

What Are MCP Servers? The New AI Trend Explained for Everyone



Sebastian Buzdugan

Follow

12 min read · Mar 8, 2025



173



Artificial intelligence is taking a big leap beyond just answering questions — it's now reaching out into the world of our data and tools. If you've heard buzz about **MCP servers** lately, you're not alone. MCP servers are being hailed as the next big thing in AI integration. But what exactly are they, and why are AI enthusiasts so excited about them? In this article, we'll break down this new trend in simple terms, explore real-world examples, compare it to traditional AI setups, and show you how to get started. By the end, you'll understand what MCP servers are, why they matter, and how they're changing the AI landscape.

Understanding MCP Servers: A General Overview

MCP stands for **Model Context Protocol**, an open standard recently introduced to bridge AI models with the outside world. At its core, MCP defines a way for AI systems (like large language models) to connect and communicate with external data sources and services. An **MCP server** is one side of that connection — essentially a service or connector that provides an

AI model with access to some resource, tool, or data in a standardized way. On the other side is an **MCP host**, which is typically the AI application or assistant (for example, Anthropic's Claude desktop app) that uses these servers. The MCP host acts as the "AI brain," and MCP servers act like its extended senses and hands, fetching information or performing actions on behalf of the AI.

In simpler terms, you can think of an MCP server as a **plugin or adapter for AI**. Just as your web browser can have plugins to add functionality, an AI model can use MCP servers to safely extend its capabilities — whether it's looking up a document, searching a database, or even controlling a web browser. What makes MCP servers special is that they all speak the same **standard protocol**. This means an AI doesn't need custom code for each new tool or data source; instead, it communicates with any MCP server using a common language. *Some have even likened MCP to being the "USB-C of AI" — a universal connector for AI models and external systems.*

MCP servers matter because they address a key limitation of today's AI assistants: **isolation from real-world data**. Even the most advanced AI models are often "trapped" in the sense that they only know what's in their training data or the prompt you give them. Every time you wanted an AI to access a new database, online service, or live information source, you historically needed to build a custom integration. This was time-consuming and hard to scale. MCP changes that. **By providing a universal, secure way for AI to connect to various systems, MCP servers let AI models fetch up-to-date, relevant information and even take actions.** An AI assistant using MCP can, for example, retrieve the latest entries from a knowledge base, check your calendar, or send an email — all through different MCP servers — without bespoke coding for each case. In short, MCP servers make AI **more**

context-aware and useful by plugging it into the tools and data we use every day.

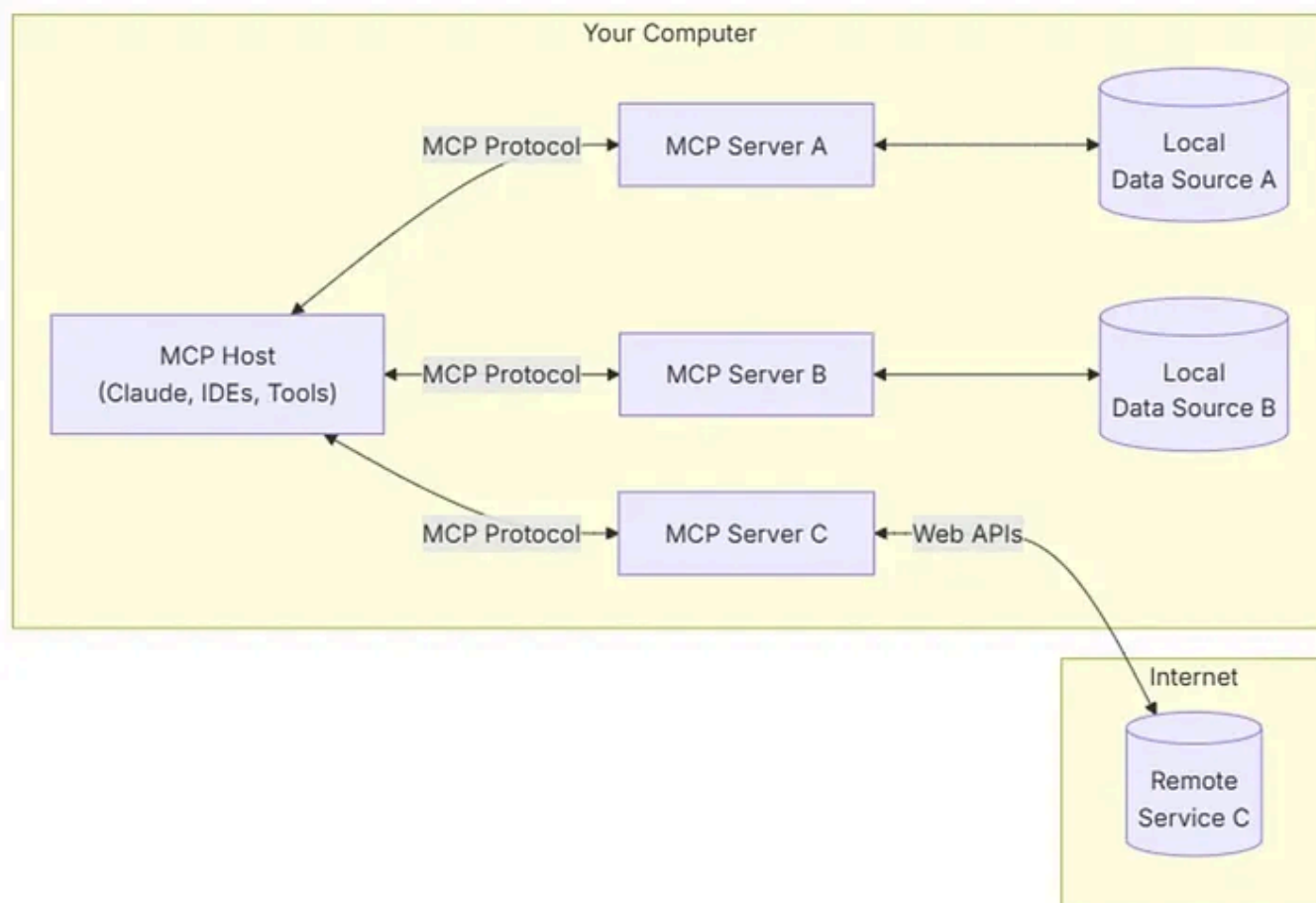


Figure: A conceptual diagram of the MCP architecture. Here, a single AI **MCP Host** (the AI assistant) is connected to multiple **MCP Servers** that each provide different capabilities — such as fetching files, querying databases, or calling APIs. This standardized client-server approach allows the AI to leverage many external resources at once

Because MCP is an open protocol, anyone can build an MCP server for a particular tool or data source. These servers advertise their capabilities (for example, “I can search documents” or “I can fetch weather data”), and any AI **MCP client** can discover and use them. Communication is two-way and secure: the AI can send requests (like asking for a file or invoking an API) and get results back, and it can even trigger actions through the server, all with appropriate user permissions. Importantly, **security is built-in** — MCP servers keep sensitive credentials (API keys, database logins) on their side, so

the AI never sees them directly. Plus, the user typically must approve actions, preventing the AI from going rogue with your data. All these design choices make MCP servers a powerful yet safe way to enrich AI models with real-world context and abilities.

Real-World Examples of MCP Servers in Action

MCP servers might sound abstract, but they're already making waves in both AI research and industry. Developers and organizations are rapidly building connectors for all sorts of services. Here are some **real-world examples** that

[Open in app](#) ↗

Medium

 Search

 Write



connect AI assistants with their internal tools and data silos. For instance, Anthropic has released pre-built MCP servers for popular business apps like **Google Drive for document storage** and **Slack for team chats**. This means an AI assistant could retrieve a file from your Drive or summarize a Slack thread on demand. There's also an **Atlassian MCP server** that integrates with Confluence (for documentation) and Jira (for issue tracking), allowing AI to pull up project docs or ticket information easily. All of this helps organizations get AI help that's specific to their own knowledge and workflows, rather than generic answers.

- **Software Development and DevOps:** MCP servers are transforming how AI co-pilots assist in coding and engineering tasks. Early adopters like the developer tools companies Zed, Replit, Codeium, and Sourcegraph have been working with MCP. For example, using a **GitHub MCP server**, an AI can fetch specific code files from a repository (without needing full repository access) and even help manage version control tasks like creating branches or drafting commit messages. Imagine asking your AI assistant to “find the function where we handle login” and it retrieves the

exact file from GitHub, or telling it “create a new branch for feature X” and it does so. This is now possible through standardized MCP calls. Similarly, a **Git MCP server** can interact with Git repositories directly, and tools like **Puppeteer MCP server** let an AI perform web browser actions (clicking buttons, filling forms) for testing or data scraping tasks. All these make AI a more handy helper in the software development pipeline.

- **Knowledge and Research:** In the realm of research, MCP servers are helping AI tap into large knowledge bases. A great example is the **ArXiv MCP server**, which allows an AI model to search and retrieve academic research papers from the arXiv database. An AI researcher could ask their AI assistant, “find the latest papers on quantum computing from arXiv,” and the MCP server would handle querying arXiv’s API and return the results. Likewise, there are MCP connectors for web search and news: for instance, **Brave Search** and **Google News** MCP servers enable AI to perform web searches or look up news articles in real time. This is incredibly useful for keeping AI-generated answers up-to-date with current information. Instead of being stuck with year-old training data, an AI with these servers can fetch up-to-the-minute facts or research data when needed.
- **Data Analytics and Monitoring:** Many teams are integrating AI with their analytics and monitoring dashboards. MCP servers exist for services like **Postgres databases**, letting an AI run queries on live data, or for error tracking tools like **Sentry** or **Raygun**, enabling an AI to fetch error reports and user analytics. For example, with a Raygun MCP server, an AI assistant could automatically pull the latest application error logs or performance stats and help a developer analyze them. This kind of integration turns AI into a smart analyst that can dive into operational data on command.

These examples are just the tip of the iceberg. The MCP ecosystem is growing every week as developers open-source new servers for different platforms. The beauty is that **any new MCP server instantly becomes a tool that any MCP-enabled AI can use** — a rising tide that lifts all boats. This collaborative growth is why many believe MCP servers are not just a fleeting trend but a foundational shift in how we build AI-powered applications.

MCP Servers vs. Traditional AI Servers: Key Differences

With the emergence of MCP, you might wonder how this differs from “traditional” AI setups or servers. Let’s clarify the **key differences and why MCP servers are gaining popularity** over older approaches:

- **Standardized Protocol vs. Custom Integration:** In the past, if you wanted an AI model to access a new service (say your calendar or a specific database), you often had to write custom code or use a specialized plugin unique to that service. Every integration was a one-off project. **MCP servers replace those fragmented connectors with a single, universal protocol.** In other words, once an AI platform supports MCP, it can talk to any MCP server in the same standardized way. This is a game-changer for interoperability. Developers can now build against one standard and have confidence that their AI can connect with numerous tools without extra hassle. It’s akin to moving from a world of proprietary chargers to a standard like USB-C — much simpler and more compatible for everyone.
- **Rich Context and Live Data vs. Isolation:** Traditional AI systems often operated in a silo, relying only on their training data and maybe some hard-coded knowledge bases. If they needed current information, there was no straightforward, safe way to get it on the fly. MCP servers, however, give AI models a live link to the outside world. This **enhanced context awareness** means an AI can pull in fresh, relevant data whenever

necessary. For example, an older AI assistant might only give general advice on travel because it lacked real-time info, whereas an MCP-enabled assistant could actually check flight prices via an API and then give you a detailed answer. Users get more accurate and up-to-date responses because the AI isn't stuck with stale information.

- **Two-Way Interaction and Actions:** Another major difference is that MCP is designed for two-way communication. Traditionally, an AI might retrieve info from somewhere (one-way) but not perform actions, or doing so required complex workarounds. With MCP, an AI can not only ask for data but also **initiate actions through the server**. This could be anything from adding an event to your calendar, to posting a message on Slack, to executing a script. Of course, these actions are all mediated by the MCP server with security in mind. The key point is that AI moves from being just an information provider to an **agent that can act** on your behalf in external systems (with your permission). This opens up new possibilities for automation and assistance that weren't easily achievable with traditional AI setups.
- **Security and Control:** One might worry that giving AI access to tools could be risky. Traditional integrations sometimes required sharing API keys with the AI or hard-coding credentials, which wasn't ideal. MCP servers were built with security from the ground up. **Sensitive credentials stay on the server side**, and the AI host only requests what it needs in a controlled fashion. Users typically have to approve any significant action. Because MCP servers run locally or in a trusted environment (at least in the current design), they are not open endpoints for attackers to hit. This local-first, user-in-the-loop approach means you get the benefits of connectivity without blindly trusting an AI with the keys to your kingdom. The result is a more secure integration compared to many ad-hoc solutions.

- **Ease of Development and Scaling:** Finally, from a developer's perspective, MCP greatly **simplifies the development** of AI-augmented applications. Instead of reinventing the wheel for each new data source, developers can use existing MCP servers or quickly build one following the standard. It's faster to prototype new AI capabilities because you can mix and match available servers (for files, emails, APIs, etc.). This also means scaling up to support more integrations is less painful — you're not maintaining a tangle of custom code, but rather plugging into a growing ecosystem. As the MCP community contributes more connectors, **AI systems can scale their scope by simply adding more servers**, not by complex re-engineering.

In summary, MCP servers offer a **more modular, extensible, and connected approach** to AI compared to traditional isolated AI servers. They're gaining popularity because they strike a balance between **power and simplicity**: giving AI much more capability, without making life harder for developers or compromising on safety.

How to Get Started with MCP Servers

Excited about MCP servers and want to dive in? Whether you're an AI developer looking to extend your app, or a researcher who wants to plug a model into new data, getting started with MCP is relatively straightforward. Here are some practical steps:

1. **Learn the Basics:** Begin by exploring the official documentation and community resources for the Model Context Protocol. Anthropic (the creators of MCP) provides documentation and examples that explain the MCP specifications and how the components (host, client, server) work. Understanding the core concepts will help you see how MCP fits your

needs. You might also find introductory tutorials or blog posts (like this one!) that demystify the concepts.

2. **Set Up an MCP-Capable Environment:** To experiment with MCP servers, you need an **MCP host** — essentially an AI platform that supports the protocol. One easy entry point is the **Claude Desktop app** (by Anthropic), which has built-in support for connecting to local MCP servers. You can download Claude Desktop and use it as the “AI side” of the equation. Other developer tools and IDEs are adding MCP support as well (for example, editors like Zed or VS Code extensions), but Claude Desktop is a beginner-friendly choice to see things in action quickly.
3. **Try a Pre-Built MCP Server:** No need to build from scratch on day one. There are many **pre-built MCP servers** available that you can install or run with minimal setup. For instance, you can grab an MCP server for Google Drive, Slack, or even a local file system. By installing one (often via a package or running a small local server program), you can then configure your AI host (like Claude Desktop) to connect to it. Anthropic has an open-source repository of example MCP servers and a **quickstart guide**. Following the guide, you might start a simple server (say, a Weather API server that can fetch forecasts) and see how the AI can use it. This hands-on trial will solidify your understanding — you’ll witness your AI model calling an MCP server to get info or perform a task.
4. **Build Your Own MCP Server:** Once you’re comfortable with using MCP servers, you can try creating one tailored to your needs. Thanks to SDKs in multiple languages (Python, TypeScript, Java, etc.), building an MCP server is meant to be accessible. You define what tools (functions) or resources your server offers, and implement how it handles requests. For example, if you have a custom database of research data, you could write an MCP server that listens for queries and returns results from that database. The official quickstart tutorials walk through making a basic

server (like a “Hello World” that perhaps returns a simple piece of data). By building your own, you not only tailor it to your environment, but you also contribute to the wider ecosystem if you choose to share it.

5. Join the Community and Iterate: MCP is a new and evolving trend, and there’s an active community of developers and researchers working with it. Consider joining forums, Discord/Reddit groups, or the GitHub community where people discuss their MCP projects. You can learn tips, discover new servers, and get help if you run into snags. Since this is an open-source movement, you’re encouraged to **contribute** back — maybe by improving an existing server or releasing your own connector for a tool that hasn’t been covered yet. This collaboration helps the technology grow. Also keep an eye on updates from Anthropic and others; features like remote MCP servers (not just local) and new host integrations are actively being developed, meaning the capabilities will expand over time.

By following these steps, you can gradually become proficient with MCP servers. Start small (use what’s already available), and step by step you’ll gain the skills to harness this powerful protocol in your own AI projects. It’s a rewarding process — there’s something almost magical about seeing your AI agent reach out to a new source of information or perform a task externally, all thanks to a few lines of MCP integration.

Conclusion

MCP servers represent a significant shift in how we integrate AI with the world around it. They turn isolated AI models into **connected, context-aware assistants** that can tap into live data and perform useful actions. In this article, we explored what MCP servers are (the connectors that link AI to external tools via the Model Context Protocol), why they’re generating so much excitement (solving the data isolation problem with a universal

standard), real examples of how they're used in industry and research (from fetching Slack messages to querying research papers), and how you can get started with this new technology.

The rise of MCP servers is making AI development more plug-and-play, much like adding Lego blocks to build capabilities — a stark contrast to the old days of custom, fragile integrations. For developers, this means faster development and more possibilities; for organizations, it means AI systems that can truly leverage existing data and infrastructure; for AI researchers, it's a playground to connect models with ever-richer context.

The trend is still in its early days, but it's growing rapidly. Big players and open-source communities alike are contributing to a vibrant MCP ecosystem. By understanding MCP servers now, you're getting a glimpse of the **future of AI applications** — one where AI is not a standalone black box, but an adaptable, integrated part of our digital world. Whether you simply use an AI that utilizes MCP servers, or you build the next great MCP connector yourself, one thing is clear: this new standard is making AI smarter, more useful, and more accessible for everyone.

Welcome to the era of AI that can truly “plug in” to everything! 🚀

Mcps

Mcp Server

AI

Machine Learning

Anthropic Claude



Written by Sebastian Buzdugan

358 followers · 349 following

Follow

No responses yet

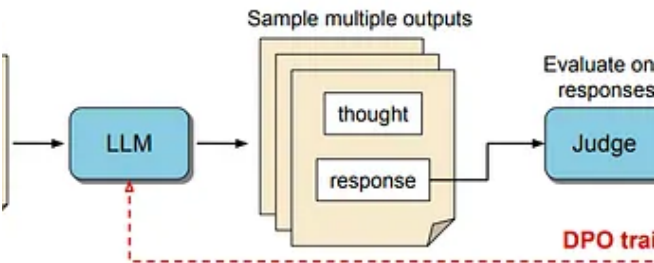


Juan Guillermo Resendiz

What are your thoughts?



More from Sebastian Buzdugan



Sebastian Buzdugan

Thinking LLMs: General Instruction Following with Thought Generatio...

In the realm of artificial intelligence, large language models (LLMs) have become...

Oct 18, 2024



64



1



Sebastian Buzdugan

What's the Best GPU for Fine-Tuning LLMs? A No-Nonsense...

Fine-tuning large language models (LLMs) requires a significant amount of VRAM, disk...


Mar 4



1






 Sebastian Buzdugan

Top 5 MCP Project Ideas for Seamless AI Integration

Anthropic's Model Context Protocol (MCP) is quickly becoming one of the most exciting...

Mar 16  64  2



 Sebastian Buzdugan

Claude Code and Cursor Are Leading in 2025, But Windsurf Ma...

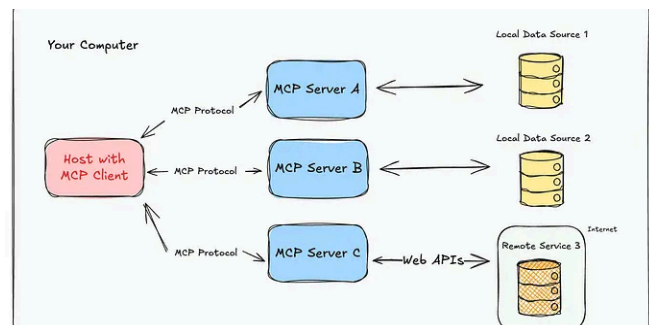
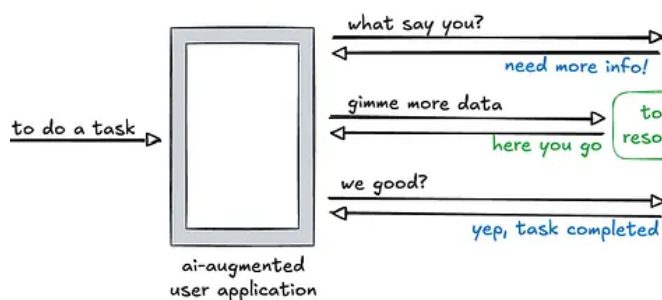
The AI-assisted coding era has matured. No longer just autocomplete toys or chat windo...

Jul 23  15



See all from Sebastian Buzdugan

Recommended from Medium

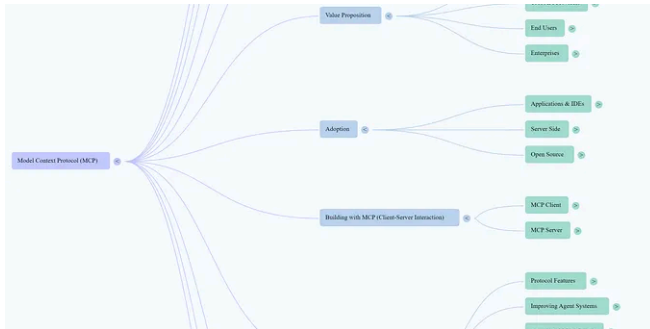


Heeki Park

Building an MCP server as an API developer

Anthropic released MCP at the end of November 2024. It took a few months for it t...

May 14 503 5



Dr. Nimrita Koul

The Model Context Protocol (MCP) —A Complete Tutorial

Anthropic released the Model Context Protocol(MCP) in Nov. 2024.

Mar 27 914 6



Sajith K

Creating an MCP Server and Integrating with LangGraph

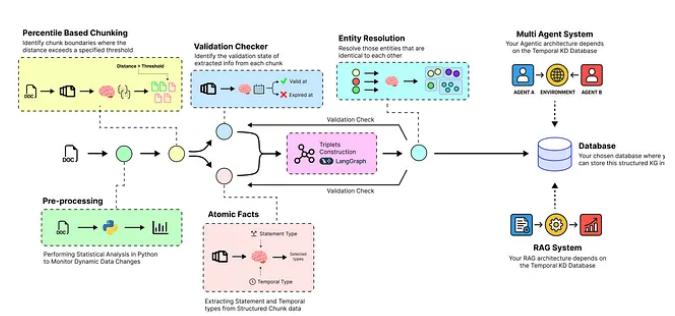
The Model Context Protocol (MCP) is an open protocol that standardizes how applications...

Edwin Lisowski

MCP Explained: The New Standard Connecting AI to Everything

How Model Context Protocol is making AI agents actually do things

Apr 15 1.4K 26



In Level Up Coding by Fareed Khan

Building a Temporal AI Agent to Optimize Evolving Knowledge...

Atomic Facts, Validation, Entity Resolution, KG and more

Aug 14 635 7



Joe Njenga

Claude Extends Context to 1 Million Tokens (Here's How I'm Using It)

In the past few days, Anthropic has released a massive update!



May 25



85



4



5d ago



116



5



See more recommendations