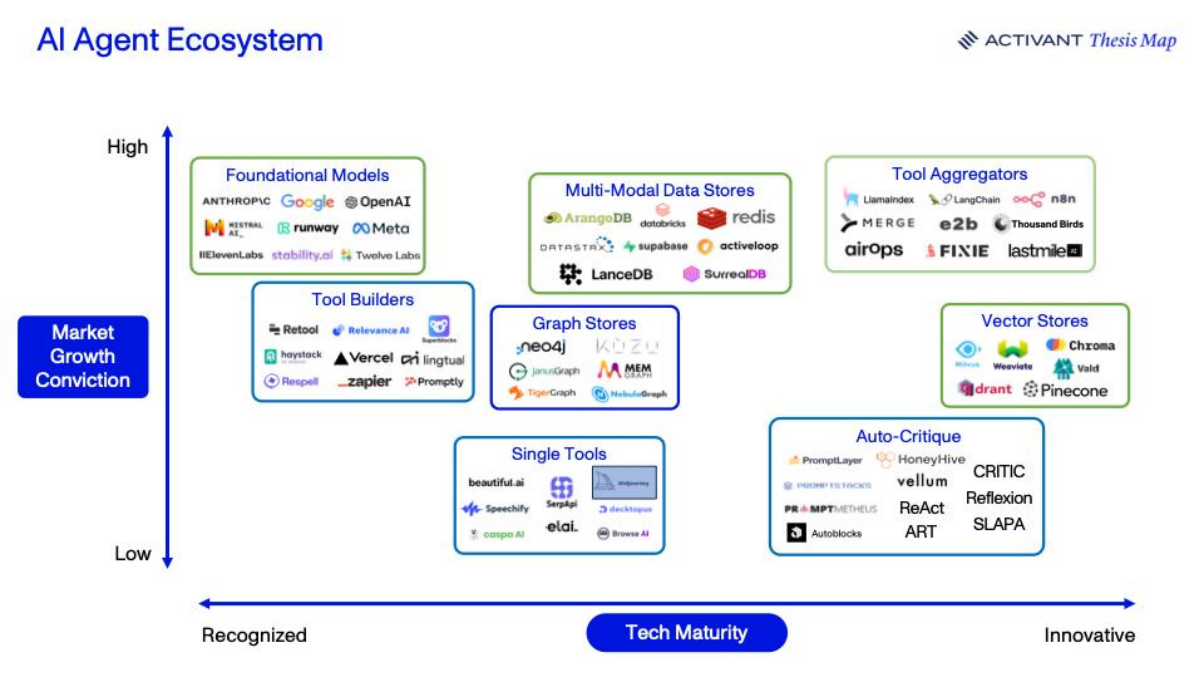
ACTIVANT RESEARCH

Towards the AI Agent Ecosystem

An informational guide for users exploring the space



Updated Q3 2024

Teddy Cohen, Andrew Steele, Marc Wu

We are hardly the first to claim a new technology is poised to upend work as we know it. With every wave of innovation comes a chorus of such pronouncements, many of which turn out to be hyperbole. Technology may make our jobs easier, but most of the time, the core of what we do remains fundamentally the same. But spend five minutes exploring ChatGPT, or any comparable Large Language Model (LLM), and it's clear that this wave is different.

The reason is simple. Most technologies are simply tools that help us accomplish tasks and be more productive. AI, on the other hand, can think – or at least provide the appearance of thinking. It's like having another knowledge worker on the payroll (for a fraction of the cost).

As we explore the world of AI Agents and the potential to shape how we work, we will share three perspectives that we hold at Activant and illustrate what we think is possible through a hypothetical example.

Activant Perspective 1: LLM-enabled software will autonomously accomplish highly complex tasks with increasingly less guidance.

This is the beauty (and threat) of LLMs: they can do work we thought would long be the purview of humans. It's unsurprising, therefore, that many people worry their job will soon be automated away by AI. In the short term, this will lead to dislocation. But technological revolutions are never zero-sum; in fact, they have always been accretive. As workers adapt, they become more productive, and the companies they work for can do more with less. Some stop there, but most businesses opt to invest the gains back into their infrastructure and people. In the end, revenues go up, companies do better, and the economy grows.

That said, we are still in the very early days of this cycle – and the early days of the technology itself.

Activant Perspective 2: LLMs alone are not enough to achieve Perspective 1.

An LLM can't complete most tasks on its own. It is merely a dense set of attention networks: essentially, statistics on a massive scale. Yes, they can engage in human-like "thought," but to generate real value from any of these models, a human would have to integrate them into actual workflows and make use of their decision-making competence.

For example, imagine that you hired workers to lay bricks. You could lead them to a stack of bricks and explain where and how you want them laid. They could perfectly understand your instructions, but understanding alone isn't enough because the workers would also need to be able to lay bricks.

Bricklayers need to have more than just cognition. They need to have arms, legs, and physical strength. Furthermore, they need a memory to recall how you wanted each of the bricks to be arranged, and a way of correcting any errors so they don't cause the eventual structure to fail.

If LLMs are to extend beyond mere Q&A they need (1) a suite of tools to impact the real world, (2) memory repositories to remember what actions they've taken, and (3) auto-critique algorithms to error correct along the way.

When all these components come together, an "Agent" is born.

Technical Definition: An AI Agent is orchestration software that combines an LLM with memory, tools, and auto-critique mechanisms.

Non-Technical Definition: An AI Agent is a highly advanced bot that can complete work that has been assigned to it in natural language.

Though still in their infancy, Agents are poised to fundamentally reshape the way we work. They will become natural extensions of our organizations and complete day-to-day tasks like newly hired employees. They will help us get work done, and even collaborate with others on our behalf. Eventually, they will be able to work together, breaking down tasks into atoms of work able to be handled by a full "office" of Agents, some of which orchestrate, while others specialize.

That is why this wave is different. Working together, Agents will be able to handle increasingly complex tasks, replacing knowledge workers in a variety of fields. As they

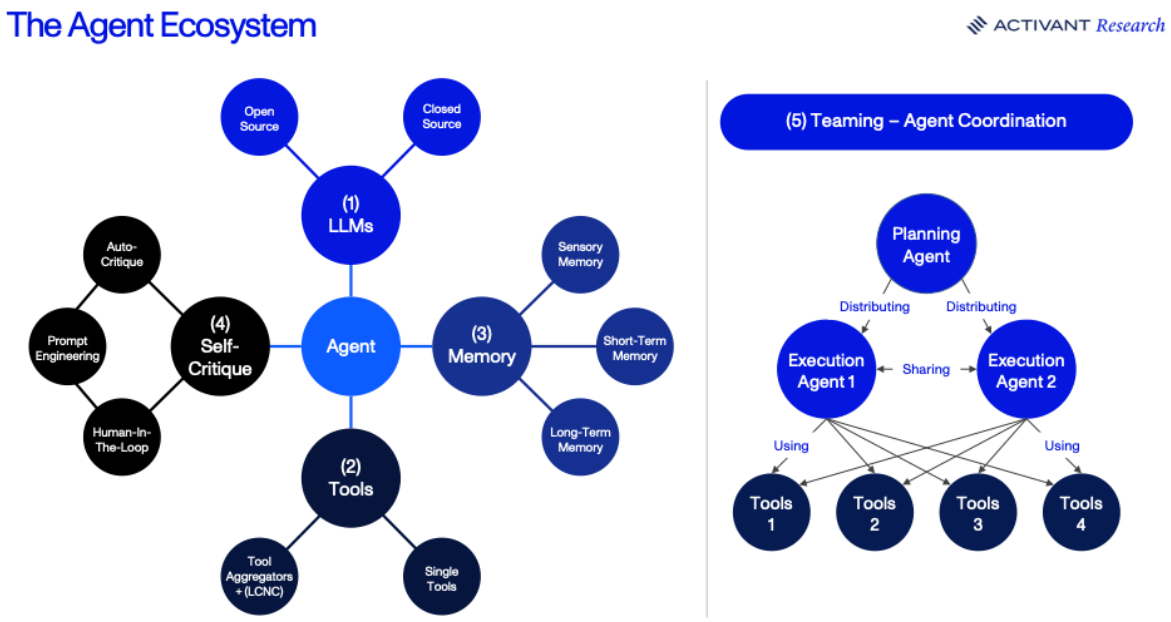
do, what it means to work – and to lead – will change dramatically, as both will increasingly depend on a mix of human and human-like capital.

Activant Perspective 3: Agent teaming is the future of autonomously completed, complex work – and that future is possible today.

Agent ecosystems have four key components:

1. **LLMs:** providing "conscious thought" within AI agents
2. **Tools:** external APIs facilitating task completion and agent interaction with the real world
3. **Memory:** data storage mechanisms that retain and recall information
4. **Auto-Critique:** the ability to correct mistakes when completing tasks

When we add a fifth component – **Coordination** – agents can effectively "team" and autonomously complete complex work. The exhibit below describes the individual components of agent ecosystems, and we will attempt to bring things to life using a hypothetical example. Let's consider the job of an Activant Investment Analyst!



Re-imagining the role of an Activant Investment Analyst

Among many other things, we often rely on our analysts to create market maps, which requires them to:

1. Understand an industry,
2. Find companies that exist in that industry,
3. Compare these companies quantitatively and qualitatively, and
4. Place them on a two-dimensional grid according to this comparison.

Until recently, this was almost inconceivable as a task for machines but let's explore how we might put our "Agent Analyst" to use.

1. Understanding the industry (and task): Large Language Models

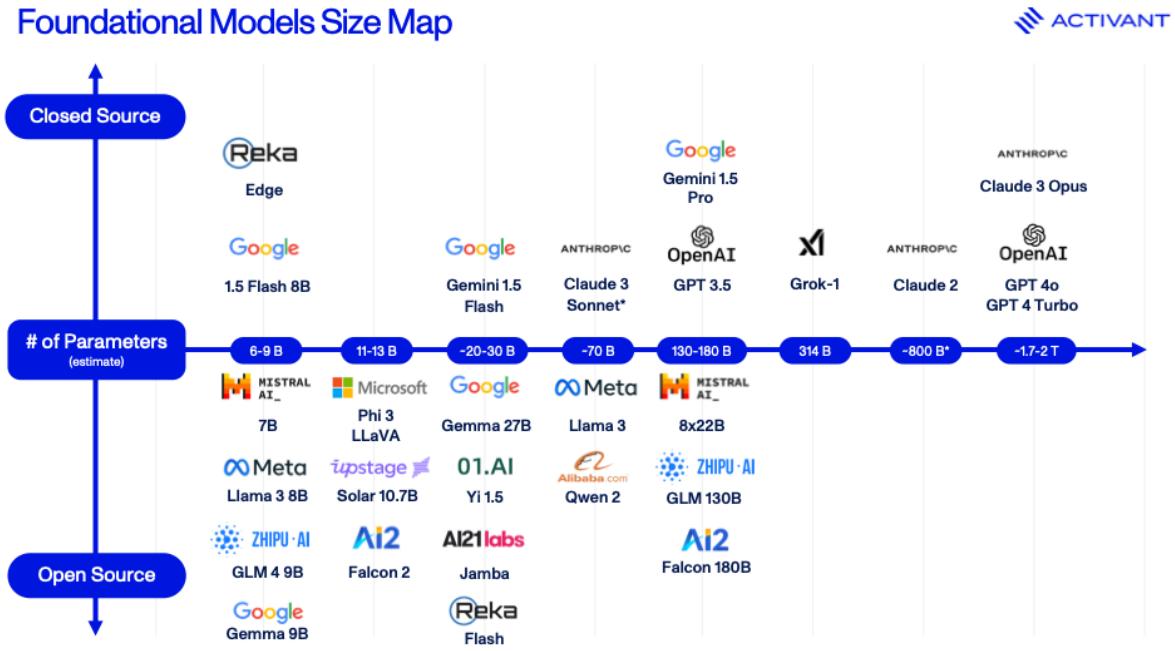
For our Agent Analyst to understand the task at hand, it needs to comprehend instructions in natural language. While LLMs provide that ability, requests literally come in many different shapes and sizes. To compare foundation models and model providers, one would likely consider the number of parameters in a model, whether it is open-source or closed-source, and the provider modality. One might also consider whether it includes "Fine-Tuning-as-a-Service" (FTaaS), the process of training a pre-trained model on subject-matter-specific data to produce a model that performs better in specific use cases.

a) Number of model parameters

When comparing foundation models, the number of parameters in a model is often a good approximation of output quality. As a rule of thumb, the more parameters a foundation model has, the better its performance across different benchmarks such as reasoning, web browsing, or even game playing.[1] Although fine-tuning a model can boost its performance on a particular task, larger models with more parameters tend to have better generalization skills, making them more useful.

As seen in the exhibit on the following page, model sizes are increasing exponentially while model performance is also increasing, but not at the same rate.

[1] AgentBench: Evaluating LLMs as Agents; Liu et al; ICLR 2024 Conference; January 16, 2024



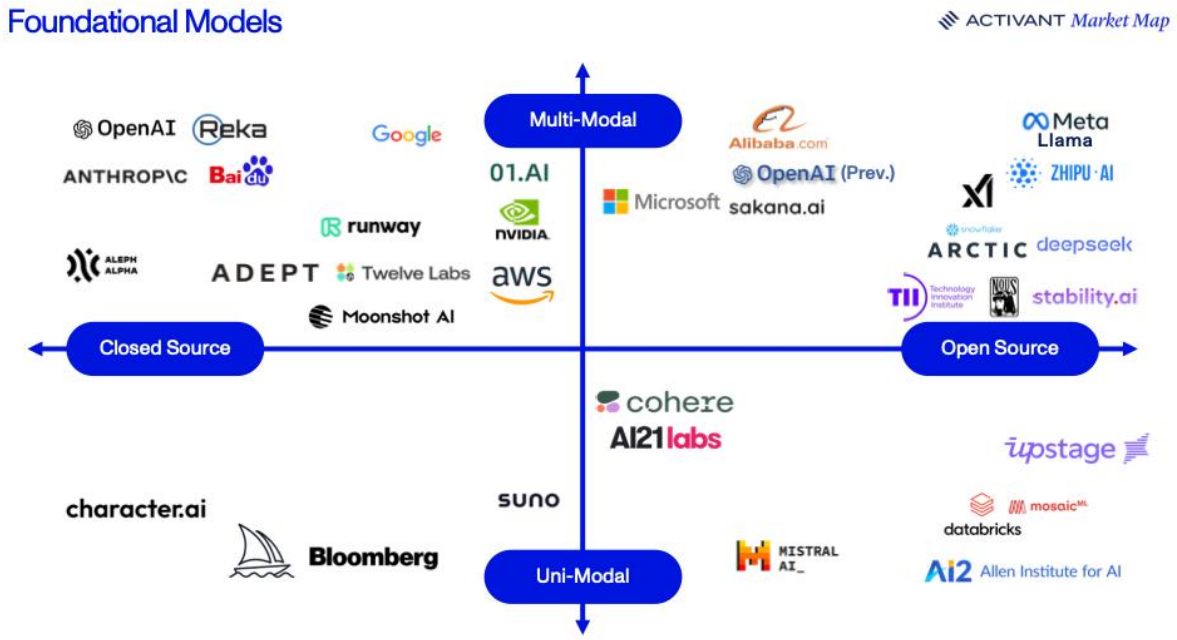
b) *Open vs. Closed Source LLMs*

At the heart of any LLM lies its architecture: a blueprint dictating how various attention heads and layers are organized. There are also "weights," which are numeric values assigned to parameters in the neural network, refined and adjusted through training to enhance performance. Training involves feeding data into the model, allowing it to learn pattern matchings, and then adjusting its weights based on prediction accuracy measured via various benchmarks. It's through this process that the model obtains its ability to understand and generate context dependent and human-like text.

Open-source models, such as Llama 3 by Meta, openly share their architecture, weights, and sometimes even their training methodologies. This approach not only reveals the intricacies of their design but also democratizes access.

Conversely, some entities opt for a more guarded approach. Closed-source models, such as GPT-4 by OpenAI, protect the details of their architecture, weights, or training processes, presenting them as a kind of "black box." Instead of allowing direct model access, they employ APIs as intermediaries, thereby controlling and often monetizing the interaction between the end user and the model.

Since this article's initial publication, the AI landscape has evolved significantly, and we have updated our Activant Market Map of foundational models. Some changes are significant. For example, OpenAI was previously placed on the open-source side, but the company has since focused on closed-source offerings and has deprecated many of its former open-source models.



c) *Provider Modality*

Provider modality relates to the data type a provider offers models for. This includes various transformations such as text-to-video, text-to-text, text-to-image, and image-to-text, among others. Companies that operate in just one domain are labelled as "uni-modal"; by contrast, "multi-modal" companies provide models spanning multiple domains. As AI Agents continue to evolve, we expect a growing emphasis on modality in foundation models (more than just LLMs).

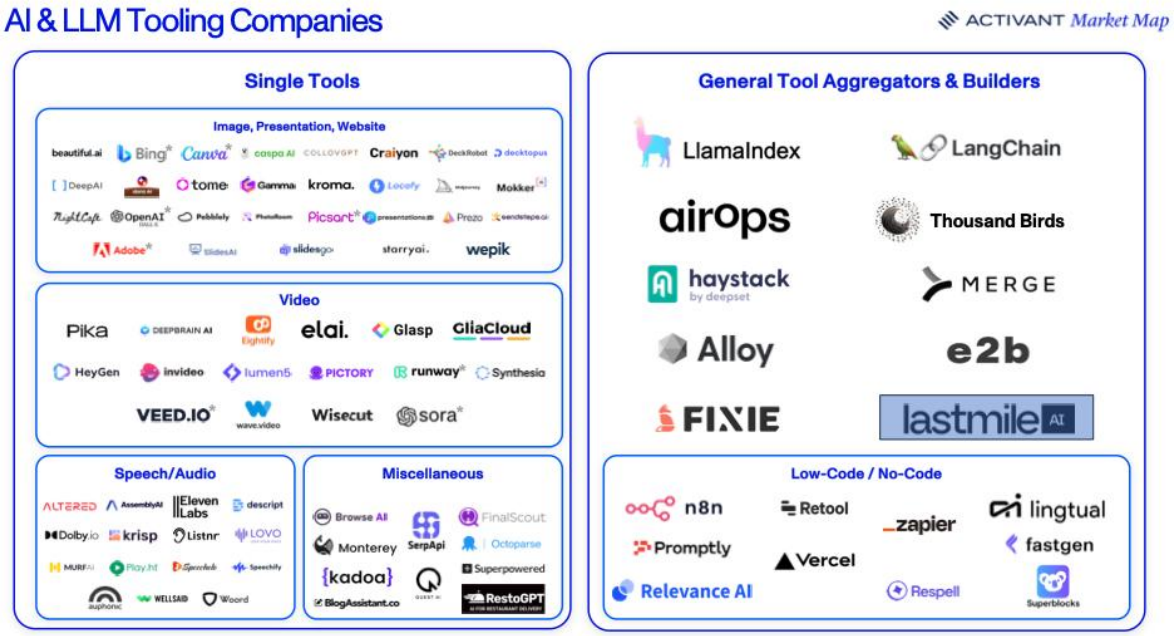
Today, OpenAI, Google, and Anthropic dominate the industry, but the next few years will see considerable competition as investment in R&D increases in a proverbial arms race. Given this uncertainty – and the promise of better models always on the horizon – it's hard to imagine agents relying on a single provider. From an investment perspective, backing one LLM provider is essentially a vote of confidence in its R&D team, but past performance is not necessarily indicative of future results.

2. Finding Companies: Tools

Having understood the industry and the task, the Agent Analyst now needs the ability to search for companies. In other words, the Agent Analyst needs a tool.

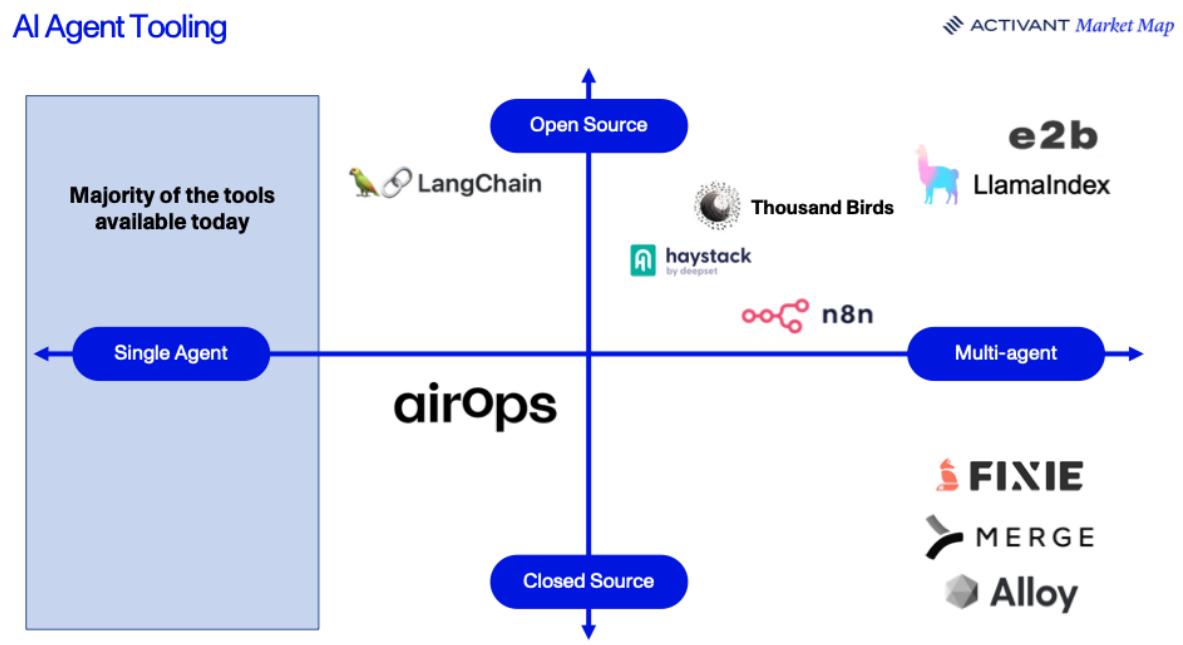
Simply put, tools are snippets of code that LLMs can use to create, modify, and utilize external resources to execute tasks they couldn't otherwise complete on their own.

Most tools are problem specific. In this case, all the Agent Analyst needs is a connection to Google or SerpAPI, a prominent wrapper for Google's Search API. Unsurprisingly, other tasks will require different tools. Game this out, and you quickly arrive at the need for toolkits and tool aggregators. As tasks become more complex, agents will rely on not one but a variety of tools to accomplish a single objective. Through tool aggregators, Agents will be able to access the resources they need programmatically and as those needs arise.



a) Single Tools

Single tools refer to companies or services that specialize in just one function, such as searching the internet, booking a flight, or creating audio with a realistic voice. While these offerings do have certain strengths, integrating them into a toolbox, and thus into agents, is often cumbersome. Agent libraries provide the orchestration framework that combines all the components in the Agent ecosystem, and in some prominent examples toolboxes need to be set up before an Agent itself is even initialized. This means that if there's a need to integrate additional tools later, the agent would have to be initialized all over again. Given these considerations, embedding single tools into an Agent's workflow can be quite labor-intensive and time-consuming.



b) Tool Aggregators

By contrast, tool aggregators offer Toolboxes-as-a-Service (TaaS). Theoretically, aggregators can compile an infinite number of tools and distribute them via a single, unified API. In practice, they typically provide tools that have been identified by an Agent as necessary to complete a given task. If, however, an Agent requires additional tools, it can access them without restarting.

c) Low-Code / No-Code (LCNC)

A subset of tool aggregators, LCNC companies could provision tools and/or provide tool-building functionality for agents.[2] As long as they offer an API, agents can leverage these solutions as they do tool aggregators, integrating (and even custom-building) tools into their workflows on the fly.

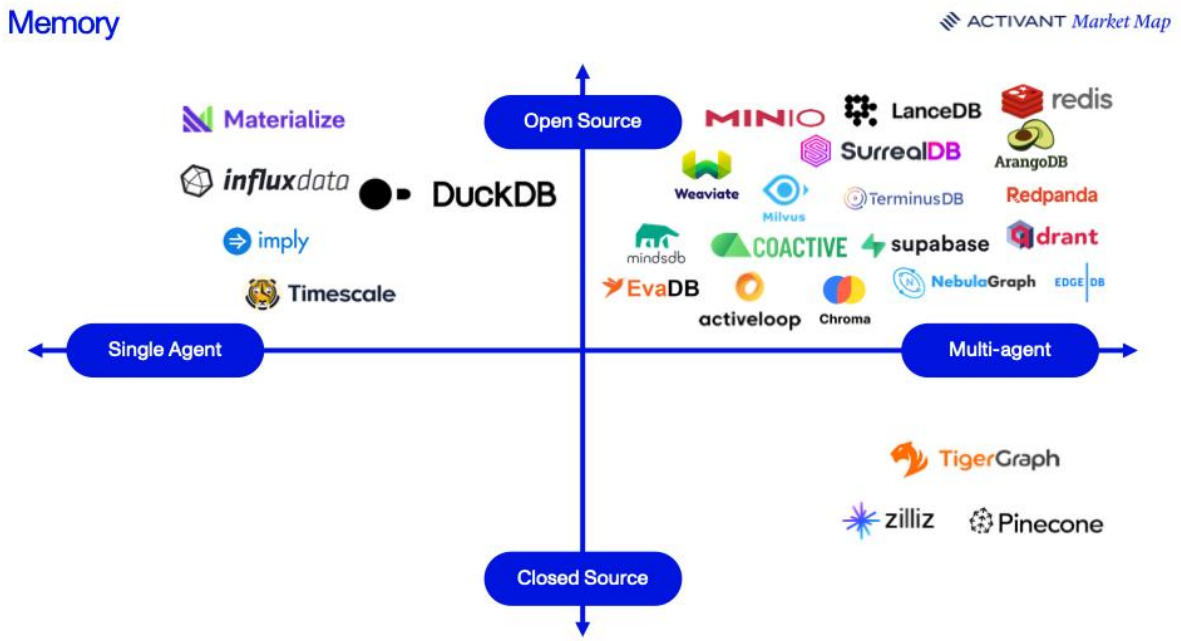
Many LCNC platforms have existed prior to the advent of LLMs and already have large libraries of powerful, ready-built tools to draw upon. Several companies, including Relevance AI and Superblocks, have begun offering Agents-as-a-Service (AaaS), using these libraries to jump ahead of competitors, many of whom are starting from scratch. Others may choose to offer APIs to facilitate quick adoption within the agent ecosystem and more reliably maintain their market share.

[2] Why using internal tools to do more with less matters today: No-Code and Low-Code; Activant Research; March 2023

Because of their ease of integration and ease of use, we believe tool aggregators and API-based LCNC platforms are the best positioned to take advantage of this new agent ecosystem. As platforms like LlamaIndex and Relevance AI grow, Agents will be able to dynamically access a vast library of toolsets without needing to update their structure or code, increasing both their functionality and value.

3. Comparing Companies: Memory

As our Agent Analyst has progressed, it has gained the capability to discern tasks and search for companies. However, identifying a company means nothing if an Agent can't remember it for later use. Agents require a dedicated memory component that can store company names and their associated descriptions.



Over the years, the software industry has seen a multitude of data-structure innovations, from traditional SQL-based relational tables to modern streaming services like Kafka. Yet, to truly grasp the intricacies of Agent development and the depth of virtual cognition, it's helpful to draw parallels with human memory.

At a high level, human memory is split into three core categories: sensory memory, short-term (or working) memory, and long-term memory.

a) Sensory Memory

In humans, sensory memory can be thought of as the direct inputs from our various senses of taste, smell, sight, touch, and hearing. These are perceived subconsciously, processed, and then filtered for relevance.

Similarly, in the realm of AI Agents, sensory memory captures the responses generated from the tools AI Agents utilize**.** Human neural frameworks efficiently filter out redundant and useless sensory data, elevating only important information for conscious processing. The sensory memory of an Agent operates on similar principles, sifting through vast data while retaining only what's significant.

Over time, we believe that sensory memory will include the ingestion of many different data types, from unstructured data files (such as DOCX, PDF, EPUB, or recordings) to semi-structured datasets (such as messages and CRM data).

b) Short-Term Memory

Having processed inputs on the subconscious level, Agents are left with the inputs that require conscious thought. In humans, conscious thoughts reside in working or short-term memory. When we actively think about a topic, we do not consciously remember the context for the thought, but it is nevertheless recalled. In fact, our short-term memory is believed to have the capacity of about seven items[5] and lasts for 20-30 seconds. For Agents, using the context windows defined by LLMs to store short-term memory is the default solution. As context windows increase in length and new techniques for prompt engineering solve existing constraints, the context window will be an even more effective location for short-term memory.

In the context of our Agent Analyst, short-term memory can be thought of as remembering the larger task at hand (i.e., creating a market map) when deciding what smaller task (i.e., finding a company) needs to be accomplished next. In practice, this information will largely live inside the context window of the LLM API calls. While it's essential for the Agent to complete a task, we believe that there is little opportunity for investment in short-term memory due to the limited additional infrastructure required to build it.

c) Long-Term Memory

Once subconscious and conscious memory is in place, an Agent Analyst will need some way of saving information for longer periods. This is where long-term memory comes into play.

In humans, long-term memory is finite due to the limited capacity of the brain. As a result, we have developed two primary mechanisms to retain knowledge: repetition and adrenaline. Agents lack this capacity constraint. Instead, they are limited by the amount of memory that they could conceivably access. For example, if our Agent Analyst was integrated into our drive and needed to search for the PowerPoint example of our Market Map, the size of the context window would limit the number of folders that it could select from in its search.

In our view, connections to long-term memory repositories are no different than tools, in that they are blocks of code that can be executed and return a response.

Over the next decade, we believe that the entire ecosystem of data management will grow even faster than it has historically because of increased Agent data utilization. Agents will be able to build pipelines from company data, leverage this data in their own processes, and integrate outside data into existing workflows. As the Agent ecosystem grows, the demand for data stores will increase even more rapidly, making the volumes of unstructured company data – which today sits untouched – significantly more valuable.

We've listed a handful of data structures and the impact we anticipate from Agents below:

Data Structures and AI Use Cases

ACTIVANT *Research*

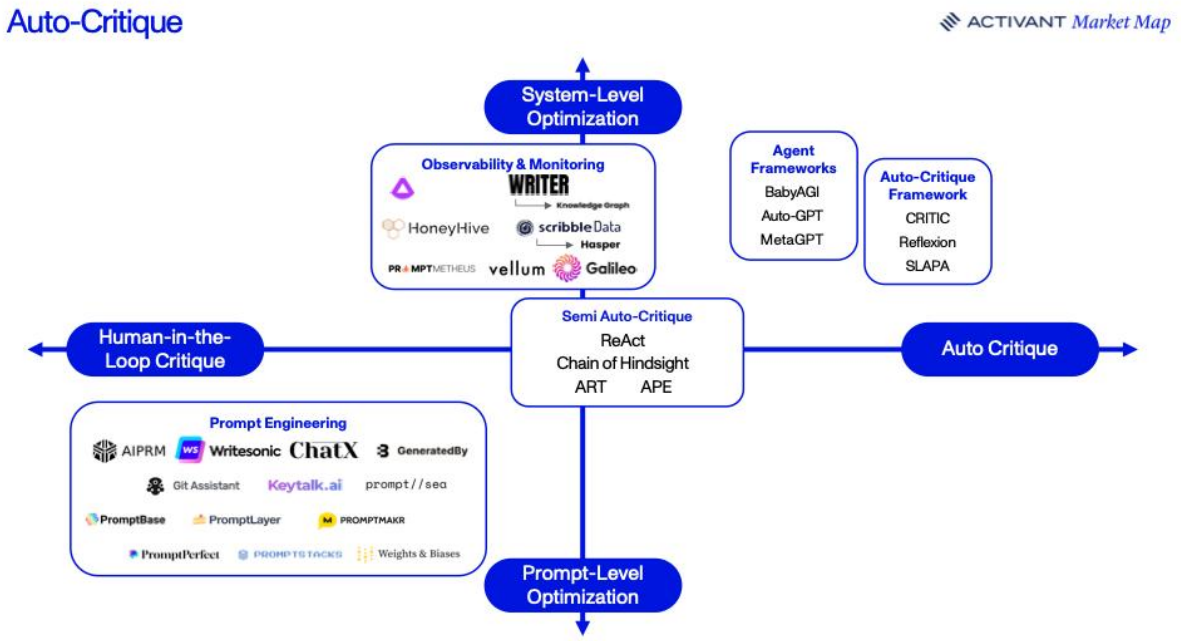
	Relational Database	Graph Database	Vector Database	Multi-Model Database
Speed	●	◑	◔	◑
Complex Relationships	◑	●	◔	◕
Semantic Meaning	○	○	●	◕
Multi-modal	◔	◔	●	◕
Corporate Adoption	●	◔	◔	◑
AI Agent Retrieval Methods	Text-to-SQL	Text-to-Cypher, Text-to-Query	Vector Embeddings Similarities	All mentioned
AI Use Cases	Model Training, Business Intelligence, Numerical Analytics	Knowledge graph, Recommendation system, Fraud detection	Semantic Search, Unstructured Search, Classification	All mentioned

4. Placing Company Logos: Auto-Critique[3]

Our Agent Analyst has successfully understood the industry, used a search tool to find all the companies within it, and saved them in a database for future output. As a final step, we need the Agent to place the logos of each of those companies onto an Activant Market Map, fully understanding the significance of each company's position.

Plotting logos, of course, is the easy part of the challenge; understanding where to place a logo in a subjective two-dimensional plane is the truly complex task. It requires iterations, constant comparisons to other companies, and perhaps even outside input from the Activant team. Therefore, the Agent needs a mechanism for critiquing itself. We call this auto-critique.

While we foresee potential challenges like defensibility, this remains a vital component of the ecosystem with several approaches underway: a) prompt engineering, b) human-in-the-loop, and c) auto-critique.



a) Prompt Engineering

Prompt engineering is the process of forming and refining natural language statements and questions to achieve an optimal outcome from an LLM.

[3] Activant is continuing to develop its thesis on AI Agents. Our view on auto-critique has evolved and will be elaborated on in an upcoming research article.

In practice, LLMs are only capable of comprehending and responding to inputs that fit within a defined context window. For example, GPT-4's context window is about eight thousand tokens, meaning only about seven thousand words can be provided as context at any one time.[4]

In our view, prompt engineering is likely a temporary solution, and one we expect the industry to shift away from as LLMs improve in terms of task comprehension and context window size (though the timeline for this shift remains uncertain). Meanwhile, existing businesses in this domain have a chance to pivot their business models, leveraging their current distribution to tap into emerging opportunities.

b) Human-In-The-Loop

Human-in-the-loop refers to the use of humans to validate that Agents are accurately processing and reasoning through tasks. It is just what it sounds like: leveraging humans to correct or modify the reasoning of LLMs. Currently, developers use this method when building Agents, as they want visibility into their inner workings. Because it requires that people scale linearly with computing resources, it will likely be unsustainable in the long term.

c) Auto-Critique

Auto-critique refers to the autonomous use of LLMs to check and modify Agent reasoning about a task. This is an autonomous process that allows Agents to improve and amend their own output by reflecting on past responses and errors, all without human intervention. To do this, Agents draw on new input from a variety of sources, including external tools, other Agents, affiliated LLMs, and heuristic functions. Once they arrive at a satisfactory result (governed by predefined metrics for evaluating self-reasoning), a stop condition is granted, allowing Agents to finalize their output.

[4] Note that here we are approximating token count to word count, which is not strictly accurate but expedient for the example. Additionally, we are leaving roughly 1 thousand tokens available for the response.

5. Teaming: The Rise of the Planet of the Multi-Agent Systems

So far, we've focused largely on the mechanisms and potential of standalone Agents. However, to fully appreciate the potential of this nascent sector, we must also look at the collective strength of Agent teaming, otherwise known as a multi-Agent system (MASs). An MAS is a network of intelligent Agents that interact with each other, forming a machine-to-machine (M2M) communication network. Within the framework of an MAS, the power of individual Agents is not merely aggregated, it is amplified, thereby boosting the overall effectiveness of the system in a multiplicative and potentially exponential manner.

Returning once again to our Agent Analyst example, we must remember that building market maps is but one of many tasks of an Activant analyst. In fact, a market map is only a small piece of the larger research and investment process. Analysts must also deal with financials, retention, defensibility, and much more. In their current state, no single Agent could write a complete investment memo, though individual Agents could accomplish each component. This is the promise of an MAS: it's able to accomplish significantly more complex tasks than a single Agent ever could.

Early attempts[5] at the concept, such as Agent-based systems (ABS), were inherently limited. In an ABS, though multiple Agents work on the same task, they can't collaborate,

[5] Around March 2023

and they're forced to work within narrow, predefined constraints. Rather than acting autonomously, Agents are confined to rudimentary roles with very strict and cyclical operating patterns. Think of BabyAGI's cycle of operation: Agents follow relatively one-dimensional orders, are only sequentially prompted, and report back to their "commanding" Agent when a given task is complete. Given the limited scope and flexibility, such systems have been less than transformative.

By contrast, an MAS encourages direct communication between two or more Agents with minimal rules governing how they operate. This supercharges the system, leveraging Agents' collective cognition and problem-solving capabilities to resolve complex problems quickly, reliably, and effectively.

As systems shift away from rigid and sequential orders to more dynamic, human-like interaction, their ability to tackle such problems will only increase.

The Value of Multi-Agent Systems

Quantifying the value-add of an MAS is a considerable challenge. The nuances of its applications – and the myriad ways it can be designed and built – mean the perceived benefits can vary widely. That said, there are distinct, universally applicable advantages that are evident now, despite the infancy of the space.

Distributed Intelligence

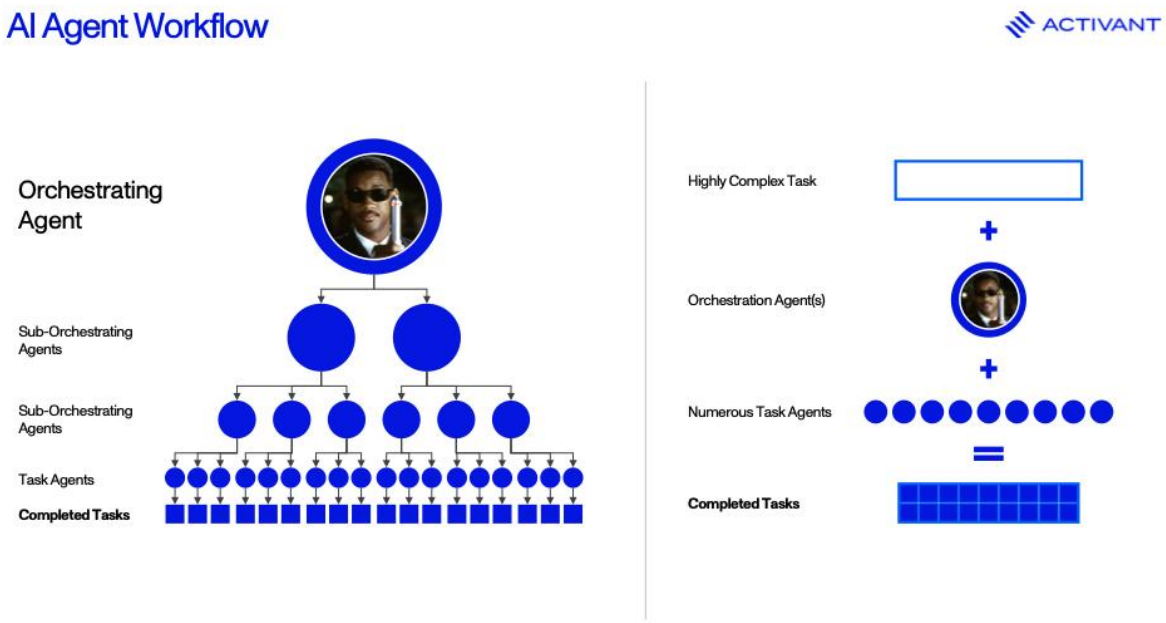
MASs address a key shortcoming of ABSs, wherein the executing Agent in an ABS might become overwhelmed and falter when confronted with a more complex task. In an ABS, the first executing Agent is responsible for the detailed planning and implementation of extensive, high-level goals given by the "planning" Agent – a load that can cause the system to buckle. An MAS instead breaks down tasks across multiple Agents recursively, consequently making them far more resilient.

In theory, you could envision an Agent structure that decomposes workflows and tasks into the smallest unit possible: a "task quantum." A multitude of Agents could then be tasked with handling a singular piece of the overall problem. By divvying up tasks in this way, it's possible for the system to tackle incredibly complex challenges, simply by having each Agent follow a clear, simple set of instructions.

Imagine an architect who designs a grand monument. While the architect can create the blueprint, they almost certainly lack the skills and the time to physically construct their vision. Instead, they delegate smaller tasks to teams of workers, who are managed by others experienced in construction tasks. Just as the architect relies on numerous

teams to execute their blueprint, an MAS distributes tasks to many Agents to achieve complex goals.

Each Agent, the equivalent of a bricklayer in our first analogy, might not be able to plan and execute the entire task, but can perform its piece of the task – its task quantum – efficiently and effectively. This accumulation of simple tasks, handled by numerous Agents, ultimately accomplishes the overarching objective.



Scalability

As the complexity of a problem escalates, more individual Agents can be integrated into the system, rather than adding whole new Agent-based systems. Scalability also relates to the distributed nature of MASs, where tasks and decision-making processes are spread across multiple Agents. This distribution allows the system to handle larger and more complex tasks without necessarily requiring a proportional increase in computational resources.

Parallelism

The power of MASs isn't just in distributing tasks. They also benefit from enabling Agents to communicate directly, allowing them to optimize their own workflows.

For example, as Agents work together on tasks, they share information on their successes or failures, creating a "knowledge network" that helps other Agents refine their efforts. If one Agent encounters a setback, it alerts the others, enabling the collective to bypass strategies that have already proven ineffective and preventing

duplicate failures.

Conversely, if an Agent finds a successful approach, it shares the insight with the rest, allowing them to focus on and expand this productive pathway. In essence, the Agents are continuously learning from each other's experiences, coalescing around promising or unexplored avenues of action. Through this exchange, groups of Agents can more efficiently navigate tasks, consequently minimizing failures and maximizing productivity.

Decentralization

Decentralization is one of the key advantages of MASs. While Agents in an ABS may be bound by centralized rules and workflows, MASs, by contrast, ensure Agents can survive independently from one another. Unlike in an ABS – where a single breakdown can cascade into a system-wide disruption – if an Agent fails in a MAS, the network will remain intact, simply reassigning that Agent's tasks.

Notable Companies

We have identified several notable companies building crucial components of AI Agents. Our Activant Thesis Map lays out some of the key segments in the market and our belief in their market growth potential. It goes without saying that the AI market overall is incredibly exciting, and growth expectations are very high across the board.

1. **LanceDB** provides open-source embedded multi-modal databases for AI companies. The company's databases enable efficient vector search, cost-effective indexing, and data filtration and formatting for LLM training, among other advantages. LanceDB works with companies like Airtable, Harvey, and Imagine Art's Midjourney to provide secure and scalable database infrastructure. Agents can use LanceDB to ensure proper database management and efficient performance on complex tasks.
2. **Supabase** is an open-source Firebase alternative that provides full PostgreSQL databases, authentication services, edge function capabilities, data storage, real-time data synchronization, and vector database integrations and embeddings management. The company generally simplifies the process of setting up and managing a backend infrastructure. Users can use Supabase with pgvector, a PostgreSQL extension that enables high-dimensional vector embedding indexing and similarity search in PostgreSQL for increased efficiency. Agents can leverage Supabase to simplify backend setup and reduce their coding needs and can also employ pgvector for efficient vector embedding and search.
3. **Qdrant** offers an open-source vector database and similarity search engine designed for applications requiring high-dimensional similarity search. The company can efficiently index and retrieve vector embeddings, positioning its

offerings for such applications as recommendation systems and anomaly detection.

4. **Twelve Labs** creates video language models that combine proprietary video encoder models with LLMs. The company's combined models can perform cross-modal reasoning quickly and accurately. TwelveLabs provides the capability to search for clips from large corpora of videos, generate text about videos, classify in-video content, and index videos. Agents can use the TwelveLabs API to perform dense video content analysis and classification, boosting their multimodal effectiveness. **n8n** is an open-source, low-code/no-code automation tool that allows users to integrate 400+ services and automate complex workflows. n8n's product is used by such companies as Splunk, Autodesk, and Adobe. Agents can use n8n to automate departmental workflows in DevOps and SecOps, giving them access to applicable data and improving task execution efficiency.
5. **LangChain** and **LlamaIndex** are open-source AI tool aggregators/AI tool builders designed to streamline the development of LLM-powered applications. Both companies allow for the creation of complex workflows by integrating different models, tool APIs, and data sources. LangChain and LlamaIndex are especially crucial for Agents, aggregating tools that can be used for any given Agent workflow.

Final Thoughts

As Agents take over these and many other aspects of our work, we see our roles transforming in exciting ways. Instead of constructing market maps, we'll evaluate them. Instead of building models, we'll test their assumptions. Instead of writing memos, we'll discuss their implications. In our work, and in most fields, experience earned over years of apprenticeship won't be replaced, and human-to-human interaction will continue to be the most important driver of value.

At Activant, we are passionate about the Agent ecosystem, and this work is only a subset of our research to date. For each of the components we discussed, there are additional considerations and frameworks when applying them to MASs – everything from the use of tools across multiple Agents at once to concurrency issues in shared working memory. Additionally, in developing enterprise-ready applications, a few more components are required, such as Agent security, compliance management, access provisioning, and Agent profile management. In addition, we've begun researching how Agents will be built in verticals like fintech, cybersecurity, commerce, and logistics.

If you're building in this space, thinking about how to approach using Agents in your organization, or just looking to ideate – **let's talk.**