendations

- Use the twelve-factor app rules and the LIFESPAR architecture principles to build cloud native applications.
- Incorporate cloud native design principles in all new applications irrespective of whether you currently plan to deploy them in the cloud. All new applications should be able to safely run on a cloud platform, even if it doesn't fully utilize cloud characteristics.
- Apply cloud native design principles as you modernize legacy applications that you plan to port to a cloud platform to ensure that they can tolerate ephemeral or unreliable infrastructure. Otherwise, they are likely to experience stability and reliability issues.
- Select an application platform that matches your cloud native architecture maturity and priorities. Recognize that low-code platforms enable rapid development of cloud-ready applications, but they won't provide you with the full flexibility to apply LIFESPAR and twelve-factor principles.

### Sample Vendors

Amazon Web Services; Google; Microsoft; Red Hat; VMware

### Gartner Recommended Reading

How to Help Software Engineering Teams Modernize Their Application Architecture Skills

How to Modernize Your Application to Adopt Cloud-Native Architecture

A Guidance Framework for Modernizing Java EE Applications

Guidance Framework for Modernizing Microsoft .NET Applications

### Software Composition Analysis

**Analysis By:** Dale Gardner

**Benefit Rating:** High

**Market Penetration:** 20% to 50% of target audience

**Maturity:** Early mainstream

**Definition:**

Software composition analysis (SCA) products are specialized application security testing tools that detect open-source software (OSS) and third-party components known to have security and/or functionality vulnerabilities, and to identify potentially adverse OSS licensing terms. It is an essential element in strategies to ensure an organization's software supply chain includes secure and trusted components and, therefore, aids in secure application development and assembly.

**Why This Is Important**

SCA's ability to help ensure the software supply chain is current, free of known vulnerabilities, and properly licensed supports the use of OSS in application development. It is an essential element of application security testing, given the ubiquity of OSS in applications and the potential for significant risk.

**Business Impact**

- SCA is broadly applicable and beneficial to all types of organizations with application development efforts. It should be considered a foundational element of application security testing.
- Its primary users are application development and security teams, tasked with ensuring the integrity of open-source components — and less frequently commercial software — in development.
- License assessments, once the primary use case for SCA, remain an important function for legal and sourcing groups.

**Drivers**

- SCA has moved from a mature to an early mainstream position in this Hype Cycle. This unusual reversal is a consequence of multiple changes, including a number of significant technological innovations (e.g., greater integration with other types of testing, more sophisticated analysis of the potential impacts of a package upgrade) and its vastly broader applicability to all types of organizations.
- The underlying driver for the growing importance of SCA is the increased use of OSS in application development. The prevalence of OSS, both as individual packages and within container images, has led to SCA becoming a key control on CI/CD pipelines and containerized environments.
- Repeated instances of high-impact vulnerabilities in OSS, rendered pervasive because of the underlying components' broad use across organizations, along with ever-present supply chain attacks, demonstrate the need to better understand the risks posed by OSS and commercial software packages.
- SCA analyzes code in use and reports issues gathered from information collected from sources such as OSS communities, private research and publicly available security vulnerability databases. Information typically includes intellectual property (IP) ownership, known security and functionality vulnerabilities, and references to the most recent versions of components. Some tools supplement this information with guidance on a preferred update version, balancing stability, remediation of flaws and potential adverse impacts on the functionality of existing code, all of which boosts developer and security team productivity and efficiency. In a few cases, tools have begun to incorporate assessments of operational risk, in an effort to help prevent supply chain attacks.
- SCA technology can help ensure developers are meeting compliance requirements and ethical and legal standards for code use, reducing the likelihood of unwanted or unapproved code that creates risks to an organization's intellectual property.

### Obstacles

- The quantity of OSS is inestimable. The number of packages, languages and interdependencies creates an intricacy impossible for a single tool to manage. SCA tools specialize in a subset of languages, limiting utility. Development teams with diverse portfolios will turn to multiple tools, increasing costs and complexity.
- SCA is — incorrectly — seen as a solution to software supply chain attacks. While the tools report known vulnerabilities, with few exceptions they don't test code to identify new issues. Vendors will expand the scope of their research to identify signs of package compromise, and expand testing to discover issues proactively.
- Commercial software libraries are often outside the scope of SCA, but pose many of the same risks. This leaves a protection gap.
- Tools are increasingly optimized for application development use cases. But, legal and sourcing teams still rely on them to identify licensing issues. Tools will develop alternative interfaces to support different users.

### User Recommendations

- Examine SCA technologies as a key ingredient of every software security program, used in conjunction with other AST tools. AST vendors should provide this capability built-in or via a partnership with a dedicated SCA vendor, and through dedicated commercial and open source tools.
- Analyze SCA warnings on licensing issues to address the legal issues stemming from components' IP ownership or license terms. This should be done by legal experts.
- Use SCA tools on a regular basis to audit repositories that contain software assets to ensure the software developed and/or used by the enterprise meets organizational standards.
- Utilize SCA tools to inspect the components application developers plan to use. SCA tools fit well within DevSecOps workflows, where scanning can be automated as part of the rapid development processes.
- Use SCA tools in conjunction with a formal corporate IP strategy that has established clear responsibility across the company.

### Sample Vendors

Checkmarx; GrammaTech; Hdiv Security; JFrog; Snyk; Sonatype; Synopsys; Veracode; WhiteSource

### Gartner Recommended Reading

Market Guide for Software Composition Analysis

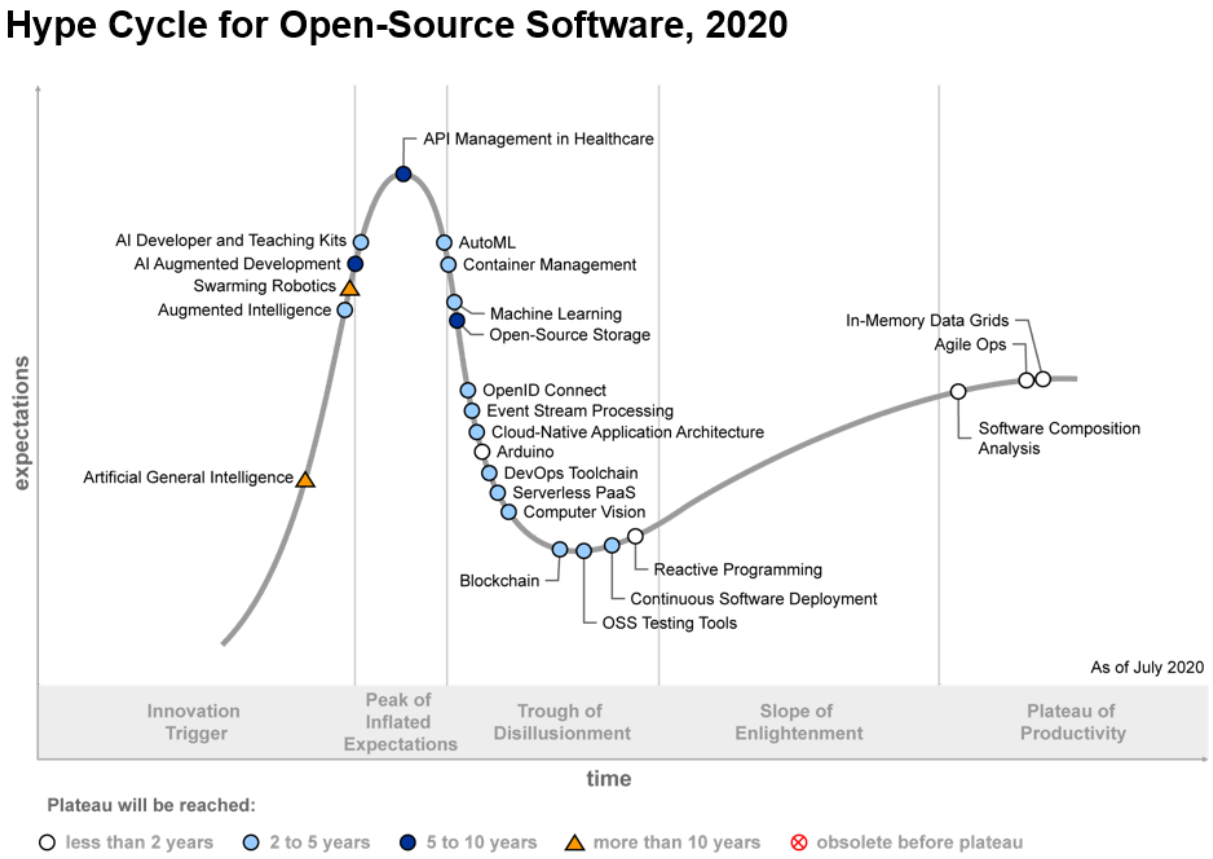
Critical Capabilities for Application Security Testing

Magic Quadrant for Application Security Testing

How to Manage Open-Source Software Risks Using Software Composition Analysis

# Appendixes

## Figure 2. Hype Cycle for Open-Source Software, 2020




Source: Gartner
ID: 450395

Gartner

Source: Gartner (July 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 (for example, 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<br>Pilots and deployments by industry leaders | First generation<br>High price<br>Much customization |
| *Adolescent* | Maturing technology capabilities and process understanding<br>Uptake beyond early adopters | Second generation<br>Less customization |
| *Early mainstream* | Proven technology<br>Vendors, technology and adoption rapidly evolving | Third generation<br>More out-of-box methodologies |
| *Mature mainstream* | Robust technology<br>Not much evolution in vendors or technology | Several dominant vendors |
| *Legacy* | Not appropriate for new developments<br>Cost of migration constrains replacement | Maintenance revenue focus |
| *Obsolete* | Rarely used | Used/resale market only |

Source: Gartner (July 2021)

## Evidence

[1] In a Gartner Research Circle survey (Examining the Maturity of Open-Source Management), which was conducted from June through July 2019, over 90% of participating respondents indicated that their organizations used OSS in various functions, including mission-critical workloads.

[2] Gartner's 2020 Building Digital Platforms Study was conducted online during May and June 2020 among 206 respondents working for organizations in North America and Western Europe with at least $1B US in annual revenue. Organizations were from the manufacturing and natural resources, communications, media, services, retail, banking and financial services, insurance, healthcare, transportation and utilities industries.

Organizations also had to be working on digital business efforts or planning to do so, defined as involving the Internet of Things (IoT), delivery of public APIs, private/B2B APIs or a combination thereof. Quotas were set to ensure a majority of organizations had a fully implemented digital business initiative.

Respondents were required to have a job title of Director or above and to be involved in either digital business, data analytics, IoT or API-based platforms for partners. In respect to digital business initiatives, they were also required to have a role in either defining technology requirements, investigating or evaluating service providers or making final decisions.

The results of this study do not represent global findings or the market as a whole, but reflect the sentiments of the respondents and companies surveyed.

[3] The 2020 State of the Octoverse, Octoverse (via GitHub).

[4] 2021 Open Source Security and Risk Analysis (OSSRA) Report, Synopsys.

[5] The 2021 Tidelift Open Source Maintainer Survey, Tidelift.

## Document Revision History

Hype Cycle for Open-Source Software, 2020 - 13 July 2020

Hype Cycle for Open-Source Software, 2019 - 6 August 2019

Hype Cycle for Open-Source Software, 2018 - 18 October 2018

Hype Cycle for Open-Source Software, 2017 - 12 October 2017

Hype Cycle for Open-Source Software, 2016 - 11 July 2016

Hype Cycle for Open-Source Software, 2014 - 30 July 2014

---

## Recommended by the Authors

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

Understanding Gartner's Hype Cycles

Create Your Own Hype Cycle With Gartner's Hype Cycle Builder

A CTO's Guide to Top Practices for Open-Source Software

Peer Connect Perspectives: Developing an Open-Source Strategy

Four Steps to Adopt Open-Source Software as Part of Your Test Automation Stack

Open-Source Options for Threat Detection and Incident Response

What the Automotive CIO Needs to Know About Open-Source Platforms

Ensure Safe and Successful Usage of Open-Source Software With a Comprehensive Governance Policy

How to Manage Open-Source Software Risks Using Software Composition Analysis

Tool: Open-Source Software Governance Policy Template

---

## Table 1: Priority Matrix for Open-Source Software, 2021

| *Benefit* ↓ | *Years to Mainstream Adoption* | | | |
|---|---|---|---|---|
| | ***Less Than 2 Years*** ↓ | ***2 - 5 Years*** ↓ | ***5 - 10 Years*** ↓ | ***More Than 10 Years*** ↓ |
| Transformational | | Event Stream Processing<br>Machine Learning | AI-Augmented Software Engineering<br>Blockchain Platforms<br>Responsible AI | |
| High | DevOps Toolchain<br>Software Composition Analysis | Container Management<br>Open-Source NLP Toolkits | Container-Native Storage<br>Open-Source Program Office<br>Open-Source Solutions for CSPs | |
| Moderate | | Cloud Native Architecture<br>Service Mesh | Innersourcing<br>Open-Source Storage | |
| Low | | | | |

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 (for example, 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<br>Pilots and deployments by industry leaders | First generation<br>High price<br>Much customization |
| *Adolescent* | Maturing technology capabilities and process understanding<br>Uptake beyond early adopters | Second generation<br>Less customization |
| *Early mainstream* | Proven technology<br>Vendors, technology and adoption rapidly evolving | Third generation<br>More out-of-box methodologies |
| *Mature mainstream* | Robust technology<br>Not much evolution in vendors or technology | Several dominant vendors |
| *Legacy* | Not appropriate for new developments<br>Cost of migration constrains replacement | Maintenance revenue focus |
| *Obsolete* | Rarely used | Used/resale market only |

Source: Gartner (July 2021)