Hype Cycle for AI in Software Engineering, 2024

29 July 2024 - ID G00818219 - 123 min read

By Adrian Leow, Deepak Seth, **and 1 more**

AI is transforming software engineering with tools like AI code assistants to enhance developer experience and productivity. Software engineering leaders are tasked with leveraging AI in both new and existing applications. This Hype Cycle tracks the latest AI technologies in software engineering.

Strategic Planning Assumption

By 2028, the use of generative AI (GenAI) will reduce the cost of modernizing legacy applications by 30% from 2023 levels.

Analysis

What You Need to Know

Software engineering leaders must understand the transformative potential of AI in application development. For example:

- AI code assistants assist in writing and analyzing software code. The assistants use foundation models such as large language models (LLMs) optionally fine-tuned for code, or program-understanding technology, or a combination of both.

- AI-augmented software engineering integrates AI tools throughout the development life cycle, optimizing processes.
- Synthetic data, crucial for testing, can enable comprehensive validation minimizing real-world data dependencies.

Understanding the complexities of engineering AI systems is crucial. ModelOps ensures the efficient deployment, monitoring and management of AI models, enabling seamless integration and continuous delivery. This innovation is further enhanced by integrating various models:

- Vision models
- Language models (covering both text and voice)
- Multimodal models
- Open domain code models

Together, these technologies push the boundaries of natural language processing and beyond, offering advanced capabilities that drive continuous innovation. Vector search, essential for managing complex, high-dimensional data, supports the rapid querying and retrieval necessary in AI-driven applications.

Balancing these aspects ensures that software developers can harness AI's full potential, driving both innovative application development and robust, scalable AI systems engineering.

The Hype Cycle

The market for AI in software engineering is rapidly evolving. Gartner sees this evolution driven by three key trends.

Using AI in software development:

- **AI code assistants:** Tools like GitHub Copilot and OpenAI Codex are transforming coding practices by providing real-time suggestions, reducing development time and increasing efficiency. These tools are among the fast movers, seeing rapid adoption due to their immediate

impact on productivity.

- **Open-source GenAI code models:** These will provide developers a self-hosted option for experimenting and choosing the model that best meets their use cases. The small size of code-specific models enables them to provide coding accuracy comparable to general-purpose models while being less resource-intensive.
- **AI-augmented software engineering:** Integrating AI across the software development life cycle, starting with design-to-code tools in planning through to AI-augmented testing tools in the DevOps loop, will become more important in realizing promised productivity gains. This trend is driven by the need for faster, more reliable development cycles.

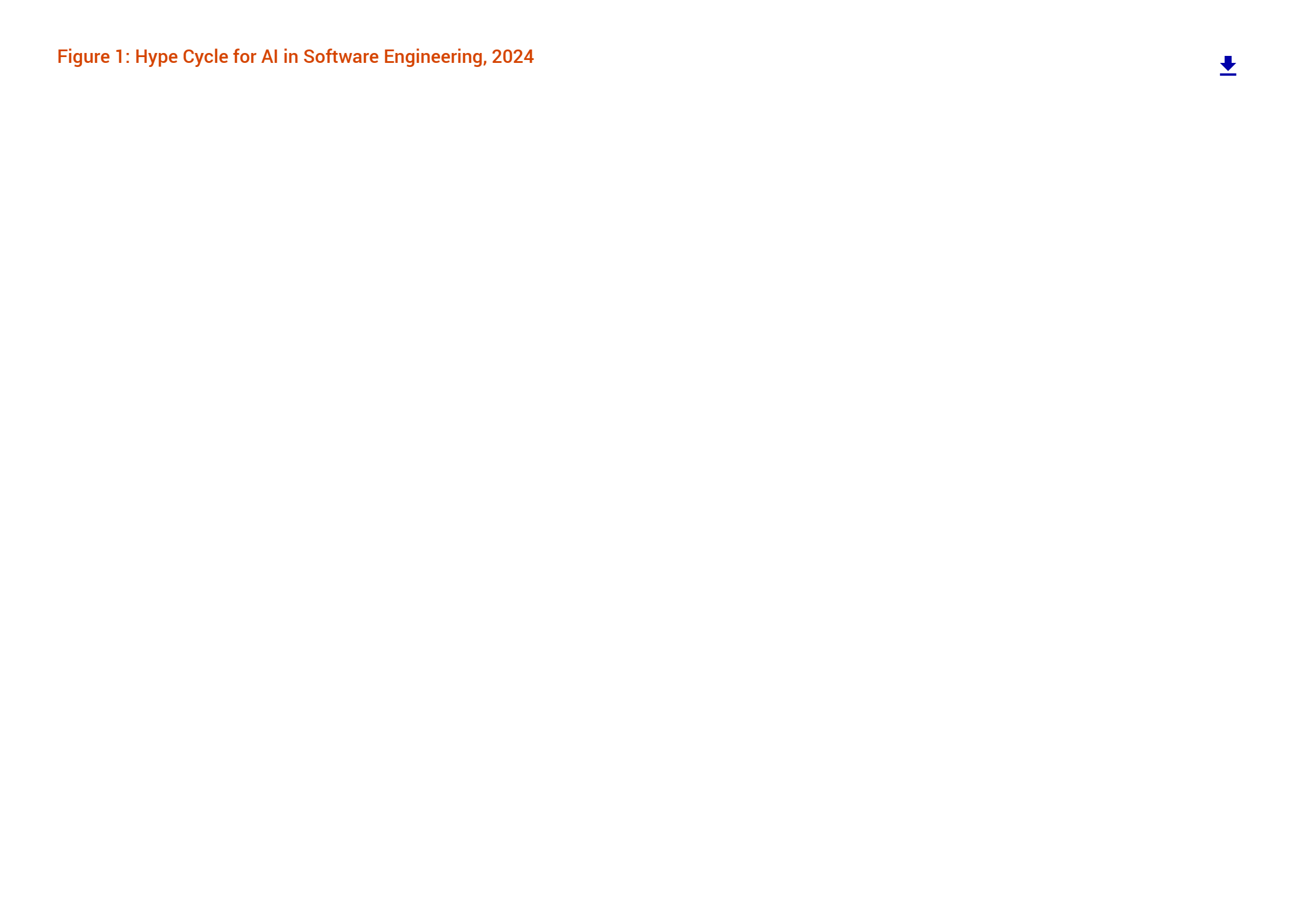
Creating software that uses AI:

- **Synthetic data:** The use of synthetic data for training and testing is gaining traction, particularly where real data is scarce or privacy concerns are paramount. It's seen as a significant innovation due to its potential to solve data availability issues.
- **AI-augmented applications:** Applications enhanced with AI capabilities, such as chatbots, recommendation systems and predictive analytics, are becoming mainstream. These innovations are critical in sectors like retail, healthcare and finance, as well as in public sector entities.

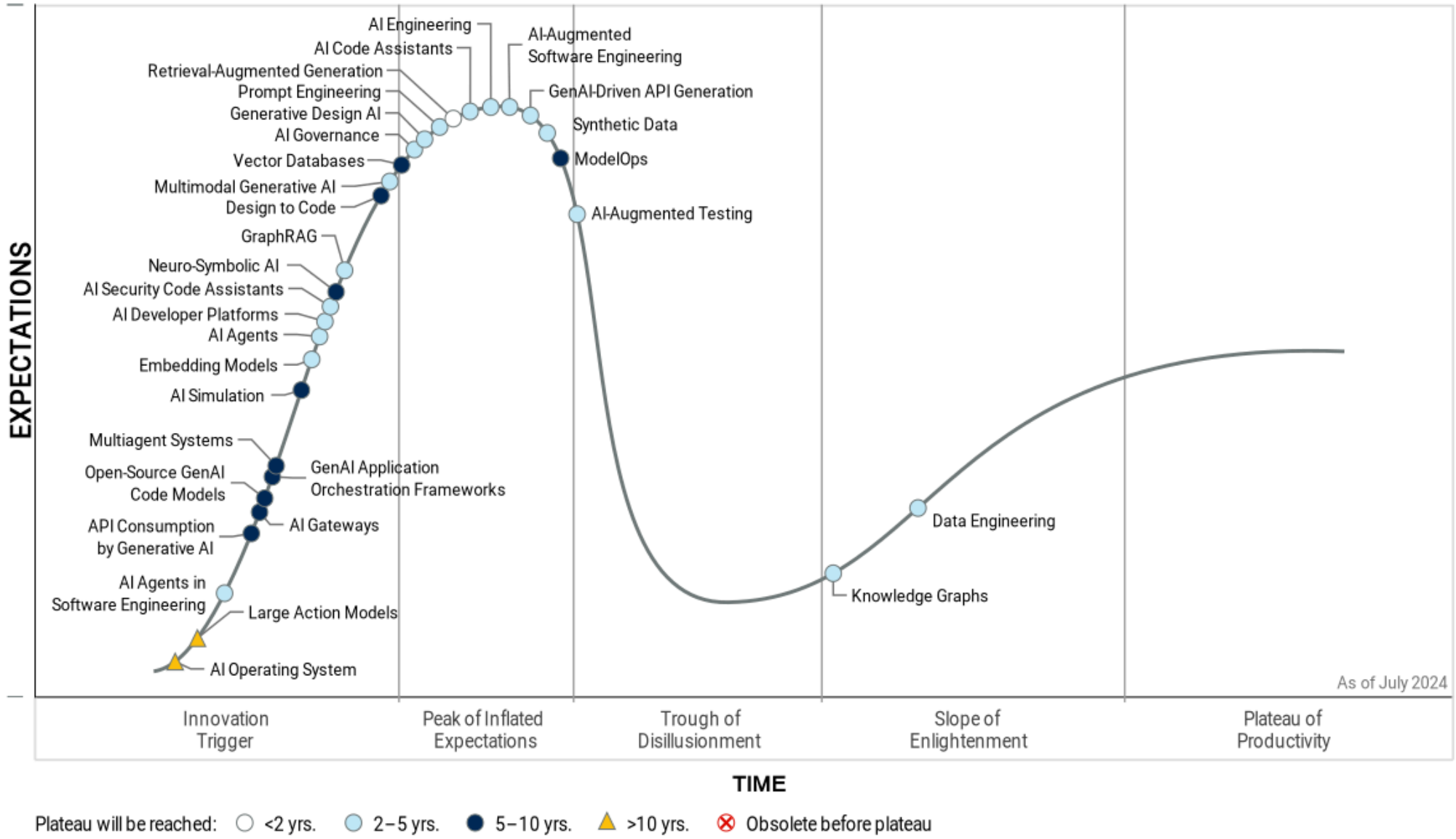
Creating artificial intelligence/machine learning (AI/ML) systems:

- **ModelOps:** Efficiently managing AI models in production is a growing focus, with tools and frameworks emerging to address deployment, monitoring and retraining needs. ModelOps is pivotal in ensuring AI systems remain performant and relevant.
- **Large language models (LLMs):** LLMs like ChatGPT, Claude, Gemini, Llama, Mistral and others are driving significant advancements in natural language processing, powering new applications in content creation, customer service and beyond. They are among the most hyped due to their broad applicability.
- **Vector databases:** Essential for handling high-dimensional data in AI applications, vector databases are seeing increased adoption. Innovations in this area are fast-moving, driven by the need for efficient data retrieval and analysis.

Figure 1: Hype Cycle for AI in Software Engineering, 2024



Hype Cycle for AI in Software Engineering, 2024



Gartner

The Priority Matrix

Software engineering leaders can use the Priority Matrix to help them stay ahead of the trends identified in the Hype Cycle, thus maintaining a competitive edge and fostering a culture of innovation.

Software engineering leaders can expedite AI adoption by prioritizing key technologies from the Innovation Trigger and Peak of Inflated Expectations phases:

- Leveraging AI code assistants and AI-augmented software engineering tools helps developer teams boost development productivity and quality.
- Embracing open-source AI models and synthetic data for testing ensures robust and efficient workflows.
- Investing in ModelOps and vector databases helps streamline AI system management and data handling, setting the stage for long-term success.

The impact of these innovations varies by role and time frame:

- **Developers:** Immediate productivity gains from AI code assistants are occurring already and will continue to increase within the next two years.
- **Data scientists:** Enhanced efficiency in managing complex datasets with vector databases is between five and 10 years.
- **AI specialists:** Improved deployment and monitoring through ModelOps, enabling faster iteration and innovation is projected to be between five and 10 years.
- **Quality assurance (QA) engineers:** More effective testing using synthetic data, which reduces reliance on real-world data, is also likely to fall in the two-to-five-year range.

These technologies will drive significant changes in roles, making some tasks more automated and freeing up time for higher-level strategic work.

Table 1: Priority Matrix for AI in Software Engineering, 2024

Benefit	Years to Mainstream Adoption			
	Less Than 2 Years	2 - 5 Years	5 - 10 Years	More Than 10 Years
Transformational		AI Agents in Software Engineering AI-Augmented Software Engineering AI Code Assistants AI Security Code Assistants Generative Design AI Multimodal Generative AI	API Consumption by Generative AI Design to Code	AI Operating System Large Action Models

Benefit	Years to Mainstream Adoption			
	Less Than 2 Years	2 - 5 Years	5 - 10 Years	More Than 10 Years
High	Retrieval-Augmented Generation	AI Agents AI-Augmented Testing AI Developer Platforms AI Engineering AI Governance Data Engineering Embedding Models Knowledge Graphs Prompt Engineering Synthetic Data	AI Simulation GenAI Application Orchestration Frameworks ModelOps Multiagent Systems Neuro-Symbolic AI Open-Source GenAI Code Models Vector Databases	
Moderate		GenAI-Driven API Generation GraphRAG	AI Gateways	
Low				

Source: Gartner

Off the Hype Cycle

On the Rise

AI Operating System

Analysis By: Svetlana Sicular, Jim Scheibmeir

Benefit Rating: Transformational

Market Penetration: Less than 1% of target audience

Maturity: Embryonic

Definition:

An AI operating system (AI OS) is an AI-enabled, integrated set of tools and agents that manages business and technology processes, hardware and software resources. It provides a common framework and services for the efficient, personalized and intuitive use of hosted technologies. AI OS contextually learns, adjusts, improves and self-manages over time to accommodate users and applications across cloud, traditional compute systems, robots, edge and smart devices, experiences, and interfaces.

Why This Is Important

A new generation of operating systems, AI OS, democratizes technologies for work and personal activities in the contextual, behavioral and knowledge-based ecosystem. AI OS allows natural language and other means of human communication for input, programming and defining outcomes. AI OS can adapt to its environment and self-manage to improve performance and feature functionality. It also facilitates the development, deployment and exploitation of complex LLM agents.

Business Impact

AI OS has the potential to transform work. It can:

- Adjust to new devices and make business and technical decisions based upon personalized context of individuals, behaviors and environments.
- Create an intuitive user experience, which potentially can boost productivity of every knowledge worker by fulfilling their requests and anticipating their needs.
- Change the design and implementation of software, hardware and programming paradigms.
- Self-tune performance to reduce costs for both sustainability and operational expenses.

Drivers

- Operating systems evolve by offering higher levels of abstraction to accommodate more users and tasks. AI and LLMs enable new levels of abstraction, both to engage traditionally nontechnical audiences and to offer new frameworks for technologists.
- Natural language, gesture and other means of human communication will dramatically expand and democratize technology capabilities to new audiences.
- AI OS is potentially a means to provide a common understanding of much of what is needed for a business, a society or a civilization to work. Ecosystems will emerge around the AI OSs, when there is critical mass for standards or adoption.
- AI has already improved all layers representing the main concepts of an OS: kernel, hardware, middleware, utilities, scheduling, orchestration and interfaces. For example, Google DeepMind's AlphaDev relies on AI for faster sorting algorithms that participate in most computer operations; Anthropic pioneered Constitutional AI to enable programmatic AI guardrails via a set of rules and principles (a "constitution") that guide training of a GenAI model.
- The rise of AI agents changes application-level concepts. AI agents can understand instructions, make decisions and take action, up to achieving autonomy. They will be able to tackle increasingly complex, multimodal tasks that require reasoning, execution and interaction with the physical world. For example, Intuit announced a proprietary generative AI operating system with custom-trained financial LLMs that specialize in solving tax, accounting, marketing, cash flow and personal finance challenges.
- Data management is at the beginning of the new cycle to ensure AI-ready data and, via data, provide context and vast knowledge for personalized activities, including autonomous decision making.

- Automotive, public sector, retail, healthcare investigate the benefits of AI OS concepts for edge devices, connected vehicles and robotics across use cases.

Obstacles

The embryonic nature of AI OS means it is not yet ready for full use. Many challenges need to be worked through before more widespread adoption:

- AI OS cybersecurity, roles, audit and governance of autonomous actions still need to be defined.
- AI OSs are still in prototype. Although they demonstrate innovative ideas that align with the trajectory of AI use cases and capabilities, they are not yet fully functional.
- Approaches to AI agents are rapidly multiplying and colliding; many changes and fast progress are happening at once.
- AI OS must provide guardrails, but currently available tooling for safe and responsible AI is fragmented.
- AI OS requires standardization. Multiple providers will produce intelligent entities that were not designed to work together with others made by rivals. The way through this is to provide a defined set of self-adjusting processes and documents that will serve as nonstatic standards across many technology vendors.
- Most technology components for AI OS are slow, flawed and hard to use.
- A process to validate the information used by AI to ensure its consistency and trustworthiness is necessary — but currently lacking. It includes community and ecosystem building for input.

User Recommendations

- Facilitate better machine-human interaction through UX redesign, feedback loops and user education on prompt engineering. Prioritize the ease of use when implementing AI for new audiences.
- Inventory your existing applications and technologies, which can benefit a greater number of users via conversational interfaces.
- Experiment with autonomous AI agents to determine their applicability in various environments.

- Outline various tasks and scenarios to explore the potential of multiagent systems to operate both collaboratively and independently.
- Establish AI life cycle guardrails, including legal and ethical guidelines around autonomy, liability, robust security measures and data privacy protocols, when implementing AI agents.

Sample Vendors

Amazon Web Services; Anthropic; Apple; Google; Lenovo; Microsoft; OpenAI

Gartner Recommended Reading

Magic Quadrant for Cloud AI Developer Services

Critical Capabilities for Cloud AI Developer Services

Become an AI-First Organization: 5 Critical AI Adoption Phases

Quick Answer: What Makes Data AI-Ready?

Large Action Models

Analysis By: Frank O'Connor

Benefit Rating: Transformational

Market Penetration: Less than 1% of target audience

Maturity: Embryonic

Definition:

Large action models (LAMs) are foundation models trained and optimized to identify and generate an action or set of actions that can be used to impact a target environment to meet a goal. For example, a large language model (LLM) solution can recommend a good restaurant to you; a LAM can book it for you.

Why This Is Important

By integrating LLMs with LAMs, users can express their goals in natural language. LAMs devise the required actions and deploy AI agents — software entities capable of autonomously executing tasks — to accomplish those goals across diverse digital or physical environments. LAMs have the potential to significantly disrupt the field of UI/user experience by providing interactions that are not just intuitive, flexible and efficient, but also highly personalized.

Business Impact

Industries likely to benefit from LAMs are automotive, education, healthcare, professional services, technology and software, with impacts including:

- Enhanced customer service experiences with more natural interactions, improving user satisfaction.
- Guided intelligent decision support solutions across different business functions.
- Delivery of personalized content and experiences as well as curated learning journeys.
- Better patient monitoring and diagnostic support, streamlining healthcare workflows.

Drivers

- **AI breakthroughs:** Recent breakthroughs in sophistication and utility of AI, such as generative pretrained transformers and LLMs, are enabling LAMs to better understand the intention behind human prompts and plan how to deliver outcomes that satisfy human prompts.
- **Multimodal understanding:** LAMs' ability to use diverse modalities like vision, audio and language enables more general and flexible AI assistants and agents. This allows automatic adaptation to changes in the workflow, user interface or API. This understanding enables LAMs to take high-level instructions and create advanced workflows without explicit programming, significantly reducing the development time and effort for automation.
- **Open-ended task capabilities:** Current AI models struggle with open-ended tasks requiring action sequences. LAMs could allow more complex and complete interactions by spanning agents to deliver more composable interactions that can achieve more wide-ranging goals.

- **Reinforcement learning integration:** LAMs provide a promising avenue to combine the pattern recognition abilities of LLMs with reinforcement learning for decision making and planning complex behaviors.
- **Autonomous systems:** Self-driving cars, drones, space exploration and other autonomous systems may employ LAMs to improve their ability to robustly execute complex objectives.
- **Robotics and physical world interaction:** LAMs could enable more capable and flexible robotic systems that can learn to execute complex action sequences from data, rather than relying solely on manually programmed routines. This has applications in markets such as manufacturing, logistics, healthcare and household assistance.
- **Virtual agents and environments:** Before being deployed in the real world, LAMs can first be developed and tested extensively in virtual environments and simulations. This allows safer iteration and scalable data collection for training the models.

Obstacles

- **Lack of trust:** Organizations are unsure if the human customer can trust the technology to accurately predict and execute tasks, and if the machine customer can trust the organization offering the service.
- **Data scarcity:** Training LAMs requires vast datasets of mapped action sequences and their corresponding goals across diverse environments.
- **Computational demands:** LAMs are computationally intensive to train due to the complexity of modeling action sequences, requiring immense computational resources.
- **Interpretability and oversight:** Action policies learned by LAMs may be opaque and have poor explainability, requiring mechanisms for human interpretability, oversight and control.
- **Accessibility:** While LAMs and AI agents have the potential to transform access for disabled people, care must be taken to ensure their interfaces are accessible and inclusive.

User Recommendations

- Start with focused, constrained applications and environments rather than aiming for general capabilities initially.
- Invest heavily in high-fidelity simulation infrastructure to enable scalable training and safe iterative development before real-world deployment.

- Build interpretability, human oversight and control mechanisms from the ground up to ensure LAM actions remain transparent and aligned with intended goals.

Sample Vendors

Covariant; MultiOn; rabbit; Toloka AI

Gartner Recommended Reading

Predicts 2024: The Future of Generative AI Technologies

Research Roundup for Generative AI

Innovation Insight: AI Agents

AI Agents in Software Engineering

Analysis By: Haritha Khandabattu, Steve Deng

Benefit Rating: Transformational

Market Penetration: 1% to 5% of target audience

Maturity: Emerging

Definition:

AI agents in software engineering are autonomous or semiautonomous software entities designed specifically to streamline and enhance different aspects of the development process, from requirements engineering to testing. They use AI techniques to perceive, make decisions, take actions and achieve goals in their digital or physical environments.

Why This Is Important

Demand for software delivery always exceeds the capacity to deliver it. Augmenting software engineering with AI, including AI agents in the development process, enables leadership to choose between delivering more of their needs with the same people, skills, capacity and cost

(factoring in AI costs), or delivering the same with fewer resources.

Business Impact

AI agents in software engineering present opportunities (e.g., reduced effort and time to delivery, better developer experience) and risks (e.g., lost knowledge, lack of scrutiny, and lower morale, job security and satisfaction). This may drive significant organizational changes with agents replacing hands-on developers. Overoptimism of AI agents' capabilities can also lead to unintended adoption consequences such as new threat vectors, software development life cycle (SDLC) bottlenecks, new quality or requirements definition issues, and new skill sets.

Drivers

- **Agentic workflows** represent a paradigm shift from traditional, linear processes to dynamic, AI-driven systems where humans and machines collaborate seamlessly. Unlike autonomous AI systems, agentic workflows emphasize the complementary roles of humans and AI in achieving shared objectives, with AI serving as a cognitive catalyst and humans providing context, intuition and oversight.
- AI-augmented software engineering: Application of AI technologies to assist software engineers throughout the SDLC includes the creation, validation, securing, deployment and maintenance of applications.
- AI code assistants assist in generating and analyzing software code and configuration. The assistants use foundation models that have been optionally fine-tuned for code, or program-understanding technologies, or a combination of both. It is important to note that agents are active entities while models, such as large language models (LLMs), are passive artifacts. AI agents in software engineering are independent software entities, acting on behalf of developers, that leverage those AI code assistants.
- AI-augmented testing comprises AI- and machine learning (ML)-based technologies and practices to make software testing activities more independent from human intervention. It continuously improves testing outcomes by learning from the data collected from performed activities.
- **Multimodal understanding** is the ability to use diverse modalities like vision, audio and language that enables more general and flexible AI assistants and agents. This allows automatic adaptation to changes in the development workflow, user interface or API.

Obstacles

- **Lack of trust:** Engineers are unsure whether they can trust the technology to accurately predict and execute tasks independently. Without a human in the loop, agents may take multiple consequential actions in rapid succession and bring about significant impacts before a human notices.
- **Lack of tooling:** Most of the current SDLC tools are designed for humans and not exactly suitable for LLMs/agents. e.g., integrated development environment (IDE) for LLMs, test harness and runtime sandbox for LLMs.
- **Data scarcity:** Preprompting and fine-tuning the LLMs require vast datasets of mapped action sequences and their corresponding goals across the software development process.
- **Security:** Since agents use LLMs to generate actions and have access to tools, they can be threats to real computing systems, applications and resources, thereby introducing dangers to confidentiality, integrity and availability in novel domains.

User Recommendations

Software engineering leaders must:

- Monitor the evolution of AI agents and adopt a phased approach to assess and deploy them into their development process.
- Incorporate AI agents into strategic planning by investing in understanding their capabilities and potential applications in SDLC, considering their increasing autonomy and wide-ranging usability. For instance: using AI agents to reproduce and isolate software bugs from bug reports or creating test cases from user stories.
- Focus on augmenting existing workflows and establishing developer skill and confidence in AI code assistant tools. When selecting such tool(s), include criteria around the code assistant vendor’s roadmap to gradually and responsibly evolve into a more agent-based tool.
- For the future, tackle challenges and seize opportunities in building AI agents (including multiagent systems) by assembling diverse, multidisciplinary teams to effectively develop and integrate agentic AI technologies in the development process, while also considering the security and ethical implications.

Sample Vendors

Cognition; MetaGPT; Open Ana; poolside; Pythagora; Stition

Gartner Recommended Reading

Innovation Insight: AI Agents

Emerging Tech: Secure Generative Communication for LLMs and AI Agents

Hype Cycle for Artificial Intelligence, 2024

Case Study: Human-Centric Generative AI Strategy

API Consumption by Generative AI

Analysis By: Mark O'Neill, Arun Batchu

Benefit Rating: Transformational

Market Penetration: 1% to 5% of target audience

Maturity: Emerging

Definition:

API consumption by generative AI (GenAI) involves large language models and other GenAI technologies consuming data and accessing application functionality via APIs. This typically involves the AI technology consuming the API definition, via an OpenAPI definition, and calling the API.

Why This Is Important

In its OpenAPI Moonwalk 2024, the OpenAPI Foundation identified “the rise of a new kind of API consumer — generative AI.” GenAI technologies will grow as users of APIs. This is driven by technologies such as AI agents and agentic flows. In addition, developers are increasingly using tools such as AI coding assistants. These tools help developers discover APIs, call APIs, and generate code, tests and documentation.

Business Impact

Software development powers business innovation and growth. Organizations must continuously improve software delivery processes and implement contemporary practices to attract developers. Gartner predicts AI coding assistants will drive developer productivity up 36% by 2028 (for more, see **How to Communicate the Value of AI Code Assistants**). Using GenAI to assist developers with consuming APIs in applications greatly impacts developer productivity and enhances the development experience.

Drivers

- Software depends more and more on APIs. A compelling use of GenAI is to assist developers with API discovery and the coding to make API requests.
- Software development tools and integrated developer environments are speedily adding GenAI capabilities to their products. Specific capabilities supporting API consumption are a common GenAI use case.
- GenAI-generated API tests significantly improve application delivery time and help ensure complete testing coverage.
- GenAI-generated documentation of API consumption in an application provides richer information and context of the application's usage of APIs.
- API design to industry specifications, in particular OpenAPI, is critical for consumption by AI models.

Obstacles

- When GenAI consumes APIs, security must be ensured. This may require the use of API security infrastructure, such as an API gateway
- When GenAI applications consume APIs, dependencies are created that may add latency.

User Recommendations

- GenAI for API consumption depends on the organization having quality APIs and API definitions. Implement API management that ensures your organization produces consistent, well-governed APIs.
- Choose a solution (such as retrieval-augmented generation) for making the organization's API inventory, standards, examples and documentation accessible to the selected GenAI technology. This enables GenAI responses (such as suggested APIs and code to invoke APIs) to reflect the enterprise's APIs and comply with its API governance.

- Advise and equip developers to serve in more strategic roles. Developers should be capable of perceptive insight and intuition to quickly but effectively vet GenAI selection of APIs and GenAI-suggested coding to invoke the APIs.
- Implement strong API security and protection that monitors all API requests, including API requests from GenAI-created code, for noncompliant and suspicious API activity.

Sample Vendors

LangChain; LlamaIndex; Tyk Technologies (Montag.AI)

Gartner Recommended Reading

Become an AI-First Organization: 5 Critical AI Adoption Phases

AI Gateways

Analysis By: Mark O'Neill, Andrew Humphreys

Benefit Rating: Moderate

Market Penetration: Less than 1% of target audience

Maturity: Embryonic

Definition:

AI gateways, also known as LLM routers or multi-LLM gateways, manage and secure connections to AI providers. Some AI gateways are based on established API gateway products, since REST APIs are the primary mechanism for connecting to AI providers, while others are purpose-built. Software engineering leaders can use AI gateways to apply security, such as protection of API keys issued by AI providers, multi-LLM routing, cost visibility, and data privacy scanning to their organization's AI usage.

Why This Is Important

As the volume and scale of generative AI and other types of AI projects have increased, organizations require greater control over their use of AI providers. AI gateways provide runtime traffic management between the organization and their AI providers. The gateways can help implement

and manage prompt-based policy controls, track AI service use and costs, route across multiple LLMs, and manage access to AI subscriptions, including protecting API keys issued by AI providers.

Business Impact

AI gateways manage interactions between applications and AI models by providing security, governance, observability and cost management. This reduces the risk of unexpected costs incurred from AI providers, and prevents private data in API traffic from being compromised or misused. An AI gateway may also enable an organization to manage usage of multiple AI models, from different providers, rather than focusing the organization on one provider.

Drivers

- **Control access costs:** Pricing models for AI services tend to be usage-based, which presents a risk for businesses as the costs associated with use of these services can accrue rapidly. Organizations are looking to control access costs, and AI gateways can help by caching responses to limit duplicate calls and tracking and controlling access to the services.
- **Enable visibility of use of AI services:** AI gateways provide a way to get greater visibility into the use of AI APIs across the organization by leveraging API management observability and analytics features for API usage.
- **Optimize access to AI engines:** AI gateways can be configured to provide a single API in front of multiple different AI providers, such as multiple LLMs. This means that developers can access multiple AI services using the same API.
- **Enforce security on AI interactions:** AI trust, risk and security management (AI TRiSM) ensures AI model governance, trustworthiness, fairness, reliability, robustness, efficacy and data protection. AI gateways address aspects of AI TRiSM, including model governance and reliability. In particular, protection of API keys issued by AI providers is vital. If an attacker gets access to an organization's API keys for an AI provider, they may access private data and run up large usage bills. AI gateways can be used to protect API keys issued by AI providers.

Obstacles

- **AI gateways are new and largely unproven.** Conducting a thorough evaluation before committing to a vendor is a requirement.
- **Latency** is a concern for any intermediary, and especially in AI where end users may notice any delay in response time. Unlike traditional API latency, the important metrics here are time-to-first token and token per seconds (or token throughput). An AI gateway may help users

optimize these two metrics.

- **A single point of failure** is also a concern if an AI gateway is deployed without redundancy in place.
- **Caching of AI API responses is problematic and complex,** since different AI models may use different prompts. Determining whether natural language requests are requesting the same information is difficult, particularly when dealing with a large context window.

User Recommendations

- Conduct a thorough evaluation of an AI gateway, since this is a new category with largely unproven products.
- For simpler scenarios such as rate-limiting of AI APIs, evaluate your existing API gateway's capabilities to act as an AI gateway, and support your requirements.
- Ask your AI providers whether they can provide similar functionality, which would remove the requirement for a separate AI gateway.

Sample Vendors

Aguru; Cloudflare; IBM; Kong; Lunar.dev; Martian; Portkey; Radiant

Gartner Recommended Reading

AI Pushes API Management Providers to Reimagine Their Products

Top Strategic Technology Trends for 2024: AI Trust, Risk and Security Management

Market Guide for API Gateways

How to Calculate Business Value and Cost for Generative AI Use Cases

10 Best Practices for Optimizing Generative AI Costs

Open-Source GenAI Code Models

Analysis By: Manjunath Bhat, Arun Batchu, Philip Walsh, Haritha Khandabattu

Benefit Rating: High

Market Penetration: 1% to 5% of target audience

Maturity: Emerging

Definition:

Open-source GenAI code models are specialized foundation models fine-tuned for coding such tasks as code generation, completion and explainability. The code model families include language-specific variants that are further fine-tuned for specific programming languages. The models are free to use, modify and distribute for research and commercial purposes; however, some use cases may be restricted. Model weights are released, but the precise datasets used to train the model may be unavailable.

Why This Is Important

Open-source GenAI code models are important for three reasons:

- They provide developers a self-hosted option for experimenting and choosing the one that best meets their use cases.
- Open access to models enables developers to extend their use based on community-driven innovation.
- Open-source code models permit further fine-tuning for specialized purposes — e.g., enterprise-specific or domain-specific languages.

Business Impact

Organizations can self-host open-source GenAI code models and manage private instances. This addresses IP theft and data exfiltration concerns that prevent organizations from adopting generative AI for software development and testing. The models' openness can democratize access to innovation and drive competition among development tool providers. The small size of code-specific models enables them to provide coding accuracy comparable to general-purpose models, while being less resource-intensive.

Drivers

- Technology providers continually release open-source GenAI code models in an attempt to thwart the competition or commoditize commercial offerings that have significant mind and market share.
- The initial success of commercial GenAI code assistants powered by proprietary GenAI code models encouraged open innovation on the part of big-tech providers, leading to the release of open-source GenAI code models.
- Freedom of choice and access to multiple, specialized code models creates a virtuous cycle between demand and supply and an incentive for rapid innovation.
- Small language models trained on a specific dataset for a specific task perform well on coding benchmarks, such as HumanEval, BabelCode, SQL-Eval and Mostly Basic Python Programming (MBPP). This is likely to result in language-specific (e.g., SQL, Python) and task-specific (e.g., in-line code completion, text-to-code) open-source GenAI code models.
- Developers in non-English speaking countries often do not see the same level of model accuracy and productivity gains for text-to-code functions. These countries will use open-source AI models to bring comparable accuracy to non-English-speaking developers.

Obstacles

- Open-source GenAI code models share risks similar to enterprise open-source software — they require governance and oversight, lack service-level commitments and need to comply with terms of use. Meta's terms of use restrict using Llama's output to train other AI models. Hence, using this output to generate synthetic data to train in-house custom models may violate license conditions.
- Open-source GenAI code models released by a single vendor may have multiple variants. The CodeGemma family includes 2B, 7B and 7B-Instruct variants; Code Llama includes Code Llama Python, Code Llama Instruct and Code Llama. Lack of expertise as to the variant to use and lack of repeatable, sustainable processes to download, manage, update and operate the models hinders adoption at scale.
- Developer experience drives developer tooling adoption. Open-source GenAI code models are downloadable artifacts, but this differs from embedding a code assistant in your workflow. Running models that provide fast responses may require GPU investments.

User Recommendations

- Include security, risk, legal and compliance teams, as well as the open-source program office in due diligence. Ensure that the open-source GenAI code adheres to the publisher's licensing conditions/terms of use.

- Establish a red team to perform adversarial attacks on code models to mitigate vulnerabilities and anomalies caused by their use.
- Create a curated open-source catalog as the single source of truth for a vetted open-source model. This catalog reduces the risk developers will arbitrarily source models from malicious repositories.
- Prioritize developer experience in going from experimentation to scaled rollouts of open source. It may require engineering teams to manage self-service tooling and development environments.
- Assess open-source AI models based on model accuracy, response time, resource requirements, integration with developer tooling, programming language support and documentation. Use a test harness with code-specific benchmarks to test models on sample code and assess their outputs.

Sample Vendors

Alibaba Cloud; Databricks; Deepseek AI; Defog; Google; Hugging Face; IBM; Meta; ServiceNow; Zhipu AI

Gartner Recommended Reading

Innovation Guide for AI Code Assistants

Quick Answer: What Are the Pros and Cons of Open-Source Generative AI Models?

Emerging Tech Impact Radar: Generative AI

How to Communicate the Value of AI Code Assistants

GenAI Application Orchestration Frameworks

Analysis By: Arun Chandrasekaran

Benefit Rating: High

Market Penetration: Less than 1% of target audience

Maturity: Embryonic

Definition:

Generative AI (GenAI) application orchestration frameworks provide an abstraction layer to enable prompt chaining, model chaining, interfacing with external APIs, retrieving contextual data from data sources and maintaining statefulness (or memory) across various model requests. These frameworks also provide templates, monitoring, evaluation and deployment capabilities for new GenAI applications.

Why This Is Important

GenAI application orchestration frameworks' functionalities expand AI foundation models' capabilities, making them more adaptable, interactive, context-aware and efficient in various applications. GenAI application orchestration frameworks enable seamless application workflows by providing a standard interface to GenAI models. These frameworks enable data integration, chaining prompts and models, effective prompting through prompt templates, input prompt optimization, and output parsing.

Business Impact

GenAI application orchestration frameworks enable production-ready, composable and extensible AI applications. AI engineers can easily swap out components and customize chains and capabilities. This is a necessary step, given the challenging, iterative nature of GenAI application development. IT leaders can use these frameworks to reduce tooling fragmentation by creating a centralized set of platform capabilities that helps diverse builders integrate data sources with models and chain prompts.

Drivers

- **Use-case maturity:** Most AI applications require composite AI capabilities, which unify a number of different models that are combined to enable the right balance of features to enable application quality, cost, performance and risk. This typically requires complex, multicomponent AI systems. Models may need to be chained together, routed between, or augmented with data retrieval or evaluation or guardrail steps. These orchestration frameworks support complex, production-ready AI applications.
- **Need for automation:** Although AI foundation models are a great advance, they still require extensive human intervention (in terms of prompts) to achieve business outcomes. GenAI application orchestration frameworks enable developers to build agents to reason about

problems and break them into smaller, executable subtasks. For example, with LangChain, developers can introduce context and memory into completions by creating intermediate steps and chaining commands.

- **GenAI application development:** Developers need to capitalize on GenAI models to build applications. GenAI application orchestration frameworks provide developers with a new way to build user interfaces (UIs) and automate application builds.
- **Model customization:** GenAI models can be combined with enterprise data through advanced prompt engineering techniques or model fine-tuning. GenAI application orchestration frameworks provide an open approach to integrating data sources with AI models.
- **Growing interest in AI agents**: AI agents dynamically interact with their environments to achieve goals. Orchestration tools can enable AI agents to be stateful, integrate with a wide array of data sources for knowledge and decision making and automatically orchestrate workflows for task completion.

Obstacles

- **Lack of awareness:** These tools are new, and there is a lack of understanding of what these tools do, which one to use and how to safely deploy them.
- **Immaturity:** Although the tools are gaining developer mind share, there are still gaps in meeting enterprise needs for stability and robust integration. Therefore, enterprise ML teams must cautiously navigate software bugs and security issues, characteristic of most early-stage open-source software (OSS) projects.
- **Lack of clear winners:** Although these orchestration frameworks are designed to bridge GenAI application development, their longevity and ability to innovate iteratively remain unknown.
- **Customization and flexibility:** Although these tools offer general capabilities for integrating GenAI models into application workflows, specific enterprise needs might require more customization than is currently available.

User Recommendations

- Encourage experimentation on what these tools do and their potential fit with your technical architecture.
- Identify the use cases in which you are implementing data retrieval or fine-tuning, which can benefit from these tools.
- Select tools that offer APIs and customization options that align with your current and future needs.

- Explore the autonomous agent capabilities of these tools cautiously, given the black-box nature of machine-to-machine interaction.
- Take a centralized platform approach to achieve standardization and automation across the GenAI applications you are building.

Sample Vendors

Amazon Web Services (AWS); deepset; Dust; Google; LangChain; LlamaIndex; Microsoft

Gartner Recommended Reading

Innovation Guide for Generative AI Models

A CTO's Guide to the Generative AI Technology Landscape

Multiagent Systems

Analysis By: Leinar Ramos, Pieter den Hamer, Anthony Mullen

Benefit Rating: High

Market Penetration: Less than 1% of target audience

Maturity: Embryonic

Definition:

A multiagent system (MAS) is a type of AI system composed of multiple, independent (but interactive) agents, each capable of perceiving their environment and taking actions. Agents can be AI models, software programs, robots and other computational entities. Multiple agents can work toward a common goal that goes beyond the ability of individual agents, with increased adaptability and robustness.

Why This Is Important

Current AI is focused on the creation of individual agents built for specific use cases, limiting the potential business value of AI to simpler problems that can be solved by single monolithic models. The combined application of multiple autonomous agents can tackle complex tasks

that individual agents cannot, while creating more adaptable, scalable and robust solutions. It is also able to succeed in environments where decentralized decision making is required.

Business Impact

Multiagent systems can be used in:

- **Generative AI:** Orchestrating AI agents for complex tasks
- **Robotics:** Swarms of robots and drones for warehouse optimization, search and rescue, environment monitoring, and other use cases
- **Energy and utilities:** Smart grid optimization and load balancing
- **Supply chain:** Optimizing scheduling, planning, routing and supply chain optimization
- **Telecom:** Network optimization and fault detection
- **Healthcare:** Using agents to model actors (individuals, households, professionals)

Drivers

- **Generative AI agents:** Large language models (LLMs) are increasingly augmented with additional capabilities, such as an internal memory and plug-ins to external applications, to implement AI agents. An emerging design pattern is to assemble and combine these LLM-based AI agents into more powerful systems, which is increasing the feasibility of and interest in multiagent systems.
- **Increased decision-making complexity:** AI is increasingly used in real-world engineering problems containing complex systems, where large networks of interacting parts exhibit emergent behavior that cannot be easily predicted. The decentralized nature of multiagent systems makes them more resilient and adaptable to complex decision making.
- **Simulation and multiagent reinforcement learning:** Advances in the realism and performance of simulation engines, as well as the use of new multiagent reinforcement learning techniques, allow for the training of multiagent AI systems in simulation environments, which can then be deployed in the real world.

Obstacles

- **Training complexity:** Multiagent systems are typically harder to train and build than individual AI agents. These systems can exhibit emergent behavior that is hard to predict in advance, which increases the need for robust training and testing.
- **Monitoring and governing multiple agents:** Coordination and collaboration between agents is challenging. Careful monitoring, governance and a common grounding are required to ensure that the combined multiagent system behavior achieves its intended goals.
- **Limited adoption and readiness:** Despite its benefits, the application of multiagent systems to real-world problems is not yet widespread, which creates a lack of enterprise awareness and readiness to implement.
- **Specialized skills required:** Building and deploying multiagent systems requires specialized skills beyond traditional AI skills, particularly the use of reinforcement learning and simulation.
- **Fragmented vendor landscape:** A fragmented vendor landscape inhibits customer adoption and engagement.

User Recommendations

- Use multiagent systems for complex problems that require decentralized decision making and cannot be solved by single AI agents. This includes problems with changing environments where agents need to adapt and problems where a diverse set of agents with different expertise can be combined to accomplish a goal.
- Shift to a multiagent approach gradually since this is an emerging area of research and the risks and benefits are not yet fully understood.
- Establish clear guardrails when implementing multiagent systems, including legal and ethical guidelines around autonomy, liability, robust security measures and data privacy protocols.
- Invest in the use of simulation technologies for AI training, as simulation is the primary environment to build and test multiagent systems.
- Educate your AI teams on multiagent systems, how they differ from single-agent AI design, and some of the available techniques to train and build these systems.

Sample Vendors

Alphabet; Ansys; Cosmo Tech; FLAME GPU; MathWorks; Microsoft; OpenAI; The AnyLogic Company

Gartner Recommended Reading

Innovation Insight: AI Agents

Innovation Insight: AI Simulation

AI Design Patterns for Large Language Models

AI Simulation

Analysis By: Leinar Ramos, Anthony Mullen, Pieter den Hamer, Jim Hare

Benefit Rating: High

Market Penetration: 1% to 5% of target audience

Maturity: Emerging

Definition:

AI simulation is the combined application of AI and simulation technologies to jointly develop AI agents and the simulated environments in which they can be trained, tested and sometimes deployed. It includes both the use of AI to make simulations more efficient and useful, and the use of a wide range of simulation models to develop more versatile and adaptive AI systems.

Why This Is Important

Increased complexity in decision making is driving demand for both AI and simulation. However, current AI faces challenges, as it is brittle to change and usually requires a lot of data. Conversely, realistic simulations can be expensive and difficult to build and run. To resolve these challenges, a growing approach is to combine AI and simulation: Simulation is used to make AI more robust and compensate for a lack of training data, and AI is used to make simulations more efficient and realistic.

Business Impact

AI simulation can bring:

- Increased value by broadening AI use to cases where data is scarce, using simulation to generate synthetic data (for example, synthetic data for generative AI [GenAI])
- Greater efficiency by leveraging AI to decrease the time and cost to create and use complex and realistic simulations
- Greater robustness by using simulation to generate diverse scenarios, increasing AI performance in uncertain environments
- Decreased technical debt by reusing simulation environments to train future AI models

Drivers

- **Limited availability of AI training data is increasing the need for synthetic data techniques, such as simulation.** Simulation techniques, like physics-based 3D simulation, are uniquely positioned to generate diverse AI training datasets. This is increasingly important for GenAI as training data becomes more scarce.
- **Advances in capabilities are making simulation increasingly useful for AI.** Simulation capabilities have been rapidly improving, driven both by increased computing performance and more-efficient techniques.
- **The growing complexity of decision making is increasing interest in AI simulation.** Simulation is able to generate diverse "corner case" scenarios that do not appear frequently in real-world data, but that are still crucial to train and test AI so it can perform well in uncertain environments.
- **Increased technical debt in AI is driving the need for the reusable environments that simulation provides.** Organizations will increasingly deploy hundreds of AI models, which requires a shift in focus toward building persistent, reusable environments where many AI models can be trained, customized and validated. Simulation environments are ideal since they are reusable, scalable and enable the training of many AI models at once.
- **The growing sophistication of simulation drives the use of AI, making it more efficient.** Modern simulations are resource intensive. This is driving the use of AI to accelerate simulation, typically by employing AI models that can replace parts of the simulation without running resource-intensive, step-by-step numerical computations.
- **Research in learned simulations (known as "world models") is driving interest in AI simulation**: Research is increasing on training world models that can learn to predict how the environment will evolve, based on its current state and agents' actions. These learned simulations could make AI simulation more feasible by not having to directly specify simulation parameters.

Obstacles

- **Gap between simulation and reality:** Simulations can only emulate — not fully replicate — real-world systems. This gap will reduce as simulation capabilities improve, but it will remain a key factor. Given this gap, AI models trained in simulation might not have the same performance once they are deployed; differences in the simulation training dataset and real-world data can impact models' accuracy.
- **Complexity of AI simulation pipelines:** The combination of AI and simulation techniques can result in more-complex pipelines that are harder to test, validate, maintain and troubleshoot.
- **Limited readiness to adopt AI simulation:** A lack of awareness among AI practitioners about leveraging simulation capabilities can prevent organizations from implementing an AI simulation approach.
- **Fragmented vendor market:** The AI and simulation markets are fragmented, with few vendors offering combined AI simulation solutions, potentially slowing down the deployment of this capability.

User Recommendations

- Complement AI with simulation to optimize business decision making or to overcome a lack of real-world data by offering a simulated environment for synthetic data generation or reinforcement learning.
- Complement simulation with AI by applying deep learning to accelerate simulation, and generative AI to augment simulation.
- Create synergies between AI and simulation teams, projects and solutions to enable a new generation of more-adaptive solutions for ever-more-complex use cases. Incrementally build a common foundation of more-generalized and complementary models that are reused across different use cases, business circumstances and ecosystems.
- Prepare for the combined use of AI, simulation and other relevant techniques —such as graphs, natural language processing or geospatial analytics — by prioritizing vendors that offer platforms that integrate different AI techniques (composite AI), as well as simulation.

Sample Vendors

Altair; Ansys; The AnyLogic Company; Cosmo Tech; Epic Games; MathWorks; Microsoft; NVIDIA; Rockwell Automation; Unity

Gartner Recommended Reading

Innovation Insight: AI Simulation

Predicts 2023: Simulation Combined With Advanced AI Techniques Will Drive Future AI Investments

Embedding Models

Analysis By: Arun Chandrasekaran

Benefit Rating: High

Market Penetration: 5% to 20% of target audience

Maturity: Emerging

Definition:

Embedding models are machine learning models used to represent high-dimensional data (like text or images) into vector embeddings. These vectors are represented across the multidimensional space, making it easier to perform tasks like semantic search, similarity comparison, clustering and classification on the data, often as part of a retrieval-augmented generation (RAG) architecture.

Why This Is Important

Embedding models enable efficient representation of enterprise data in AI workflows, and boost the quality of information retrieval. In the context of RAG, embedding models play a crucial role by presenting relevant context from a vast amount of information, which is then used to generate coherent and contextually appropriate responses. Embedding models add value across a broad set of NLP use cases such as similarity search, recommendation systems, sentiment analysis and machine translation, as well as computer vision use cases such as image recognition and fault detection.

Business Impact

Embedding models drive business value by allowing generative AI (GenAI) models to use data that isn't part of their original training to deliver accurate responses. They aid in use cases such as personalization, anomaly detection and search, and support better decision making by

leveraging large datasets at scale.

Drivers

- **Growing RAG deployments for GenAI use cases:** As companies increasingly invest in RAG architectures, embedding models become essential. They provide the foundational technology for data representation in RAG workflows, facilitating more accurate and insightful outcomes.
- **Model as a service:** Embedding models are increasingly offered as a managed API by large AI platform providers, including the large cloud hyperscalers, making them more accessible and easy to manage.
- **Data explosion:** With the exponential growth in data generation from various sources like social media, customer transactions and enterprise knowledge management systems, enterprises need efficient ways to process, understand and leverage this vast amount of information. Embedding models help by converting unstructured data into formats that are easier to analyze and act upon by GenAI applications and models, often in conjunction with vector databases.
- **Multimodal data integration:** Modern enterprises often have to process and analyze data from diverse sources and formats, including text, images and video. Embedding models facilitate the integration of these multimodal data sources into a unified representation, enhancing data coherence and the quality of insights.
- **Customer experience enhancement:** Enterprises are keen on improving customer engagement and satisfaction. Embedding models support these goals by enabling personalized recommendations, targeted marketing and customer support automation, all of which rely on a deep understanding of customer and industry data.
- **Scalability of AI applications:** As AI applications grow in complexity and size, embedding models offer a scalable solution that can handle increasing data volumes without a corresponding increase in computational demand, making them ideal for deployment in resource-constrained environments.

Obstacles

- **Data privacy and security concerns:** Handling sensitive information with embedding models — particularly cloud-based models — can introduce security and compliance issues, especially in regulated industries.

- **Data quality and availability:** The quality of embeddings (numerical representation of data) is highly dependent on the quality of enterprise data. Lack of adequate data collection, cleaning and preprocessing can result in suboptimal outcomes from the embedding models.
- **Integration challenges:** Integrating embedding models with existing IT infrastructure and workflows can be complex and disruptive.
- **Scalability issues:** Scaling embedding models to handle enterprise-level data volumes and real-time processing needs can be technically challenging.
- **Cost:** Embedding models add an additional layer of cost for the model API, or the compute and operational overheads of managing the life cycle of the models.
- **Effectiveness:** There could be use cases where using the direct vector representations from an embedding model that has only been generically pretrained may not produce a useful embedding representation of the task. In such scenarios, the embedding models need to be fine-tuned, adding to the costs and complexity and requiring additional know-how.
- **Skills gap:** There is a shortage of skilled professionals who can manage and interpret embedding models, slowing adoption.

User Recommendations

- **Assess data infrastructure:** Ensure the existing data infrastructure can support the demands of embedding models, including data collection, storage, quality and processing capabilities. Consider upgrades if necessary to handle large-scale data efficiently.
- **Prioritize data privacy and security:** Address potential security and privacy concerns from the outset by implementing robust data governance policies and ensuring compliance with relevant regulations.
- **Start small:** Begin with pilot projects to demonstrate the value of embedding models. Select use cases that can show quick wins to build support and provide learning experiences.
- **Leverage cloud APIs:** Consider cloud platforms for their scalability and flexibility. Most GenAI cloud platform providers offer embedding models as an API to consume.
- **Measure impact:** Develop metrics to measure the effectiveness and ROI of embedding model initiatives. Use these metrics to justify further investment and guide the scaling of successful applications.

Sample Vendors

AI21 Labs; Amazon Web Services (AWS); Cohere; Google; Meta; Microsoft; Mistral AI; NVIDIA; OpenAI; Voyage AI

Gartner Recommended Reading

Innovation Guide for Generative AI Models

Innovation Insight: Vector Databases

AI Agents

Analysis By: Haritha Khandabattu, Tom Coshow

Benefit Rating: High

Market Penetration: 1% to 5% of target audience

Maturity: Emerging

Definition:

AI agents are autonomous or semi autonomous software entities that use AI techniques to perceive, make decisions, take actions and achieve goals in their digital or physical environments.

Why This Is Important

AI agents have agency and the ability to reliably adapt, plan and act open-endedly over long time horizons to achieve organizational goals. Current large language models (LLMs) lack the necessary agency to perform even simple tasks, but given increasing investments in AI research, organizations are creating and deploying AI agents to achieve complex tasks.

Business Impact

AI agents have the potential to:

- Revolutionize a broad range of industries and environments with their ability to automate tasks from consumer, industrial, data analytics, content creation and logistics.
- Make informed decisions and interact intelligently with their surroundings.

Drivers

- **GenAI breakthroughs:** Chain of thought using LLMs and large action models (LAMs) advances the ability to plan a complex series of actions.
- **Multimodal understanding:** The ability to use diverse modalities like vision, audio and language enables more general and flexible AI agents. This allows automatic adaptation to changes in the workflow, user interface or API. With this, one can create advanced workflows without explicit programming, significantly reducing the development time and effort for automation.
- **Increased decision-making complexity:** AI is increasingly used in real-world engineering problems containing complex systems, where large networks of interacting parts exhibit emergent behavior that cannot be easily predicted. AI agents can learn, plan and execute in complex environments.
- **Composite AI, including neurosymbolic models:** Advances in models that improve planning and problem solving are enabling more complex AI agents. AI agents can utilize a wide variety of AI practices to forecast, make decisions and plan.

Obstacles

- **Vulnerabilities**: Due to the complexity of the agent system, all components face various potential vulnerabilities.
- **Lack of trust:** Users are unsure whether they can trust the technology to accurately predict and execute tasks independently. Without a human in the loop, agents may take multiple consequential actions in rapid succession and bring about significant impacts before a human notices.
- **Interpretability and oversight**: Action policies may be opaque and have poor explainability, requiring mechanisms for human interpretability, oversight and control.

User Recommendations

- Incorporate AI agents into strategic planning by investing in understanding their capabilities and potential applications in various environments, considering their increasing autonomy and wide-ranging usability.

- Investigate the possibilities of utilizing multiagent systems (MAS), collectives of AI agents, that can operate both collaboratively and independently, enhancing adaptability and flexibility in response to different tasks and scenarios.
- Promote the development and integration of the use of a variety of AI practices, enabling learning, negotiation and decision-making capabilities.

Sample Vendors

AutogenAI; Cognition; crewAI; LangChain; Merlynn Intelligence Technologies; Openstream.AI; Sema4.ai; UiPath; XLANG Lab

Gartner Recommended Reading

Innovation Insight: AI Agents

Emerging Tech: Secure Generative Communication for LLMs and AI Agents

When Not to Use Generative AI

AI Developer Platforms

Analysis By: Jim Scheibmeir, Mike Fang

Benefit Rating: High

Market Penetration: 5% to 20% of target audience

Maturity: Early mainstream

Definition:

AI developer platforms provide the required technology and workflows to design, build, test and deploy AI-embedded applications. These platforms include access to large language models (LLMs) and the ability to ground and guardrail them. Additionally, these platforms provide full life cycle support for responsible AI and use cases such as building AI assistants, avatars and agents.

Why This Is Important

AI developers are being tasked with building new types of applications such as AI agents and avatars, as well as embedding AI into existing enterprise systems. AI developer platforms aid developers by providing a single platform across the life cycle of creating and embedding AI into enterprise solutions. These tools enable enterprises to build skill sets more quickly than using a combination of open-source technology, prebuilt models and APIs across disparate providers.

Business Impact

AI development requires precision, performance and responsible AI as a foundation to enterprise use. AI developer platforms ease the design, creation, testing, evaluation, monitoring and auditability of such technologies. The use cases of AI avatars, AI agents, retrieval-augmented generation (RAG) for chatbots and more will be focal points in enterprise strategies for 2025 and beyond. Each of these use cases offer customers and employees new methods to be successful within their interactions and journeys.

Drivers

- AI is driving product development roadmaps. However, consistency is lacking across the developer skill sets, technology provider capabilities, and open-source models and tools. This hampers the inclusion of AI into enterprise solutions at scale. Furthermore, AI may create unacceptable risk by introducing bias and hallucinations into enterprise workflows. Developers require tooling across the design, build, test, and deploy workflow with a foundation of responsible AI features.

- AI developer platforms enable key AI trends:

  - **AI agents:** Autonomous or semiautonomous software entities that use AI techniques to perceive, make decisions, take actions and achieve goals in their digital or physical environments. AI agents will enable productivity across employees, consumers and customers, as well as offer transformation of business models.

  - **AI avatars:** Humanlike virtual personas created using various AI techniques and applications, like natural language processing, synthetic voice, computer vision and video translation, among others. AI avatars can be a representation of a real person or a physical entity to represent the brand or support interactions.

  - **AI assistants:** AI assistants boost employee efficiency, minimize cognitive load, amplify problem solving, accelerate learning pace, foster creativity and maintain their state of flow. They leverage orchestration framework to enable process-driven decision support, personalization, contextualization and domain-specific knowledge.

- **Opportunities to capitalize on new insights:** The wealth of data from both internal and third-party sources delivers insights such as the incorporation of predictive machine learning models that enable data-driven decision intelligence in applications.
- **Support demand for conversational interactions:** The emergence of generative AI and LLMs facilitates conversationally enabled applications where users can use LLMs with data sources to gain insights.

Obstacles

- **Specialized skills:** Building and deploying AI agents, assistants and avatars require the use of new technologies and specialized skills beyond traditional software development.
- **Monitoring and governing AI solutions:** Coordination between agents, assistants and avatars with enterprise systems is challenging. Careful monitoring, governance and a common grounding are required to ensure that the combined system behavior achieves intended goals.
- **Limited readiness:** Despite its benefits, the application of multiagent, avatars and even AI assistants has not made the leap from novel technology to high-priority business requirements.
- **Lack of understanding by developers:** There is a gap in knowledge on how to adapt these AI agents and avatars to specific business and enterprise use cases.
- **Grounding generative AI models:** The need to ground generative AI models is challenging, requiring well-crafted RAG solutions.
- **Pricing models:** Usage-based pricing models for these platforms and the underlying infrastructure for inference present a risk for businesses as the costs can accrue rapidly.

User Recommendations

- Opt for AI developer platforms or those with the best ecosystem instead of a combination of disparate vendors, LLMs and AI services.
- Increase the likelihood of success in your AI strategy by conducting experiments with AI techniques.
- Ensure that generative AI models are loosely coupled to adapt to the rapidly evolving technology.

- Utilize AI developer platforms across common use cases such as RAG for chatbots, AI avatars and AI agents, to embed AI throughout employee and customer-facing technologies.
- Empower your software engineers with AI developer platform functionalities like automated algorithm selection, dataset preparation, and more across the MLOps and ModelOps life cycles.
- Employ pretrained generative AI models to facilitate swift prototyping and deployment of LLM-enabled solutions.
- Leverage the responsible AI features within AI developer platforms, including guardrails, to protect your investment, brand and reputation while leveraging novel AI technologies.

Sample Vendors

Alibaba Cloud; Amazon Web Services; Google; Huawei; H2O.ai; IBM; LangChain; Microsoft; Tencent

Gartner Recommended Reading

Critical Capabilities for Cloud AI Developer Services

Magic Quadrant for Cloud AI Developer Services

Market Opportunity Map: Cloud Infrastructure and Platform Services, Worldwide

Voice of the Customer for Cloud AI Developer Services

Hype Cycle for Artificial Intelligence, 2024

AI Security Code Assistants

Analysis By: Mark Horvath

Benefit Rating: Transformational

Market Penetration: 1% to 5% of target audience

Maturity: Emerging

Definition:

Artificial intelligence (AI) code security assistants (ACSAs) are technologies that help developers identify and remediate security vulnerabilities in code. To do so, they offer auto-remediation suggestions, and direct code assistance or chatbots, using forms of AI, such as generative AI (GenAI). ACSAs are primarily delivered as features of application security testing (AST) products. They use cloud services to analyze code and make recommendations.

Why This Is Important

Developers are not typically well-trained to identify and resolve security issues in their code. Code assistants offer an alternative. When trained on secure coding data, the large language models (LLMs) offered by AST vendors take complex ideas and format them for maximum clarity. Developers write better secure code when they get effective answers in a security context. Although most organizations supply answers through developer coaches, LLMs and ASCAs offer more-scalable, security-centric code advice.

Business Impact

- ASCAs go beyond more-general ASCAs by offering just-in-time (JIT) solutions to the specific security problems found by AST tools.
- Unlike most tools, ASCAs may be in a unique position to optimize several variables (e.g., security and code quality) at the same time.
- ASCAs are next-generation tools for closing one of the remaining gaps — JIT training and security remediation — in shifting security left in the modern, secure software development life cycle (SDLC).

Drivers

- Code assistants, such as GitHub Co-Pilot or Google Vertex, are growing rapidly as they become increasingly popular with developers. Developers are becoming accustomed to AI assistants working with them in multiple areas of their code.
- Shift-left initiatives — that is, moving work from the runtime environment (the right) to the developer (left) — are often slow to be adopted or fail completely. This is because the channels needed to supply developers with solid security guidance are limited and fragile. Initiatives often

rely on experts to guide the organization. ASCAs offer an opportunity to add virtual experts at many layers of the organization in a way that is easily consumed by developers.

- AST vendors, which have traditionally supplied tools for AppSec, have offered security code assistance for more than a decade, and have access to the core security coding data their tools generate. Using this data as a training corpus, and combining them with foundational models into ASCAs, was a logical next step.

- Developers believe that ASCAs will provide reliable security advice. A 2024 Checkmarx survey found that a surprising 70% of developers expressed high or medium confidence in the tools.

Obstacles

- Most organizations worry about exfiltrating private data or intellectual property (IP) through AI assistants. There are ways to protect against this, but trust is low.

- A recent **Stanford University study** found that developers using security assistants ended up making more mistakes after adopting them than before.

- GenAIs occasionally hallucinate, identifying problems that are not real or are wrong, or giving explanations that sound plausible, but are unsecure or unwise to use. This can be hard to detect, especially for junior developers.

- Code ownership issues are a concern, especially if an ASCA has contributed a significant amount of code. Given that the assistants are generally using code they were trained on, rather than creating original code, accidentally acquiring another organization's IP is a concern.

- Overcompensating for security, while ignoring issues such as performance, typifies how these kinds of optimizers have failed in the past. There is concern this could happen with ASCAs.

User Recommendations

- Continue to use traditional AST tools — e.g., static AST (SAST) or SCA — to measure security and code quality issues when adding an ASCA. Although it's common to see an immediate increase in security metrics (e.g., code quality), watch the other indicators of code quality.

- Ask vendors about how they handle privacy and confidentiality issues with your code and the code used to train the model.

- Retain and adjust existing methods of improving code security, as ASCAs become a bigger part of the developer experience.

- Continue programs such as security coaches and code reviews so they act as a balance to ensure that the advice developers receive is correct and helpful.

Sample Vendors

Checkmarx; GitHub; GitLab; HCL; Mend.io; Mobb.AI; Qwiet AI; Semgrep; Snyk, Synopsys; Veracode

Gartner Recommended Reading

Emerging Tech: Generative AI Code Assistants Are Becoming Essential to Developer Experience

Innovation Insight for Generative AI

Invest Implications: Emerging Tech: Generative AI Code Assistants Are Becoming Essential to Developer Experience

Magic Quadrant for Application Security Testing

Neuro-Symbolic AI

Analysis By: Erick Brethenoux, Afraz Jaffri

Benefit Rating: High

Market Penetration: Less than 1% of target audience

Maturity: Emerging

Definition:

Neuro-symbolic AI is a form of composite AI that combines machine learning (ML) methods and symbolic systems (for example, knowledge graphs) to create more robust and trustworthy AI models. This fusion enables the combination of probabilistic models with explicitly defined rules and knowledge to give AI systems the ability to better represent, reason and generalize concepts. This approach provides a reasoning infrastructure for solving a wider range of business problems more effectively.

Why This Is Important

Neuro-symbolic AI is important because it addresses limitations in current AI systems, such as incorrect outputs, lack of generalization to a variety of tasks and an inability to explain the steps that led to an output. This leads to more powerful, versatile and interpretable AI solutions and allows AI systems to tackle more complex tasks with humanlike reasoning.

Business Impact

Neuro-symbolic AI will have an impact on the efficiency, adaptability and reliability of AI systems used across business processes. The integration of logic and multiple reasoning mechanisms brings down the need for ever larger AI models and their supporting infrastructure. These systems will rely less on the processing of huge amounts of data, making AI agile and resilient. Neuro-symbolic approaches can augment and automate decision making with less risk of unintended consequences.

Drivers

- Limitations of AI models based exclusively on ML techniques that focus on correlation over understanding and reasoning. The newest generation of large language models is well-known for its tendency to give factually incorrect answers or produce unexpected results. Neuro-symbolic AI addresses these limitations.
- The need for explanation and interpretability of AI outputs that are especially important in the regulated industry use cases and in systems that use private data.
- The need to prioritize understanding the meanings behind words, not just their arrangement (semantics over syntax) in systems that deal with real-world entities to ground meaning to words and terms in specific domains.
- The set of tools available to combine different types of AI models is increasing and becoming easier to use for developers, data scientists and end users. The dominant approach is to chain together results from different models (composite AI) rather than using single models.
- The integration of multiple reasoning mechanisms necessary to provide agile AI systems eventually leads to adaptive AI systems, notably through blackboardlike mechanisms.

Obstacles

- Most neuro-symbolic AI methods and techniques are being developed in academia or industry research labs. Despite the increased availability of tools, there are still limited implementations in business or enterprise settings.
- There are no agreed-upon techniques for implementing neuro-symbolic AI, and disagreements continue between researchers and practitioners on the effectiveness of combining approaches, despite the emergence of real-world use cases.
- The commercial and investment trajectories for AI startups allocate almost all capital to deep learning approaches, leaving only those willing to bet on the future to invest in neuro-symbolic AI development.
- Currently, popular media and academic conferences do not give as much exposure to the neuro-symbolic AI movement as compared to other approaches.

User Recommendations

- Adopt composite AI approaches when building AI systems by utilizing a range of techniques that increase the robustness and reliability of AI models. Neuro-symbolic AI approaches will fit into a composite AI architecture.
- Dedicate time to learning and applying neuro-symbolic AI approaches by identifying use cases that can benefit from these approaches.
- Invest in data architecture that can leverage the building blocks for neuro-symbolic AI techniques, such as knowledge graphs and agent-based techniques.
- Consider neuro-symbolic AI architectures when the limitations of generative AI models prevent their implementation in the organization.

Sample Vendors

Elemental Cognition; Franz; Google DeepMind; IBM; Microsoft; RelationalAI; Wolfram|Alpha

Gartner Recommended Reading

Innovation Insight: AI Simulation

Go Beyond Machine Learning and Leverage Other AI Approaches

When Not to Use Generative AI

Predicts 2023: Simulation Combined With Advanced AI Techniques Will Drive Future AI Investments

GraphRAG

Analysis By: Afraz Jaffri

Benefit Rating: Moderate

Market Penetration: Less than 1% of target audience

Maturity: Adolescent

Definition:

GraphRAG is a technique to improve the accuracy, reliability and explainability of retrieval-augmented generation (RAG) systems. The approach uses knowledge graphs (KGs) to improve the recall and precision of retrieval, either directly by pulling facts from a KG or indirectly by optimizing other retrieval methods. The added context refines the search space of results, eliminating irrelevant information.

Why This Is Important

RAG techniques in an enterprise context suffer from problems related to the veracity and completeness of responses caused by limitations in the accuracy of retrieval, contextual understanding and response coherence. KGs, a well-established technology, can represent data held within documents and the metadata relating to the documents. Combining both aspects allows RAG applications to retrieve text based on the similarity to the question and contextual representation of the query and corpus, improving response accuracy.

Business Impact

Enterprise generative AI applications based on the RAG architecture can benefit with decreased time to value. Currently, moving applications from 70% to 80% accuracy onwards requires multiple iterations and tuning of parameters. KGs can decrease the number of iteration cycles. KGs are key to supporting AI-ready data. Organizing data in a KG has added benefits beyond RAG implementation for metadata management, catalogs, accessibility and semantic representation.

Drivers

- Standard RAG approaches fail to meet the needs for many enterprise use cases where high thresholds of accuracy need to be guaranteed.
- There is an increasing interest in KGs for data management, semantic enrichment, knowledge representation and data integration.
- An increasing focus on the combination of symbolic and nonsymbolic AI models to achieve higher levels of comprehension, reasoning and explainability have driven the adoption of GraphRAG.
- The addition of vector search capabilities inside KG platforms and databases eases implementation difficulties of integrating separate vector and KG retrieval.
- Graph databases are already an established part of many enterprises' data landscape, providing a natural starting point for RAG use-case implementation.
- The flexibility of graph data models provides a method for answering questions that can be retrieved by structured queries, reducing the need for large language models (LLMs) with high token input and output costs.

Obstacles

- The creation of KGs for use in GraphRAG architectures requires a high level of expertise in data and knowledge modeling. LLMs are helping to augment the process, but still need to be guided by an expert.
- GraphRAG architectures have not been tested at scale for performance and scalability.
- The number of different RAG approaches is increasing rapidly, causing confusion and indecision on which method or approach to use for long-term viability.
- The evaluation of GraphRAG, as with other RAG approaches, is difficult and requires custom metrics to be created.
- Using a GraphRAG approach requires high-quality data and additional metadata, which may not be present in sufficient quantities for solutions that require immediate delivery.

User Recommendations

- Include GraphRAG as an approach for building RAG systems by creating services that can easily transform data into KGs using associated metadata.
- Utilize the GraphRAG approach for nontrivial RAG use cases, or for those where error tolerance levels are low.
- Use domain experts to verify and enhance the construction of KGs for use in GraphRAG systems by creating semantic layers that include domain-specific terminology.
- Expand the existing methods for metadata extraction and management to include KG models that provide context for downstream applications.

Sample Vendors

Diffbot; Franz; LlamaIndex; Neo4j; Semantic Web Company; TigerGraph; Vesoft

Gartner Recommended Reading

AI Design Patterns for Knowledge Graphs and Generative AI

How Large Language Models and Knowledge Graphs Can Transform Enterprise Search

Demystifying Taxonomies, Ontologies and Data Models

Design to Code

Analysis By: Frank O'Connor, Brent Stewart

Benefit Rating: Transformational

Market Penetration: 1% to 5% of target audience

Maturity: Emerging

Definition:

Design-to-code tools automatically generate near-production-quality user interface (UI) code from sketches and/or screen designs. By automating work that would have otherwise required a developer, efficiency is improved. Design-to-code tools also facilitate closer collaboration between UI designers and developers by enabling designers to iteratively deliver snippets of near-production-quality code, which can be integrated and tested earlier.

Why This Is Important

When user experience (UX) designers and front-end developers work in silos, collaboration is difficult. When key UX decisions are handed off as images or documents, much can be lost in translation to code (for example, details like microinteractions or UI control behaviors). Design-to-code tools help teams deliver near-production-ready front-end code from UI designs. Teams using design-to-code tools are more productive, more aligned around design decisions, and have fewer design miscommunications.

Business Impact

- **Improved UX:** Design-to-code tools enable designers and developers to work iteratively. Generating UI code from designs means that coded UIs can be delivered in small batches, which can be integrated and tested sooner.
- **Better productivity:** Teams using design-to-code tools are more efficient as they can automatically generate UI code from screen designs. These tools can also scan design tools for similarly styled elements and create common classes, reducing dependencies and producing cleaner code.

Drivers

Two adjacent technology innovations are underpinning successful design-to-code solutions:

- **Design systems:** Increasing adoption of design systems by leading product-led organizations is a significant boost for design-to-code solutions. These tools rely on design systems to be able to generate UI code, which is based on established UX standards and practices, and reflect the organization's established brand and UI style.
- **Generative design AI:** Text-to-image AI tools like Midjourney and Adobe Firefly have demonstrated impressive ability to generate images from text prompts. Also of note is the emergence of GPT-enabled design-to-code tools in which a large language model (LLM) can take natural

language descriptions and generate editable screen designs. Design-to-code will continue to use generative AI to produce more sophisticated UIs based on combinations of design files and conservational style prompts.

Other drivers for design to code are:

- **Speed:** Design to code significantly improves development cycle time by eliminating design handovers between designers and developers, and by delivering near-production-ready front-end code generated from UI designs.
- **Consistency:** Design to code enables the creation of consistent UIs across disparate teams by using a standard, consistent procedural method of generating UI code from designs.
- **Usability:** UIs that are generated from design-to-code tools are typically built in accordance with time-tested, established product types and UI design patterns that are familiar to most users.
- **Democratization:** Design to code helps bridge the gap between professional and nonprofessional developers, such as designers, researchers and citizen developers. Design-to-code tools enable nonprofessional developers to create near-production-quality front ends by creating the UIs in design tools and relying on design-to-code tools to generate the code.

Obstacles

- **Maintenance considerations:** Designers will most likely lack the technical knowledge of how the existing codebase and technology stack are constructed. Therefore, they may unknowingly create UIs that are difficult to maintain or expensive to operate.
- **Implementation drift:** Design-to-code tools will attempt to generate code close to the provided design, but in most cases, there will not be a one-to-one mapping between the design and the implementation. Code produced by a design-to-code tool is close to the design, but it may have some minor differences that can be frustrating and time-consuming to correct.
- **Limited UI framework support:** The majority of the design-to-code tools support leading popular UI frameworks, such as Angular and React, but if your organization's UI stack is nonstandard you may not be able to use design-to-code tools easily.

User Recommendations

- Conduct a thorough design-to-code tool evaluation and selection process. Make sure to involve UX designers and front-end developers in this process to enable a sense of ownership and ensure future adoption of the selected tool.
- Choose a tool that allows you to operationalize your existing design system or UI code component library.
- Educate your team about the business value of design to code to decrease development time and increase quality.

Sample Vendors

Anima; Builder.io; Clutch; Figma; Framer; H2O.ai; Infragistics; Locofy.ai; Overlay; TeleportHQ

Gartner Recommended Reading

Predicts 2024: Generative AI Is Reshaping Software Engineering

Quick Answer: What is Design-to-Code?

Quick Answer: How Can We Incorporate User-Centric Design Into the Features We Build for Our Product?

Multimodal Generative AI

Analysis By: Danielle Casey, Roberta Cozza

Benefit Rating: Transformational

Market Penetration: 1% to 5% of target audience

Maturity: Emerging

Definition:

Multimodal generative AI (GenAI) is the ability to combine multiple types of data inputs and outputs in generative models, such as images, videos, audio, text and numerical data. Multimodality augments the usability of AI by allowing models to interact with and create outputs across various modalities.

Why This Is Important

Multimodal GenAI is important because data in the real world is typically multimodal. Multimodality helps capture the relationships between different data streams and scales the benefits of GenAI across potentially all data types and applications. This allows AI to support humans in performing more tasks, regardless of the environment. Multimodal functionality will increasingly be present in tech offerings, as foundation models lower the technical barrier to multimodal functionality and as users demand associated performance.

Business Impact

Multimodal GenAI will have a transformational impact on enterprise applications by enabling the addition of new features and functionality otherwise unachievable. The impact of multimodality is not limited to specific industries or use cases, and can be applied at any touchpoint between AI and humans. Today, many multimodal models are limited to two or three modalities, though this will increase over the next few years to include more modalities. The future of AI is multimodal.

Drivers

Multimodal GenAI is being driven by:

- **Improved technology capability:** Since 2023, more generative AI models have come to market that support various modalities. This is lowering the technology barrier for providers to offer multimodal functionality in their offerings. As GenAI matures, multimodal capabilities will be supported across more applications.
- **Improved user experience (UX):** Multimodality improves UX by enabling richer experiences, by meeting users with the data that is available. The ability to interact with both unstructured and structured data is supporting additional use cases within the organization. Multimodality allows users to interact with, manipulate and create outcomes from numerous data types. Most data in reality is multimodal.
- **Extensibility of automation:** Multimodality supports new tasks, such as data extraction, converting one data type to another, and creating new data outcomes. Consequently, applications that support multimodality will have a higher automation potential.

Obstacles

Multimodal GenAI models can be inhibited by:

- **Training challenges:** Multimodal GenAI models use deep learning, data alignment and fusion techniques to train, integrate and analyze data sourced from the multiple modalities. Multimodal data has varying degrees of quality and formats compared to unimodal data, amplifying challenges with training cost and time, output accuracy and de-bias from multimodal data sources.
- **Lack of data:** Data availability may be limited in some modalities. For example, availability of large-scale audio datasets is more limited compared to other modalities like images and text. This impacts model training and accuracy.
- **Data governance:** Multimodal GenAI increases the exposure to a wider range of sensitive data. Examples of particularly sensitive data types include maps or geolocation data, biometric data or health data. Also, regulations and standards are a work in progress and are lagging GenAI's capability advancement.

User Recommendations

- Identify the two or three modalities that are most important to your organization when choosing which GenAI model to use.
- Assess the technical complexities of processing and integrating data inputs and outputs from diverse multimodal sources. Then, validate early on how these can best be integrated with key legacy or more current workflows.
- Create specific cross-modal data policies and tools aimed at protecting privacy, detecting bias and ensuring compliance to emerging AI regulations.

Sample Vendors

Google; Meta; Microsoft; NVIDIA; OpenAI; Stability AI; Twelve Labs

Gartner Recommended Reading

Emerging Tech: The Key Technology Approaches That Define Generative AI

Emerging Tech: Multimodal Generative AI Interfaces Transform User Experiences

Emerging Tech Impact Radar: Generative AI

At the Peak

Vector Databases

Analysis By: Arun Chandrasekaran, Radu Miclaus

Benefit Rating: High

Market Penetration: Less than 1% of target audience

Maturity: Embryonic

Definition:

Vector databases store numerical representations of data. In such databases, each point is represented by a vector with a fixed number of dimensions, which can be compared via mathematical operations, such as distance measures. Vector databases are commonly used in machine learning (ML) solutions, where vectors represent data features/attributes, such as text embeddings. Storing these vectors in a database enables users to search for similar data points at scale with low latency.

Why This Is Important

Vector databases serve use cases such as similarity search and product recommendation. Emerging approaches in generative AI (GenAI) have made vector databases often part of retrieval-augmented generation (RAG), a popular way of grounding AI with enterprise data. When customers adopt GenAI models, vector databases store the embeddings for fast retrieval. With the indexed and stored vector embeddings, the database can do a similarity search, which matches a prompt (the question) with specific or similar vector embedding.

Business Impact

Vector databases enable digital business by enhancing semantic search, personalizing recommendations and supporting AI applications with efficient management of unstructured data. They enable deeper insights, scalable data handling and better accuracy of AI, crucial for data-intensive operations. They can be valuable in leveraging data for competitive advantage, particularly in industries where data volume, complexity, and the need for precision in search and analysis are growing exponentially.

Drivers

- **Popularity of vector embeddings in GenAI for grounding large language models:** The surge in AI and ML applications across industries necessitates databases that can efficiently handle complex, high-dimensional data. With the rise of AI foundational models, embeddings have become the cornerstone for semantic search and RAG architectures. Hence, they are the working inputs for grounding large foundation models with enterprise data.
- **Advanced search and personalization needs:** As users demand more personalized experiences and relevant search results, enterprises need databases capable of semantic search and personalization at scale. Vector databases enable searching based on the meaning behind data, rather than just matching keywords, allowing businesses to deliver personalized content and product recommendations, thus enhancing user engagement and satisfaction.
- **Performance and scalability needs:** The applications looking to embed generative methods that use embeddings-based models need back-end services that can respond with low latency to high-concurrency requests (prompts) and responses (completions) for GenAI use cases.
- **Regulatory compliance and data security:** As regulatory requirements around data privacy and security tighten globally, enterprises must ensure their data management practices comply with laws. Vector databases can offer better control over data access, processing and storage, helping enterprises navigate the complex landscape of data privacy regulations more effectively.
- **Developer focus:** Developers of new applications are driving the demand for vector databases by presenting use cases that cannot scale without the ability for embeddings to be stored in an optimized structure for high-throughput production applications.

Obstacles

- Enterprises lack an understanding of what vector databases do and the unique use cases they enable. Moreover, know-how of data chunking and retrieval optimization is needed to ensure vector databases deliver value within GenAI applications.
- Vector databases are superspecialized databases that may cause challenges around data migration, integration and limited extensibility across use cases, although many large database vendors are now starting to offer vector support in existing databases.
- Most vector databases are delivered as cloud-managed services. The complexity of deploying, configuring and operating them outside cloud environments requires deep technical skills and know-how.
- Vector databases can be expensive to implement, given the newness of the technology and lack of industry skills on ongoing optimization.

- The vector database market is nascent and populated mostly by startups, which may not have extensive experience working with enterprise clients or have proven product-market fit. Also, it is uncertain whether this will be a separate market category in the long run.

User Recommendations

- Determine whether your functional requirements can be satisfied by incumbent vendors that can support the storage and retrieval of vector embeddings — you may not always need a purpose-built vector database.
- Prioritize developer experience, ecosystem integration, use-case fit, reliability and performance as important selection criteria, and validate them thoroughly via a proof-of-concept process.
- Select managed, cloud-based vector databases as deployment modes, unless you have stringent requirements and deep technical skills for an on-premises, self-managed deployment mode.
- Conduct internal training and education on the appropriate use cases for vector databases, how to leverage their true potential, and effective ways to optimize their deployment and maximize their value.

Sample Vendors

Amazon Web Services (AWS); DataStax; Google; KX; Microsoft; MongoDB; Pinecone Systems; Qdrant; Redis; Weaviate

Gartner Recommended Reading

Innovation Insight: Vector Databases

A CTO's Guide to the Generative AI Technology Landscape

How to Choose an Approach for Deploying Generative AI

Innovation Guide for Generative AI Models

AI Governance

Analysis By: Svetlana Sicular

Benefit Rating: High

Market Penetration: 5% to 20% of target audience

Maturity: Emerging

Definition:

AI governance is the process of creating policies, assigning decision rights, and ensuring organizational accountability for risks and decisions for the application and use of AI techniques. AI governance is part of adaptive data and analytics governance, addressing the predictive and generative capabilities of AI.

Why This Is Important

With AI delivering value in the enterprise, scaling AI without governance is ineffective and dangerous. Effective AI governance is essential for balancing the business value of AI with appropriate oversight. Generative AI models and applications introduce new AI governance challenges, such as intellectual property and copyright protection. AI draws the attention of legislators worldwide, who mandate actions by clarifying AI governance priorities.

Business Impact

AI governance, as part of the organizational governance structure, enacts responsible AI and provides a common operating model when it comes to:

- Trust and transparency to support AI adoption via explainability, bias mitigation, model governance, operationalization and collaboration norms and capabilities. Transparency of governance decisions gains importance in the light of AI regulations.
- Diversity to ensure the right technology and roles for each AI project.
- Ethics, fairness and safety to protect the business and its reputation.

Drivers

- AI governance is in the peak area of the Hype Cycle. Enterprise practitioners are taking aggressive steps to establish AI governance. Organizations in various industries address standards for AI development and operations, providing guidelines and best practices for model management and monitoring, data labeling and interpretation, explainability, fairness, bias mitigation, security, and legal.

- Regulations around the globe target AI directly and affect AI practices indirectly, making AI governance goals more concrete. The objective of the EU AI Act is to "enhance governance and effective enforcement of existing law on fundamental rights and safety requirements applicable to AI systems." The U.S. Executive Order on the Safe, Secure, and Trustworthy Development and Use of Artificial Intelligence calls for governance across, at a minimum, the areas of information technology infrastructure, data, workforce, leadership and risk management. The Algorithmic Impact Assessment is a mandatory risk assessment tool intended to support the Treasury Board of Canada. Singapore's Model AI Governance Framework guides organizations in developing appropriate governance structures and mechanisms.

- The probabilistic and opaque nature of AI is new to audiences used to deterministic outcomes. AI governance can minimize misinterpretations of AI results by scrutinizing trust in data sources and the explainability of AI decisions. It provides specific testing and validation guidelines, differentiating "life-critical AI." Trust and transparency of AI solutions are crucial for AI adoption.

- AI governance organizations are establishing AI accountability, which is challenging because use cases differ in terms of their data, solution and outcome requirements. For example, accountability could include roles, actions and procedures in the case of unanticipated and unintended consequences. AI governance also ensures that ethics are considered for each use case.

Obstacles

- AI governance is often siloed from mainstream governance initiatives, which stalls its progress. The best method is to extend existing governance mechanisms to take advantage of recognizable policies and methods, such as in data governance. AI governance benefits from a conversation with the security, legal and customer experience functions.

- Many governance initiatives assume command and control. Instead, adaptive governance supports freedom and creativity in AI teams but also protects the organization from reputational and regulatory risks. Little or no governance in AI teams to facilitate freedom and creativity is an acceptable approach if this is a governance decision.

- AI value assurance and model risk management are new in AI. While methods exist — for example, in the financial industry — they are largely unknown to others, and every governance organization is inventing its own.

- Technologies to support AI governance are fragmented and are often designed for a single industry.

User Recommendations

- Extend to AI your existing governance mechanisms, such as risk management or data and analytics governance.
- Establish and refine processes for handling AI-related business decisions.
- Aim to align your AI governance framework with the laws and regulations in your jurisdiction(s) to directionally assure your efforts amid evolving AI-specific considerations. Gain agreement on AI risk guidelines that are driven by the business risk appetite and regulations.
- Decide on the organizational structure and accountability for propagating responsible AI — for example, what to centralize and what to do locally.
- Implement tools for AI review and validation. For each AI use case, require an independent AI model validator. Have all parties in the process defend their decisions in front of their peers and validators.
- Ensure that a feedback loop is in place to allow users to report issues and mitigate deficiencies of AI automation.

Sample Vendors

Collibra; Credo AI; FICO; Google; Holistic AI; IBM; Monitaur; Prodago; Weights & Biases; YOOI

Gartner Recommended Reading

Artificial Intelligence Requires an Extended Governance Framework

Applying AI — Governance and Risk Management

Quick Answer: The EU AI Act and Its Anticipated Impact

The Impact of the 'U.S. Executive Order on AI'

Generative Design AI

Analysis By: Brent Stewart

Benefit Rating: Transformational

Market Penetration: 5% to 20% of target audience

Maturity: Adolescent

Definition:

Generative design AI is a technology that utilizes AI to autonomously generate design options based on parameters and constraints specified by the user. This AI uses algorithms to iterate rapidly through numerous variations, optimizing designs to achieve the best possible outcomes while adhering to predefined goals and improving efficiency in the design process.

Why This Is Important

Generative design AI has emerged in several markets, including digital product design tools, architectural design and industrial design. Generative design AI has led to major leaps in designer productivity, quality and skills enhancement. The technology already exists as feature-level support for user experience (UX) designers and front-end developers (such as intelligent design suggestions) and will transition rapidly to full digital product design and front-end development capabilities.

Business Impact

In a future powered by generative design AI, UX design and front-end code will be generated in hours or days, rather than weeks or months. The resulting designs will be based on established design principles and patterns that ensure maximum usability, usefulness and accessibility. In this future, UX practitioners will shift from low-value design production tasks to higher-value activities such as design curation, user and market research, and product strategy.

Drivers

- **Productivity —** While not all UX work will benefit equally from generative design AI, it is expected to lead to significant productivity gains, specifically for UX designers and front-end developers. These increases are expected to be similar to productivity gains in other areas such as coding, in which software engineers have more than doubled productivity using AI coding assistants.

- **Accessibility —** AI-generated screen designs and front-end code are expected to adhere more strongly to Web Content Accessibility Guidelines standards, leading to more accessible products for people with disabilities.
- **Democratization —** Nonprofessional (citizen) designers, researchers and front-end developers are beginning to produce high-quality user experiences without deep design training or the help of a UX designer. This trend is expected to accelerate rapidly as generative design AI becomes standard in professional-grade digital product design tools like Figma, Sketch and UXPin.
- **UI design standardization —** The overwhelming majority of digital products are based on established product types and UI design patterns. In general, the standardization of common digital experiences continues to expand, further fueling the potential of generative design AI for digital products.
- **Rapid advancement of key technologies —** The following technologies will accelerate the quality, effectiveness and adoption of generative design AI: multimodal generative AI, machine learning, cloud computing, simulation and finite element analysis, optimization algorithms and parametric modeling.

Obstacles

- **Design systems —** In UX, generative design AI is dependent on a quality design system as the source of style elements, templates, patterns and components needed to generate a UI.
- **Cost —** Generative design AI is a heavy lift that requires deep talent, long time frames and/or deep pockets to license established models.
- **Jobs —** Generative design AI drastically reduces low-level UX production tasks, potentially reducing the need for some production designers, front-end developers and UX writers. Many organizations may struggle to adapt to a more-efficient staffing model.
- **Originality —** Since generative design AI pulls from established product types and design patterns, it will not be notable for its originality. Many UX practitioners are concerned that user experiences will become too uniform and lack originality.
- **Laws and ethics —** AI algorithms may have trained on copyrighted or private data or contain inherent gender, cultural and selection biases.

User Recommendations

- Assess developments in generative design AI, specifically at Adobe, Canva, Figma and in the low-code/no-code market.

- Prepare digital product teams for the emergence of generative design AI — first through design-to-code technology followed by tools that produce high-fidelity screen designs and UX writing.
- Transition the role of humans in the design process from production-level creators to strategic curators only after generative design AI advancements in key platforms, like Figma.

Sample Vendors

Adobe; Builder.ai; Builder.io; Canva; Figma; Locofy.ai; TeleportHQ; Uizard Technologies

Gartner Recommended Reading

How Generative AI Will Change User Experience

Emerging Tech: Multimodal Generative AI Interfaces Transform User Experiences

9 Regulatory and Legal Implications of Generative AI

Prompt Engineering

Analysis By: Frances Karamouzis, Jim Hare, Afraz Jaffri

Benefit Rating: High

Market Penetration: 5% to 20% of target audience

Maturity: Emerging

Definition:

Prompt engineering is the discipline of providing inputs, in the form of text or images, to generative AI (GenAI) models to specify and confine the set of responses the model can produce. The inputs prompt a set that produces a desired outcome without updating the actual weights of the model (as done with fine-tuning). Prompt engineering is also referred to as “in-context learning,” where examples are provided to further guide the model.

Why This Is Important

Prompt engineering is the linchpin to business alignment for desired outcomes. It is important because large language models (LLMs) and GenAI models in general are extremely sensitive to nuances and small variations in input. A slight tweak can change an incorrect answer to one that is usable as an output. Each model has its own sensitivity level, and the discipline of prompt engineering is to uncover the sensitivity through iterative testing and evaluation.

Business Impact

Prompt engineering has the following business impacts:

- **Performance:** It helps improve model performance and reduce hallucinations.
- **Business alignment:** It allows subject data scientists, subject matter experts and software engineers to steer foundation models, which are general-purpose in nature, to align to the business, domain and industry.
- **Time to market, quality, efficiency and effectiveness:** There are a number of architecture options as well as execution options that AI leaders must balance. There is also a myriad of prompt optimization tools that will diminish (or at the very least shift) the need for manual engineering.

Drivers

- **Balance and efficiency:** The fundamental driver for prompt engineering is it allows organizations to strike a balance between consuming an “as is” offering versus pursuing a more expensive and time-consuming approach of fine-tuning. GenAI models, and in particular LLMs, are pretrained, so the data that enterprises want to use with these models cannot be added to the training set. Instead, prompts can be used to feed content to the model with an instruction to carry out a function.
- **Process or task-specific customizations or new use cases:** The insertion of context and patterns that a model uses to influence the output generated allows for customizations for a particular enterprise or domain, or regulatory items. Prompts are created to help improve the quality for different use cases — such as domain-specific question answering, summarization, categorization, and so on — with or without the need for fine-tuning a model, which can be expensive or impractical. This would also apply to creating and designing new use cases that utilize the model’s capability for image and text generation.

- **Validation and verification:** It is important to test, understand and document the limits and weaknesses of the models to ensure a reduced risk of hallucination and unwanted outputs.

Obstacles

- **Prompt engineering is a new discipline:** The craft of designing and optimizing user requests to an LLM or LLM-based chatbot to get the most effective result is still emerging. Engineers are finding that desired outputs using GenAI can be challenging to create, debug, validate and repeat. Communities worldwide are developing new prompt engineering methods and techniques to help achieve these desirable outcomes.
- **Approaches, techniques and scalability:** A unified approach to performing prompt engineering does not exist. Complex scenarios need to be broken down into smaller elements. It is challenging to debug complex prompts. Understanding how specific prompt elements influence the logic of the LLM is vital. Scalable and maintainable methods of prompt engineering are still a work in progress for most organizations.
- **Role alignment:** Data scientists are critical to understanding the capabilities and limits of models, and to determining whether to pursue a purely prompt-based or fine-tuning-based approach (or combination of approaches) for customization. The ultimate goal is to use machine learning itself to generate the best prompts and achieve automated prompt optimization. This is in contrast to an end user of an LLM who concentrates on prompt design to manually alter prompts to give better responses.

User Recommendations

- Build awareness and understanding of prompt engineering to quickly start the journey of shape-shifting the appropriate prompt engineering discipline and teams.
- Build critical skills among different team members that will synergistically contribute critical elements. For example, there are important roles for data scientists, business users, domain experts, software engineers and citizen developers.
- Educate the team with the myriad of options of prompt optimization tools that will diminish (or at the very least shift) the need for manual engineering.
- Communicate and cascade the message that prompt engineering is not foolproof. Enterprise teams apply rigor and diligence to permeate and work to ensure successful solutions.

Sample Vendors

FlowGPT; Google; HoneyHive; Magniv; Microsoft; PromptBase; Salesforce

Gartner Recommended Reading

How to Engineer Effective Prompts for Large Language Models

Prompt Engineering With Enterprise Information for LLMs and GenAI

Quick Answer: How Will Prompt Engineering Impact the Work of Data Scientists?

Generative AI Changes Software Engineering Leaders' Responsibilities

Retrieval-Augmented Generation

Analysis By: Radu Miclaus

Benefit Rating: High

Market Penetration: 5% to 20% of target audience

Maturity: Emerging

Definition:

Retrieval-augmented generation (RAG) is an architecture design pattern that uses search functionality to retrieve relevant data and add it to the prompt of a generative AI (GenAI) model in order to ground the generative output with factual information. RAG can be used for both retrieving public internet data and retrieving data from private knowledge bases.

Why This Is Important

The majority of available large language models (LLMs) are trained on general public information. As a result, information necessary to respond to a specific prompt request, rather than a general question, is not available. Additionally, even if the information was part of the original training set, the response may not focus on the correct information source, increasing the chance of inaccurate information. The correct RAG implementation minimizes hallucinations.

Business Impact

Traditionally, workers spend 20% to 30% of their time looking for the information needed to complete business tasks (based on widely reported surveys). RAG is used to enhance how information is summarized and created, and can minimize hallucinations and accelerate productivity. The applications for self-service content consumption for employees and for customer-facing applications will have a significant impact on employee and customer satisfaction, brand equity and workforce productivity.

Drivers

- The current worker experience of searching and finding current information and knowledge documented in enterprises can be frustrating. Workers will benefit from a better information retrieval and synthesis approach.
- The advances of generative AI applied toward content summarization and contextualization has enabled search applications to achieve a more robust and focused content consumption experience.
- GenAI service vendors, either via a hyperscale marketplace or via specialty model APIs, are making it easier for organizations to configure the RAG pattern on their knowledge bases. Many more pilot projects are moving to production.
- Competitive pressure to adopt the latest innovations in order to increase productivity and maintain competitive edge is a driver for organizations to use their knowledge bases to support the RAG pattern for both internal- and external-facing applications.
- Availability of technologies for building RAG as well as RAG being baked in the GenAI assistant experiences is penetrating enterprises.

Obstacles

- The need for making knowledge bases available for retrieval is not new. Underinvestment in enterprise search has been the norm for years. Companies that are still building the discipline and skill sets for building robust retrieval and search to support the RAG pattern will experience friction during adoption.
- Configuration of knowledge base access controls for the RAG pattern is not a trivial task and can add obstacles in widespread adoption across all organizational knowledge.

- More complex RAG architectures, like knowledge graph-based retrieval, will be needed to complement hybrid retrieval (keyword and vector search). This challenge is compounded as multimodal documents enter the enterprise.
- Concerns about IP protection and responsible AI in the use of LLMs will remain obstacles in the adoption of RAG-based applications.
- Vendor RAG support tools are still immature.

User Recommendations

IT and data and analytics leaders looking at adopting GenAI capabilities on top of private and public corporate data should:

- Prioritize RAG investments by assessing the maturity and readiness of the enterprise in relation to knowledge management and quality and information retrieval for employees and customers.
- Pilot applications using the RAG pattern on a well-known knowledge base to assess the lift in the content consumption experience and to gain buy-in for further investment.
- Plan for investment in filling any skills gaps that exist in internal knowledge engineering capabilities, either through upskilling or external hiring.
- Engage with technology vendors or service providers that combine both technology and services to accelerate adoption of RAG architecture, including data, vector database, graph database and LLM vendors.

Sample Vendors

Amazon Web Services; Charli AI; Glean; Google; Microsoft

Gartner Recommended Reading

Getting Started With Retrieval-Augmented Generation

Quick Answer: How to Supplement Large Language Models With Internal Data

Prompt Engineering With Enterprise Information for LLMs and GenAI

AI Code Assistants

Analysis By: Arun Batchu, Philip Walsh, Haritha Khandabattu, Matt Brasier

Benefit Rating: Transformational

Market Penetration: 5% to 20% of target audience

Maturity: Adolescent

Definition:

AI code assistants, integrated into developer tools, use pretrained models to interact with software developers through natural language and code snippets. They generate, analyze, debug, fix, refactor code, create documentation and translate code between languages. These assistants can be customized to an organization's specific code base and documentation.

Why This Is Important

AI code assistants, unlike their predecessors, enhance developer velocity by predicting and autocompleting multiple lines of code in real time. They increase understanding and expedite problem-solving by explaining and debugging code issues. Beyond merely completing code, these assistants foster a conversational interaction with developers, facilitating the exploration of multiple problem solutions. Productivity expectations are very high, yet challenges with the technology and inability of developers to exploit them fully are limiting their value proposition.

Business Impact

AI code assistants:

- Enhance work environments by reducing stress from repetitive tasks and boosting team morale and job satisfaction.
- Enable developers to focus on innovative solutions by automating mundane tasks.
- Expedite skill improvement for junior developers and facilitate reskilling for experienced engineers.

- Automate routine tasks, enabling focus on complex software development and boosting productivity without staff increase.

Drivers

- Advances in foundational large language models (LLMs) and multimodal models are accelerating the capability of code assistants.
- AI code assistants, previously available only to developers in technology companies, are now accessible to all.
- Enterprises, desperate for ways to increase the productivity of developers, are finally able to justify the cost of rolling out these tools to all their development staff.
- Developers are rapidly adopting these tools in their personal projects, as quickly as their employers make them available.
- Vendors of these assistants are increasing, with the hope of monetizing the demand.
- Competition is driving rapid innovation, fueling the adoption rate.
- Non-computer-science or non-software-engineering majors are gaining software development skills, opening up new talent pools that otherwise were not available for enterprises.

Obstacles

- Much of training data is from open-source software projects, which may contain problematic code with security, intellectual property, toxicity and bias issues.
- Customers fear that vendors will use their prompts and conversations to improve models.
- Regional AI regulations could prevent organizations from adopting these tools, or force technological compromises that decrease the tools' effectiveness.
- Rapid productivity gains in coding activities might get reduced by the overall inefficiency of an organization's software development life cycle.
- Return on investment is minimal without additional training and upskilling for developers to maximize the efficacy of code assistants.
- Trusting the generated code and text without verification may lead to an increase in the amount of poor quality and hard-to-maintain code.

- AI code assistant adoption demands prioritizing long-term gains, often resisted by CFOs favoring immediate ROI.

User Recommendations

- Trial the tools and toolchain to catch downstream errors by forming a pilot team of engineers, testers, security staff and compliance experts.
- Refine vendor options by considering factors such as private network needs, licensing requirements and enterprise code context.
- Scale deployment by developing a rollout plan.
- Remove bottlenecks from the software development life cycle value stream to maximize ROI on coding productivity gains by using software engineering intelligence tools.
- Achieve prompt engineering quality across the enterprise by creating a small, highly skilled team that develops and publishes prompt engineering patterns and best practices.
- Create a community of practice for users to share prompts and tips on how to use the assistants.
- Develop a learning and development plan for developers to master the tools.
- Maintain flexibility by being prepared to switch vendors in the short term, in case it becomes untenable to use your primary vendor.

Sample Vendors

Alibaba; Amazon; Exafunction (Codeium); GitHub; GitLab; Google; Huawei Technologies; IBM; Sourcegraph; Tabnine

Gartner Recommended Reading

Innovation Guide for AI Code Assistants

Assessing How Generative AI Can Improve Developer Experience

Quick Answer: How Can Generative AI be Used to Improve Testing Activities?

How to Communicate the Value of AI Code Assistants

How Platform Engineering Teams Can Augment DevOps With AI

AI Engineering

Analysis By: Soyeb Barot, Anthony Mullen, Leinar Ramos, Joe Antelmi

Benefit Rating: High

Market Penetration: 1% to 5% of target audience

Maturity: Adolescent

Definition:

AI engineering is the foundation for enterprise delivery of AI and generative AI (GenAI) solutions at scale. The discipline unifies DataOps, MLOps and DevOps pipelines to create coherent enterprise development, delivery (hybrid, multicloud, edge), and operational (streaming, batch) AI-based systems.

Why This Is Important

The demand for AI solutions has dramatically increased, driven by the unrelenting hype surrounding GenAI. Few organizations have built the data, analytics and software foundations needed to move individual pilot projects to production at scale, much less operate portfolios of AI solutions at scale. There are significant engineering, process and culture challenges to address. To meet the demands for scaling AI solutions, enterprises must establish consistent AI pipelines supporting the development, deployment, reuse, governance and maintenance of AI models (statistical, machine learning, generative, deep learning, graph, linguistic and rule-based).

Business Impact

AI engineering enables organizations to establish and grow high-value portfolios of AI solutions consistently and securely. Most AI developments are currently limited by operational and cultural bottlenecks. With AI engineering approaches — DataOps, ModelOps and DevOps — it is possible to deploy models into production in a structured, repeatable factory-model framework.

Drivers

- DataOps, ModelOps and DevOps provide best practices for moving artifacts through the AI development life cycle. Standardization across data and model pipelines accelerates the delivery of AI solutions.
- The elimination of traditional siloed approaches to data management and AI engineering doubles the data engineering effort and reduces impedance mismatches across data ingestion, processing, model engineering and deployment, which inevitably drift once the AI models are in production.
- AI engineering enables discoverable, composable and reusable AI artifacts (data catalogs, knowledge graphs, code repositories, reference architectures, feature stores, model stores and others) across the enterprise context. These are essential for scaling AI enterprisewide.
- AI engineering makes it possible to orchestrate solutions across hybrid, multicloud, edge AI or Internet of Things.
- Broader use of foundational platforms provides initial success at scaling the production of AI initiatives with existing data, analytics and governance frameworks.
- AI engineering practices, processes and tools must be adapted to address GenAI. GenAI specific adaptations include support for prompt engineering, vector DBs/graph KBs, architecting and deploying multiagent, and interactive deployment models.
- AI engineering tools can be subdivided into model-centric and data-centric tools. Terms such as DataOps, LLMOps, LangOps or FMOps, or more broader terms such as ModelOps or MLOps, are used frequently, but we believe they are all a subset of AI engineering.

Obstacles

- Sponsorship for foundational enterprisewide AI initiatives is unclear. The transformational promise of AI enablement has led executives to actively compete for enterprise AI responsibility.
- AI engineering needs simultaneous development of pipelines across domains and platform infrastructure maturity.
- AI engineering requires integrating full-featured solutions with specific tools, including open-source technologies, to address enterprise architecture gaps with minimal functional overlap. These include gaps around extraction, transformation and loading stores, feature stores, model stores, model monitoring, pipeline observability, and governance.

- AI engineering requires cloud maturity and possible rearchitecting, or the ability to integrate data and AI model pipelines across deployment contexts. Potential complexity and management of analytical and AI workloads alongside costs may deter organizations that are in the initial phases of AI initiatives.
- Enterprises often seek “unicorn” experts to productize AI platforms. Spot-fix vendor solutions will bloat costs and potentially complicate already intricate integration and model management tasks.

User Recommendations

- Establish a leadership mandate for enterprisewide foundational AI initiatives.
- Maximize business value from ongoing AI initiatives by establishing AI engineering practices that streamline the data, model and implementation pipelines.
- Simplify data and analytics pipelines by identifying the capabilities required to operationalize end-to-end AI platforms and build AI-specific toolchains.
- Use point solutions sparingly and only to plug feature/capability gaps in fully featured DataOps, MLOps, ModelOps and PlatformOps tools.
- Develop AI model management and governance practices that align model performance, human behavior and delivery of business value to make it easier for users to adopt AI models.
- Leverage cloud service provider environments as foundational to build AI engineering. At the same time, rationalize your data, analytics and AI portfolios as you migrate to the cloud.
- Adopt a platform approach to GenAI by investing in centralized AI engineering tools for automation, governance and use-case enablement across a broad set of AI models and providers.
- Upskill data engineering and platform engineering teams to adopt tools and processes that drive continuous integration/continuous development for AI artifacts.

Sample Vendors

Amazon Web Services; Dataiku; DataRobot; Domino Data Lab; Google; Microsoft; neptune.ai; OctoAI; Seldon Technologies; Weights & Biases

Gartner Recommended Reading

Top Strategic Technology Trends for 2022: AI Engineering

Demystifying XOps: DataOps, MLOps, ModelOps, AIOps and Platform Ops for AI

A CTO's Guide to Top Artificial Intelligence Engineering Practices

Cool Vendors in AI Core Technologies — Scaling AI in the Enterprise

Case Study: AI Model Operations at Scale (Fidelity)

AI-Augmented Software Engineering

Analysis By: Arun Batchu, Manjunath Bhat, Oleksandr Matvitskyy, Jim Scheibmeir

Benefit Rating: Transformational

Market Penetration: 5% to 20% of target audience

Maturity: Early mainstream

Definition:

AI-augmented software engineering (AIASE) is the application of artificial intelligence technologies to assist software engineers throughout the software development life cycle. This includes the creation, validation, securing, deployment and maintenance of applications.

Why This Is Important

The software development life cycle involves routine tasks in both creative and operational DevOps loops. AI automation minimizes manual effort, freeing engineers to innovate, reduce technical debt, improve quality, security and team collaboration, and lower operational cost. AI technologies augment engineers' cognitive tasks such as analyzing logs, optimizing configurations, and generating scripts, code, unit tests and documentation.

Business Impact

Engineers using AI technologies, such as knowledge graphs, reinforcement learning, retrieval augmented generation and GPT-powered models trained on various data types, are more efficient and creative. AIASE accelerates application delivery by enabling software engineers to solve business problems and release solutions to production faster. AIASE helps engineers increase the nonfunctional quality properties of software such as security, performance and runtime costs.

Drivers

The drivers for demand include:

- The increasing complexity of software systems to be engineered.
- The increasing demand for developers to deliver high-quality code faster.
- The increasing number of security risks associated with software development.
- The need to optimize operational costs.

Drivers for providing the necessary technology solutions include:

- The application of AI models to prevent application security vulnerabilities by detecting them and suggesting fixes.
- The increasing impact of software development on business.
- The application of foundation models such as large language models (LLMs) to software code generation and optimization.
- The application of deep learning models to software operations.

Obstacles

- The hype surrounding this innovation has led to misconceptions and unrealistic expectations about the advantages of AIASE.

- Awareness about the tools ready for production is limited.
- Often, there's a lack of clarity about the origin and use of data for model training.
- Solutions are uneven and fragmented, automating only some tasks in the software development life cycle, such as code creation.
- There's a shortage of AI skills like prompt engineering, training, tuning, maintaining and troubleshooting models.
- High costs are associated with fine-tuning, training and inference of models at scale.
- Intellectual property risks arise from models trained on nonpermissive licensed code.
- Privacy issues arise from proprietary data and leaked code as training data for AI models.
- Technical employees fear that job automation by AI will lead to redundancy.

User Recommendations

- Pilot, measure and roll out tools only if there are clear gains.
- Innersource best practices, including examples of the best prompts to use for utilizing AIASE technology.
- Verify the maintainability of AI-generated artifacts, including executable requirements, code, tests and scripts.
- Track this rapidly evolving and highly impactful market to identify new products that minimize development toil and improve the experience of software engineers, such as those that ease security and site operations burdens.
- Reassure software engineers that AIASE is an augmentation toolset for human engineers, not a replacement.
- Choose providers (including open-source vendors) that provide visibility into training data and transparency on how the model was trained.
- Establish the correct set of metrics, such as new release frequency and ROI, to measure the success of AIASE.

Sample Vendors

Amazon Web Services; Anima; CAST; Dynatrace; GitHub; GitLab; SeaLights; Sedai; ServiceNow; Veracode

Gartner Recommended Reading

How Platform Engineering Teams Can Augment DevOps With AI

Innovation Guide for AI Code Assistants

Top Strategic Technology Trends for 2024: AI-Augmented Development

Top Strategic Technology Trends for 2024: AI-Augmented Development

Market Guide for AI-Augmented Software-Testing Tools

Predicts 2024: Generative AI Is Reshaping Software Engineering

Predicts 2024: Generative AI Is Reshaping Software Engineering

GenAI-Driven API Generation

Analysis By: Max van den Berk

Benefit Rating: Moderate

Market Penetration: 1% to 5% of target audience

Maturity: Adolescent

Definition:

API generation assists developers in automatically creating APIs based on code or specifications. API generation features are usually included in stand-alone API creation tools and low-code platforms, and recently in generative AI (GenAI)-powered tools. Benefits of API generation include decreased development time, consistency and security.

Why This Is Important

Gartner sees GenAI as fundamentally changing software delivery. GenAI assistance with API generation is becoming an expected capability in the software development process as it helps developers speed up their delivery (see **2024 Planning Guide for Software Development**).

Business Impact

APIs are a well-understood and common way of integrating between applications. The options for monetization that APIs offer are an important part of their business appeal. API generation offers faster design and delivery with possibilities of better consistency and governance. Gartner predicts AI coding assistants will drive developer productivity up 36% by 2028. GenAI can further speed up API generation (see **How to Communicate the Value of AI Code Assistants**).

Drivers

- The increase in pace of delivery heightens the demands on integration teams that develop and support integrations between applications.
- The increased availability of cloud-based services and API-centric SaaS in part drives a need for wider support of API types.
- Software development tools, integrated developer environments and platforms as a service all support some API generation capabilities that are accelerated by the use of GenAI.

Obstacles

- API generation has limits of extrapolation. “Garbage in, garbage out” remains true and its effects may be amplified by the addition of GenAI.
- Stakeholder involvement in the usability, correctness and effectiveness of APIs is not sped up by generative processes, but it remains vital in creating APIs that have actual demand. Skipping stakeholder involvement risks increasing unnecessary churn on APIs.
- Large language models (LLMs) are still striving to prove their reliability. Lack of confidence in GenAI-generated code can result in extensive validation processes. These time-consuming efforts offset productivity gains.
- Approaches such as retrieval-augmented generation (RAG) are needed to infuse an organization’s API inventory and API governance into the GenAI engagement. Enterprises must select products that support tactics such as including API standards and guidelines in prompts or as part of LLM training data.

User Recommendations

- Ensure guardrails for API generation by making generated APIs part of life cycle management processes that enable governance.
- Limit approaches within the organization that circumvent efforts to shift left in the development process.
- Consider other available options for generating APIs and the producer or consumer code. While GenAI is driving interest, existing options such as API management, low-code platforms or integration platforms may be more dependable.
- Choose a solution (such as RAG) to make the organization's API inventory, standards, examples and documentation accessible to the selected GenAI technology. This enables GenAI responses (suggested APIs, code to invoke APIs, etc.) to reflect the enterprise's APIs and comply with its API governance.
- Implement strong API security and protection that monitors all API requests and responses, including API requests from GenAI-created code for noncompliant and suspicious API activity.

Synthetic Data

Analysis By: Arun Chandrasekaran, Anthony Mullen, Alys Woodward

Benefit Rating: High

Market Penetration: 5% to 20% of target audience

Maturity: Adolescent

Definition:

Synthetic data is a class of data that is artificially generated rather than obtained from direct observations of the real world. Synthetic data is used as a proxy for real data in a wide variety of use cases, including data anonymization, AI and machine learning (ML) development, data sharing, and data monetization.

Why This Is Important

A major problem with AI development today is the burden involved in obtaining real-world data and labeling it. This time-consuming and expensive task can be remedied with synthetic data, where data can be generated faster and cheaper. Additionally, for specific use cases such as training models for autonomous vehicles, collecting real data for 100% coverage of edge cases is practically impossible. Furthermore, synthetic data can be generated without personally identifiable information (PII) or protected health information (PHI), making it a valuable technology for privacy preservation.

Business Impact

Adoption is increasing across various industries. Gartner predicts a massive increase in adoption as synthetic data:

- Avoids using PII when training ML models via synthetic variations of original data or synthetic replacement of parts of data.
- Reduces cost and saves time in ML development.
- Improves ML performance as more training data leads to better outcomes.
- Enables organizations to pursue new use cases for which very little real data is available.
- Is capable of addressing fairness issues more efficiently.

Drivers

- In healthcare and finance, buyer interest is growing as synthetic tabular data can be used to preserve privacy in AI training data.
- To meet the increasing demand for synthetic data for natural language automation training, especially for chatbots and speech applications, new and existing vendors are bringing new offerings to market. This is expanding the vendor landscape and driving synthetic data adoption.
- Synthetic data applications have expanded beyond automotive and computer vision use cases to include data monetization, external analytics support, platform evaluation and the development of test data.
- Transformer and diffusion architectures, the architectural foundations for generative AI (GenAI), are enabling synthetic data generation at quality and precision not seen before. AI simulation techniques are improving the quality of synthetic data by better recreating real-world representations.

- There is an expansion to other data types. While tabular, image, video, text and speech applications are common, R&D labs are expanding the concept of synthetic data to graphs. Synthetically generated graphs will resemble, but not overlap the original. As organizations begin to use graph technology more, we expect this method to mature and drive adoption.
- The growing adoption of GenAI models and future customizations of such models will drive the demand for synthetic data to pretrain these models.

Obstacles

- Synthetic data can have bias problems, miss natural anomalies, be complicated to develop or not contribute any new information to existing, real-world data.
- Data quality is tied to the model that generates the data.
- There are no clear best practices on how to combine synthetic and real data for AI development.
- Synthetic data generation methodologies lack standardization.
- It is difficult to validate the accuracy of synthetic data. While a synthetic dataset may look realistic and accurate, it is difficult to know for sure if it accurately captures the underlying real-world environment.
- Buyers are still confused over when and how to use the technology due to the lack of skills.
- Synthetic data can still reveal a lot of sensitive details about an organization, so security and privacy are concerns. An ML model could be reverse-engineered via active learning. With active learning, a learning algorithm can interactively query a user (or other information sources) to label new data points with the desired outputs, meaning learning algorithms can actively query the user or teacher for labels.
- If fringe or edge cases are not part of the seed dataset, they will not be synthetized. This means the handling of such borderline cases must be carefully accommodated.
- There may be a level of user skepticism as data may be perceived to be “inferior” or “fake.”

User Recommendations

- Identify areas in your organization where data is missing, incomplete or expensive to obtain, and is thus currently blocking AI initiatives. In regulated industries, such as healthcare or finance, exercise caution and adhere to rules.
- Use synthetic variations of the original data, or synthetic replacement of parts of data, when personal data is required but data privacy is a requirement.
- Educate internal stakeholders through training programs on the benefits and limitations of synthetic data. Institute guardrails to mitigate challenges such as user skepticism and inadequate data validation.
- Measure and communicate the business value, success and failure stories of synthetic data initiatives.

Sample Vendors

Anonos (Statice); Gretel; Hazy; Howso; MOSTLY AI; Parallel Domain; Rendered.ai; Tonic.ai; YData

Gartner Recommended Reading

Innovation Guide for Generative AI Models

Case Study: Enable Business-Led Innovation with Synthetic Data (Fidelity International)

Predicts 2024: The Future of Generative AI Technologies

ModelOps

Analysis By: Joe Antelmi, Soyeb Barot, Erick Brethenoux

Benefit Rating: High

Market Penetration: 1% to 5% of target audience

Maturity: Emerging

Definition:

Model operationalization (ModelOps) is primarily focused on the end-to-end governance and life cycle management of advanced analytics, AI and decision models, such as models based on machine learning (ML), generative AI (GenAI), knowledge graphs, rules, optimization, linguistics, agents and others.

Why This Is Important

ModelOps helps companies in standardizing, scaling and augmenting their analytics and AI initiatives. It helps organizations to move their models from the lab environments into production. MLOps primarily focuses on monitoring and governance of ML models, while ModelOps assists with the operationalization and governance of all advanced analytics, decision and AI models, including GenAI and retrieval-augmented generation (RAG) systems.

Business Impact

ModelOps, as a practice:

- Provides the capability for the management and operationalization of diverse AI, analytics and decision systems.
- Enables the complex subsystems required for AI, analytics and decision system observability including versioning, monitoring, automation, data orchestration, experimentation, and explainability.
- Ensures collaboration among a wider business, development and deployment community, and the ability to associate AI, analytics and model outcomes with business KPIs.

Drivers

- Modern AI systems are being built with a symbiotic combination of generative and classic AI models, agents and intelligent software capabilities. As the number of advanced analytics, AI and decision models at organizations increases, organizations will have to manage different types of prepackaged or custom-made models in production.
- Organizations want to be more agile and responsive to changes within their advanced analytics and AI pipelines not only with models but also with data, application and infrastructure.

- ModelOps provides a framework to separate responsibilities across various teams for how models (including GenAI, foundational models, analytics, ML, physical, simulation, symbolic, etc.) are built, tested, deployed and monitored across different environments (for example, development, test and production). This enables better productivity and collaboration, and it lowers failure rates.
- ModelOps provides tools to address model degradation via drift, and bias. In other scenarios, enabling model governance, explainability and integrity is paramount.
- The operationalization challenges of ML models are not new, but the capability to enable diverse models in production at the organization level using ModelOps is still evolving.
- Organizations don't want to deploy an unlimited number of open-source offerings to manage ModelOps, but there are few comprehensive solutions that provide end-to-end capabilities in every domain of model operationalization. Moreover, not every capability is required immediately. Often, versioning, monitoring and model orchestration precede the full implementation of feature stores, pipelines and observability.
- GenAI will require an increased focus on testing, and the introduction of capabilities to version, manage and automate prompts, routers, and retrieval-augmented generation systems. Fine-tuning will also require enhanced ModelOps capabilities to manage complex domain and function training datasets.

Obstacles

- Organizations using different types of models often don't build the right ops, governance and management capabilities until they already have a chaotic landscape of unmanaged advanced analytic systems.
- Not all analytical techniques currently benefit from mature operationalization methods. Because the spotlight has been on ML techniques, MLOps benefits from a more evolved AI practice, but some models, like agentic modeling and optimization techniques, require more attention in ModelOps practices and platforms.
- ModelOps capabilities that help productionize GenAI are emerging but immature. Moreover, organizations are struggling to get GenAI into production, due to data, security and regulatory concerns.
- Organizations may adopt ModelOps platform capabilities that they don't immediately need. At the same time, organizations that are siloed and fail to adopt a comprehensive ModelOps strategy create redundancy in effort with respect to operationalization.

User Recommendations

- Buy ModelOps capabilities integrated into your primary AI platforms. Enrich these capabilities with best-of-breed open-source or proprietary ModelOps offerings where unique problems, like feature stores or observability, require enhanced solutions.
- Utilize ModelOps best practices across composite AI, data, models and applications to ensure transition, reduce friction and increase value generation.
- Recruit/upskill additional engineers who can master ModelOps on AI systems that utilize unstructured data, search, graph and optimization.
- Encourage collaboration between development and deployment teams, and empower teams to make decisions to automate, scale and bring stability to the analytics and AI pipeline.
- Collaborate with software engineering teams to scale ModelOps. Offloading operationalization responsibilities to production support teams enables increased ModelOps specialization and sophistication across the ecosystem of complex AI-enabled applications.

Sample Vendors

DataRobot; IBM; ModelOp; Modzy; Neptune.ai; OctoAI; SAS; Valohai; Verta; Weights & Biases

Gartner Recommended Reading

Launch an Effective Machine Learning Monitoring System

Innovation Guide for Generative AI in Trust, Risk and Security Management

The Logical Feature Store: Data Management for Machine Learning

Operationalize Machine Learning by Using Gartner's MLOps Framework

Toolkit: Delivery Metrics for DataOps, Self-Service Analytics, ModelOps and MLOps

Sliding into the Trough

AI-Augmented Testing

Analysis By: Joachim Herschmann, Jim Scheibmeir

Benefit Rating: High

Market Penetration: 5% to 20% of target audience

Maturity: Adolescent

Definition:

AI-augmented testing comprises AI- and machine learning (ML)-based technologies and practices to make software testing activities more independent from human intervention. It continuously improves testing outcomes by learning from the data collected from performed activities. It extends traditional test automation beyond the automated execution of test cases to include fully automated planning, creation, maintenance and analysis of tests.

Why This Is Important

Software engineering leaders seeking to release faster without degrading quality are looking for more efficient ways of testing across all phases of the software life cycle. AI-augmented testing enables the automation of a broad set of testing activities related to the quality of requirements, design quality, code quality, release quality and operational resilience. This increases the degree of autonomy of those activities.

Business Impact

The adoption of AI-augmented testing has the potential to significantly improve an IT organization’s ability to serve and delight its customers. It can enable the fine-tuning of scenarios for testing as part of a continuous quality strategy aimed at optimizing the end-user experience. It will also help to constitute a closed-loop system that quickly provides continuous feedback about critical quality indicators and helps to reduce the costs of creating and maintaining tests.

Drivers

- A high dependency on human expertise and interaction limits how quickly modern digital businesses can design, build and test new software.

- Where automated testing is already in place, current levels of automation often remain below expectations due to a continued dependency on human intervention to maintain the automation as applications under test (AUT) evolve.
- The pressure to innovate quickly for market differentiation without compromising on quality relies on both an increased velocity and a higher degree of autonomy of the related activities.
- Product teams struggle to deal with the increasing complexity of applications, leading to increased cognitive overload. Complex architectures increasingly require an understanding of an array of elements including cloud-native architecture, replacement of technologies, use of microservices, and support for multiple frontends and AI-powered services.
- Increased adoption of agile and DevOps results in a faster development and delivery cadence, but also comes with additional responsibilities.
- Existing work backlogs to replace manual tests with automated tests that support continuous delivery of software are ever-increasing.
- There is a shortage of skilled test automation engineers to close these automation backlogs.
- Businesses want to reduce test operation and maintenance costs associated with traditional tools and open-source system (OSS) solutions.
- Businesses are aiming to improve the user experience of testing tools so quality engineers can be more productive and avoid mistakes.
- Compliance regulations such as GDPR for data privacy and WCAG 2.1 Level AA for accessibility are enhanced by AI-augmented testing.

Obstacles

- Currently available tools are still relatively new, have a narrow scope and still need to prove their value. Large language model (LLM)-based generative AI, in particular, is the latest and most disruptive example. Hallucinations (content that is nonsensical or untruthful in relation to certain sources), subpar training data, potential copyright violations and security issues are the main risks associated with LLM-based AI technologies.
- Waiting until better AI-augmented testing solutions are available leads to a loss in competitive advantage and fewer innovations. It also incurs greater testing costs and the risk of undertesting.
- Underestimating the time required to acquire new skills and setting wrong expectations about the time required to become successful can be obstacles.

- Gathering, cleaning and processing data and training the model are not trivial tasks, and require adequate skills. Moreover, they are not yet autonomous processes.

User Recommendations

- Set the right expectations about the potential and limitations of AI-augmented testing and ensure that humans are always in the loop to verify the results produced by AI-augmented testing tools. This is particularly relevant for tools employing generative AI to automatically create tests, as generated tests may be completely useless or result in false positives or negatives.
- Start evaluating AI-augmented testing tools now to understand the current possibilities and limitations of these products. Build a roadmap to solve the development organization's most pressing quality challenges.
- Increase the value of AI-augmented testing tools by exploring additional use cases beyond core test automation scenarios, which limit automation primarily on the execution of tests. For example, look for shift-left scenarios such as generating test scenarios from requirements or from user stories and contextual information contained within the codebase (including code and documentation).

Sample Vendors

ACCELQ; Applitools; Avo Automation; CodiumAI; Diffblue; Functionize; mabl; ProdPerfect; testRigor

Gartner Recommended Reading

Market Guide for AI-Augmented Software-Testing Tools

Quick Answer: How Can AI Provide Benefits for Software Testing?

Innovation Insight: Continuous Quality

Improve Software Quality by Building Digital Immunity

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

Climbing the Slope

Knowledge Graphs

Analysis By: Afraz Jaffri

Benefit Rating: High

Market Penetration: 5% to 20% of target audience

Maturity: Early mainstream

Definition:

Knowledge graphs are machine-readable representations of the physical and digital worlds. They include entities (people, companies and digital assets) and their relationships, which adhere to a graph data model — a network of nodes (vertices) and links (edges/arcs).

Why This Is Important

Knowledge graphs capture information about the world in a visually intuitive format yet are still able to represent complex relationships. Knowledge graphs act as the backbone of a number of products, including search, smart assistants and recommendation engines. Knowledge graphs support collaboration and sharing, exploration and discovery, and the extraction of insights through analysis. Generative AI models can be combined with knowledge graphs to provide context for more accurate outputs in a technique becoming known as GraphRAG or G-RAG.

Business Impact

Knowledge graphs can drive business impact in a variety of different settings, including:

- Digital workplace (such as collaboration, sharing and search)
- Automation (such as ingestion of data from content to robotic process automation)
- Machine learning (such as augmenting training data)
- Investigative analysis (such as law enforcement, cybersecurity and risk management)

- Digital commerce (such as product information management and recommendations)
- Data management (such as metadata management, data cataloging and data fabric)

Drivers

- The need to complement AI and machine learning methods that detect only patterns in data (such as the current generation of foundation models) with the explicit knowledge, rules and semantics provided by knowledge graphs.
- The desire to make better use of unstructured data held in documents, correspondence, images and videos, using standardized metadata that can be related and managed and provide the foundation for AI-ready data.
- The increased usage of knowledge graphs with large language models (LLMs) to provide enhanced contextual understanding when answering questions on large quantities of enterprise data.
- The increasing awareness of the use of knowledge graphs in consumer products and services, such as smart devices and voice assistants, chatbots, search engines, recommendation engines and route planning.
- The emerging landscape of Web3 applications and the need for data access across trust networks, leading to the creation of decentralized knowledge graphs to build immutable and queryable data structures.
- The need to manage the increasing number of data silos where data is often duplicated, and where meaning, usage and consumption patterns are not well-defined.
- The use of graph algorithms and machine learning to identify influencers, customer segments, fraudulent activity and critical bottlenecks in complex networks.

Obstacles

- Awareness of knowledge graph use cases is increasing, but business value and relevance are difficult to capture in the early implementation stages.
- Moving knowledge graph models from prototype to production requires engineering and system integration expertise. Methods to maintain knowledge graphs as they scale — to ensure reliable performance, handle duplication and preserve data quality — remain immature.

- The graph DBMS market is fragmented along three properties: type of data model (Resource Description Framework or property), implementation architecture (native or multimodal) and optimal workload (operational or analytical). This fragmentation continues to cause confusion and hesitation among adopters.
- Organizations want to enable the ingestion, validation and sharing of ontologies and data relating to entities (such as geography, people and events). However, making internal data interoperable with external knowledge graphs is a challenge.
- In-house expertise, especially among subject matter experts, is lacking, and identifying third-party providers is difficult. Often, expertise resides with vendors of graph technologies. Skills in scalability and optimization are also hard to acquire.

User Recommendations

- **Create a working group of knowledge graph practitioners and sponsors** by assessing the skills of data and analytics (D&A) leaders, practitioners and business domain experts. Factors like use case requirements, data characteristics, scalability expectations, query flexibilities and domain knowledge of knowledge graphs should be addressed.
- **Run a pilot to identify use cases that need custom-made knowledge graphs.** The pilot should deliver not only tangible value for the business, but also learning and development for D&A staff.
- **Create a minimum viable subset that can capture the information of a business domain to decrease time to value.** Assess the data, both structured and unstructured, needed to feed a knowledge graph, and follow Agile development principles.
- **Utilize vendor and service provider expertise** to validate use cases, educate stakeholders and provide an initial knowledge graph implementation.
- **Include knowledge graphs within the scope of D&A governance and management.** To avoid perpetuating data silos, investigate and establish ways for multiple knowledge graphs to interoperate and extend toward a data fabric.

Sample Vendors

Cambridge Semantics; Diffbot; eccenca; Fluree; Neo4j; Ontotext; Stardog; TigerGraph; TopQuadrant

Gartner Recommended Reading

[How to Build Knowledge Graphs That Enable AI-Driven Enterprise Applications]

[3 Ways to Enhance AI With Graph Analytics and Machine Learning]

[How Large Language Models and Knowledge Graphs Can Transform Enterprise Search]

Data Engineering

Analysis By: Robert Thanaraj, Ehtisham Zaidi, Roxane Edjlali

Benefit Rating: High

Market Penetration: More than 50% of target audience

Maturity: Early mainstream

Definition:

Data engineering is the discipline of translating raw data into usable forms by building and operationalizing data pipelines across various data and analytics systems meeting use-case requirements, data governance principles and SLAs. Data engineering is a team competency that brings together three different practices — data management, software engineering and I&O management — to achieve frictionless, trustworthy data delivery with agreed-upon business and technical SLAs.

Why This Is Important

Data engineering caters to the full data supply chain, provisioning trusted, quality data to be used at the right business moments with agreed service levels. It enables the creation, operationalization and maintenance of data pipelines across heterogeneous environments, aimed at delivering integrated data as per consumer needs. It also manages data delivery debt around reusability, governance, compliance and operational readiness through data cataloging, data quality and data observability efforts.

Business Impact

- Contextualized data leads to efficient data utilization.

- High-quality data improves data trust among data consumers.
- It becomes faster to onboard new data to existing analytics and data science models with robust data pipeline change management.
- It becomes easier to fulfill regulatory requirements to meet data transparency expectations through cataloging efforts in tracking sensitive data and enabling data governance enforcement.

Drivers

- **Productivity through automation:** Data management teams spend most of their time on data preparation, data integration and operationalization. As a result, these are the primary candidates for automation.
- **Agility:** Organizations are forced to change the way they traditionally work with the waterfall methodology of data pipeline delivery because they are unable to keep up with the demand increase and skills shortage.
- **Self-service and customer experience:** Organizations seek successful consumer experiences, the last mile in the "data insights decisions" continuum. Data engineering enables self-service data management among citizen users and domain experts, and also provides guardrails and established best practices to follow. See Lesson No. 2 in **6 Lessons Data Leaders Can Learn From the Early Adopters of Data Mesh**.
- **Cloud data management:** As organizations struggle to manage multiple data gravities across their on-premises and multicloud setup, the data engineering practice plays a major role in balancing the collect and connect approaches in distributed architectures.
- **Analytics/business intelligence (BI) and data science success:** Organizations are bound to fail if they launch data science initiatives without onboarding the necessary data engineering skills because of high technical debt associated with data management (such as governance, compliance and operations readiness), which must be managed.

Obstacles

- **Inefficient legacy practices:** Baggage around poor integration and operations practices hurts and/or delays data engineering practice adoption.
- **Unrealistic expectations:** Many think a data engineer can "do it all," catering to the full spectrum from data management to software engineering to infrastructure and operations. Data engineering is a team competency involving data architecture, data modeling, data stewardship, software engineering skills (such as DevOps engineering, Python development or test engineering), domain knowledge, incident

management and operationalization expertise. Sometimes, organizations might even need adjacent roles for test engineering and infrastructure automation.

- **Inability to scale:** Data engineering roles are a critical part of data engineering teams. However, adding more data engineers in response to increasing data demands is not sustainable. Organizations need DataOps practices to streamline and scale data engineering delivery.

User Recommendations

- Establish a data engineering discipline with roles that support end-to-end delivery of data pipelines aligned to business use cases, SLAs and governance requirements.
- Catalog an inventory of data assets and make them searchable. Use metadata to drive automation of data pipelines and related artifacts. Study data usage patterns among consumers and systems, and employ this metadata to improve efficiency and optimize delivery.
- Evaluate progress measures — such as time to market, productivity, CI/CD automation of data pipelines, code quality, and cost efficiency of build and operations — quantitatively and regularly, and share them with your stakeholders. As a data engineering leader, add this to your communication plan.
- Foster a collaborative environment by adopting a hub-and-spoke operating model. This model establishes a centralized data engineering team that supports multiple decentralized data engineering teams embedded within various business functions and the data science lab.

Sample Vendors

Ab Initio; Ascend.io; Coalesce Automation; dbt Labs; Informatica; Nexla; Prophecy; Qlik; Upsolver

Gartner Recommended Reading

5 Ways to Enhance Your Data Engineering Practices

6 Lessons Data Leaders Can Learn From the Early Adopters of Data Mesh

Critical Capabilities for Data Integration Tools

Data and Analytics Essentials: DataOps

Appendixes

Hype Cycle Phases, Benefit Ratings and Maturity Levels

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.

Source: Gartner (July 2024)

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 2024)

Table 4: Maturity Levels

Maturity Levels	Status	Products/Vendors
Embryonic	In labs	None
Emerging	Commercialization by vendors Pilots and deployments by industry leaders	First generation High price Much customization
Adolescent	Maturing technology capabilities and process understanding Uptake beyond early adopters	Second generation Less customization
Early mainstream	Proven technology Vendors, technology and adoption rapidly evolving	Third generation More out-of-box methodologies
Mature mainstream	Robust technology Not much evolution in vendors or technology	Several dominant vendors

Legacy	Not appropriate for new developments Cost of migration constrains replacement	Maintenance revenue focus
Obsolete	Rarely used	Used/resale market only

Source: Gartner (July 2024)

Evidence

Gartner Software Engineering Survey for 2024. This survey was conducted to identify the most important roles and skills for software engineering leaders and the change in their demand and importance since last year, understand how talent is sourced generally and for acquiring necessary artificial intelligence (AI)/machine learning (ML) skills, and what tools are seen to increase developer productivity and the metrics used to measure them. It also examines how software engineering leaders anticipate change in their operating budgets and the cost management steps taken. It further aims to identify the quality and testing techniques and programming languages software engineering leaders currently use and/or plan to use; their frequency of usage of UX design, user research and AI in generating components of user experience; and its impact on user satisfaction, accessibility and usability. It also intends to understand the software engineering leaders' responsibilities they find most difficult, the career paths available for senior-level individual contributors and how they are set up, how organizations attract and retain top performers in those career paths, and what management training is offered to staff. The survey was conducted online from October through December 2023 among 300 respondents from the U.S. (n = 241) and U.K. (n = 59). Qualifying organizations operated in multiple industries (excluding the IT software industry and education sector) and reported enterprisewide revenue for fiscal year 2022 of at least $250 million or equivalent, with 63% over $1 billion in revenue. Qualified participants were highly involved in managing software engineering/application development teams and the activities they perform. Disclaimer: 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.