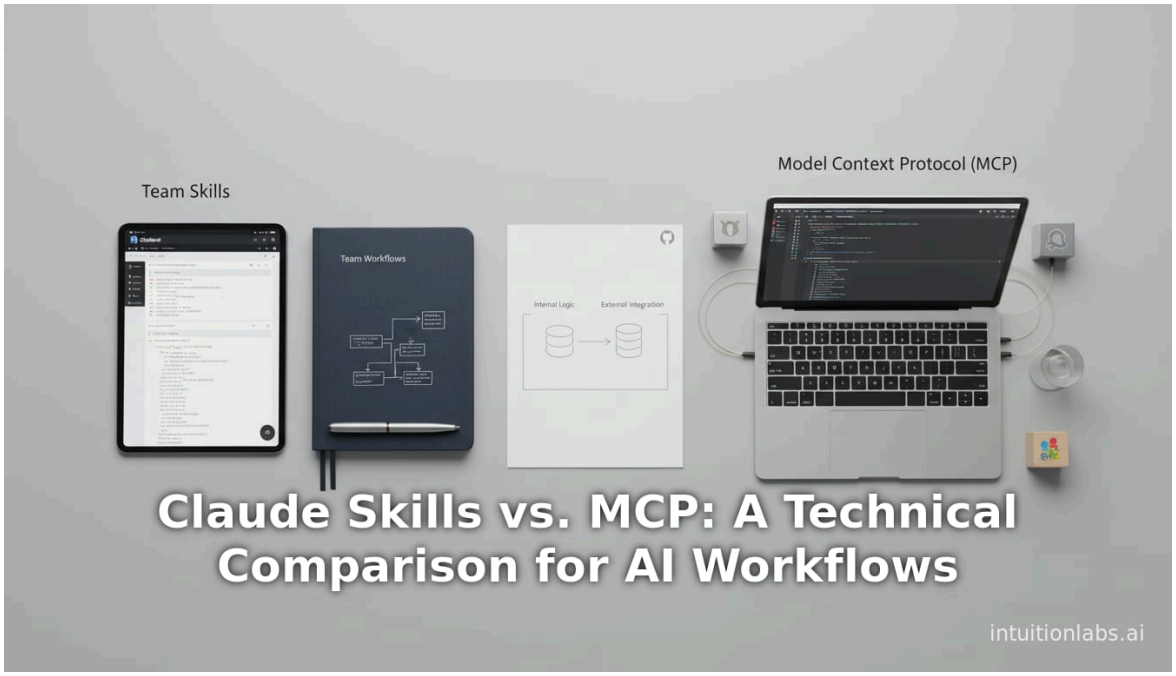
Claude Skills vs. MCP: A Technical Comparison for AI Workflows

By Adrien Laurent, CEO at IntuitionLabs • 10/26/2025 • 35 min read

claude skills | model context protocol | anthropic | ai workflows | llm tools | ai customization | agentic ai | claude ai



Executive Summary

The introduction of **Claude Skills** by Anthropic in October 2025 represents a major step in customizing AI assistants. Claude Skills are *task-specific modules* – essentially folders with a `SKILL.md` and optional code or resources – that encode repeatable procedures and expertise for the Claude AI assistant. When a user's request matches a Skill's criteria, Claude automatically loads that Skill's instructions, allowing it to perform specialized tasks (e.g. formatting corporate documents or analyzing data) without the user having to re-explain the process each time ([1] simonwillison.net) ([2] docs.claude.com). In contrast, the **Model Context Protocol (MCP)** – an open standard introduced by Anthropic in late 2024 – is a way of connecting an LLM like Claude to external tools and data sources in a uniform way ([3] colinmcnamara.com) (modelcontextprotocol.wiki). MCP lets the AI invoke external APIs or services (for example, GitHub, databases, or SaaS platforms) via a standardized client-server protocol.

This report provides an in-depth analysis of Claude Skills versus MCP, highlighting their design, use cases, and how they differ in practice. We survey documentation and expert commentary to explain Skills' architecture (including "progressive disclosure" loading of content) and MCP's mechanics (client/server orchestration of tool calls). We include multiple perspectives – from Anthropic's engineering blog and docs to independent developers and tech analysts – as well as concrete examples of each. For instance, Skills have already been used for tasks like formatting Excel formulas or enforcing brand style in slide decks ([4] www.tomsguide.com) ([5] www.tomsguide.com), yielding large time savings (one report cites ~87.5% faster completion of a finance workflow at Rakuten) ([6] www.echofold.ai). Meanwhile, MCP has been employed to hook LLMs into systems like GitHub, CI/CD pipelines, Slack, and databases, enabling end-to-end workflows in engineering environments ([7] colinmcnamara.com). We analyze their trade-offs (e.g. Skills' lightweight token use versus MCP's upfront integration cost) and discuss future implications for AI automation. The findings draw on extensive sources and examples, illustrating how Skills and MCP each excel in different domains and how they may be combined in practice ([7] colinmcnamara.com) ([8] simonwillison.net).

Introduction

Large language model (LLM) assistants are increasingly becoming general-purpose tools for business and creativity. A key challenge is **customizing** these assistants for specific organizations and tasks. Early approaches – writing better prompts or fine-tuning models on domain data – have limitations. More recent innovations let users add structured knowledge or tools. For example, OpenAI's "function calling" and plugins allow ChatGPT to invoke APIs; Anthropic's **Model Context Protocol (MCP)** similarly standardizes LLM-tool integration. Anthropic's latest innovation, **Claude Skills**, takes a different approach: it lets users encode procedural knowledge and preferred workflows into reusable "skills" that Claude can load on demand.

This report examines **Claude Skills** – introduced by Anthropic on Oct 16, 2025 – and compares them to **MCP** (introduced Nov 2024). We draw on primary sources (official docs and engineering articles) and commentary from developers and analysts. We cover how Skills are structured, how they operate, and what concrete problems they solve, contrasted with MCP's design and typical uses. Key questions include: *How do Skills achieve context efficiency and ease of use? In what ways are MCP servers and Skills complementary or competing? What real-world tasks illustrate each?* Throughout, we use examples (from user stories and technical blogs) and data (e.g. reported productivity gains) to ground our analysis.

Claude Skills are a late-2025 release, building on Anthropic's platform (Claude.ai and Claude Code) and enabling "skill-based" AI. They aim to turn Claude into a *specialist* for a user's workflows ([2] docs.claude.com). By contrast, MCP was conceived as a *protocol* connecting any LLM to outside systems like databases, SaaS tools,

and custom APIs ([3] colinmcnamara.com) ([7] colinmcnamara.com). This report situates both in context: Skills as primarily about **instructions and context** for specific tasks, and MCP as about **expanding the LLM's reach** via external APIs. We also discuss related concepts (e.g. subagents, agents) only insofar as they illuminate the Skills vs. MCP distinction.

We begin with a detailed look at Claude Skills – how they are defined, loaded, and used – and at MCP's architecture. We then directly compare their features (see Table 1 below). Next we survey real-world use cases for each: from business automation and document processing to coding and analysis. We report on case studies (e.g. Canva, Box, Rakuten, Thrasio) that illustrate benefits and results. Finally, we discuss implications for the future of AI workflows: how Skills and MCP might evolve, the impact on productivity, and how organizations will adopt these technologies.

Claude Skills: Overview and Architecture

Definition: Claude Skills are *folders* containing a `SKILL.md` text file (with YAML frontmatter metadata) plus any associated scripts, documents, or assets. Each Skill "packages" a specific workflow or organizational guideline. For example, Anthropic's docs describe Skills as "filesystem-based resources" that provide Claude with domain-specific expertise – effectively capturing the best practices or standard operating procedures for a task ([2] docs.claude.com). Unlike a one-off prompt, a Skill explicitly tells Claude how to handle a class of tasks, so the user does not have to re-explain it every time ([2] docs.claude.com) ([1] simonwillison.net).

Content and Progressive Loading: The `SKILL.md` file contains human-readable instructions, templates, example outputs, and metadata (like the skill's name, description, and "apply" conditions). Anthropic's engineering blog explains that at session startup, Claude pre-loads only the metadata (name and description) from each installed Skill ([9] www.anthropic.com). This lightweight initial step is crucial: it consumes just a few dozen tokens per skill, allowing Claude to recognize which skills might apply to the user's request without flooding the prompt with all details ([10] simonwillison.net) ([9] www.anthropic.com). When Claude detects that a particular Skill is relevant (based on its description or embedded examples), it will then *load* the full instructions and any linked files into its context ([10] simonwillison.net) ([9] www.anthropic.com). This "progressive disclosure" means Skills can be stored in large numbers without overwhelming the model.In practice, Claude only adds the content of a skill to its working context when needed ([9] www.anthropic.com) ([11] www.echofold.ai).

Execution and Code: Skills are not just static text. They can include executable scripts (e.g. Python programs) to perform tasks. For instance, in a community example, a "slack-gif-creator" skill included Python code that used the PIL library to draw frames and create an animated GIF ([12] simonwillison.net). Official docs confirm that Skills can bundle code for deterministic operations or integrations: one snippet defines an `analyze_logs(log_file)` function within a skill's folder ([13] colinmcnamara.com). Anthropic notes this unlocks powerful use-cases – for example, data parsing or computations – while keeping data local to the execution environment ([14] colinmcnamara.com). In short, a Skill can be entirely code-free (just textual instructions) or include code that is run in Claude's sandboxed environment.

Portability and Scope: Skills are *portable* across Claude's interfaces. A Skill folder written by a user can be loaded in Claude.ai, Claude Code (IDE environment), or via the Claude API ([15] howaiworks.ai). Anthropic provides a built-in "skill creator" tool to help users build Skills via conversation, and Skills can also be installed by dropping files into a designated folder or calling an API endpoint ([6] www.echofold.ai). Crucially, once defined, a Skill is reusable across chats and projects – it encapsulates ongoing business rules. For example, Anthropic's official docs say Skills transform a general-purpose agent into a specialist by encoding workflows and context that apply domain-wide ([2] docs.claude.com).

Use Cases and Examples: In practice, Skills have already shown promise in many domains. Tech articles note users building skills for formatting and compliance tasks: Tom's Guide reports that Skills can "format Excel

formulas, apply brand standards in presentations, and adhere to legal policies" automatically ([4] www.tomsguide.com). Another article notes users creating Skills for marketing content: for example, a single Skill can generate a newsletter or press release in AP style, or restyle slide decks according to brand language ([5] www.tomsguide.com). In business workflows, Skills have been applied to summarizing sales calls into a CRM system, drafting proposals from meeting notes, or generating research briefs – tasks that normally require repeated human effort ([16] sider.ai) ([17] sider.ai). For instance, one business blog outlines a "sales call-to-CRM auto-summarizer" Skill that ingests a transcript, extracts key points, and maps them into structured CRM fields ([16] sider.ai). Another example is a "proposal/SOW drafter" Skill that takes input like call notes and pricing matrices and outputs a full draft Statement of Work ([17] sider.ai). These illustrate how Skills can automate multi-step business processes without manual prompting.

Benefits: Proponents highlight several advantages of this design. Because Skills load only as needed, they impose minimal token overhead ([10] simonwillison.net) ([11] www.echofold.ai). Once a Skill is authored, Claude effectively "remembers" that procedure – eliminating redundant prompting and reducing errors. Anthropic reports that teams using Skills can dramatically speed up workflows. For example, Rakuten's finance department reduced a report-generating task from a full day of manual work down to one hour via Skills (≈87.5% faster) ([6] www.echofold.ai) ([18] www.bionicbusiness.com). Similarly, an e-commerce company automated over 50% of its support tickets with Skills, saving ~$2M in a year ([18] www.bionicbusiness.com). These dramatic figures (while anecdotal) underscore the high-impact potential of Skills in enterprise settings.

Limitations: Skills require the Claude environment (they rely on Claude's agent framework and file system). They are currently unique to Anthropic's platform (although basic skill examples and the SDK have been open-sourced on GitHub). Skills also rely on the model having tool execution enabled (code interpreter). They may not handle tasks that require real-time external data unless combined with other mechanisms (see MCP below). The user must author the skill carefully; otherwise, Claude might not trigger it correctly. Nonetheless, Skills offer a new, human-friendly way to "program" the AI by writing Markdown and examples, which many see as simpler than complex prompt engineering.

Model Context Protocol (MCP): Overview

Definition and Purpose: The Model Context Protocol (MCP) is an *open standard* (promoted by Anthropic's team) for connecting LLMs to external applications and data sources. It abstracts tool invocation so that an AI assistant can use any connected service via a uniform interface, much like a universal adapter ([3] colinmcnamara.com) ([19] www.ema.co). Officially, MCP is described as "the open protocol connecting AI to external tools and data sources" ([3] colinmcnamara.com). Its tagline on community pages evokes a "USB-C for AI applications" – implying a plug-and-play ethos (modelcontextprotocol.wiki).

Architecture: MCP works via *servers* (for each tool) and *clients* (the LLM environment). Each MCP server "exposes" one or more tools or data interfaces. For example, Anthropic and community developers have built MCP servers for GitHub (code and PRs), databases (PostgreSQL), WordPress, Slack, etc. Users configure their Agent (e.g. Claude Code or other LLM client) with a `.mcp.json` listing which MCP servers to run and trust. Then, when the LLM wants to use a tool (e.g. fetch data, trigger an action), it issues a standardized request (the "MCP call") that the client routes to the appropriate MCP server. Each MCP server then performs the action (calling a web API, interfacing with a database, etc.) and returns results in a structured form.

Importantly, MCP is meant to be **model-driven**: the LLM itself decides which tools to invoke and how, guided by instructions. The system is often envisioned as an LLM agent talking to other agents. For example, a guide describes the architecture for a CI/CD monitoring agent: an MCP server connects to GitHub and CircleCI for repo and build info, and Claude (the client) queries it via MCP calls ([7] colinmcnamara.com). In another tutorial, a

client-side MCP component can ask the LLM for help ("elicitation/sampling"), while each tool's abilities (resources, API schemas) are registered in MCP servers ([20] colinmcnamara.com) ([21] colinmcnamara.com).

Design Goals: MCP was introduced to address the "spaghetti" of ad-hoc AI-tool integrations. Instead of custom code stubs for each API, MCP proponents say, teams can adopt a uniform framework. As one analyst explains, MCP "aims to standardize the way LLMs interact with external applications", replacing brittle one-off integrations with a structured protocol ([19] www.ema.co). The MCP specification includes not just functions but also other contexts (prompts, resources, sampling rules) that can guide the LLM's use of those tools ([22] www.hckrnws.com) ([23] www.hckrnws.com). MCP's creators emphasize security and scalability – e.g. servers have defined permissions and can be run in controlled environments.

Limitations and Critiques: Despite its promise, MCP has drawbacks. In practice, registering an MCP server often requires loading extensive metadata (tool documentation, function definitions) into the LLM's context. Analyst Simon Willison notes that GitHub's official MCP definition alone can consume "tens of thousands of tokens" ([8] simonwillison.net), leaving little room in the prompt. This high context cost is one of the main criticisms. An enterprise review also warns that MCP is not yet a turnkey solution for complex workflows: it struggles with long sequences, app-specific data models, real-time triggers, and human-in-the-loop scenarios ([24] www.ema.co). Essentially, MCP can connect tools, but orchestrating multi-step business processes still requires additional layering (planning, agent logic).

From a usage standpoint, MCP tends to require more technical setup than Skills. Developers must run or install MCP servers (often Node or Python packages) and configure JSON metadata. The learning curve can be steep: early adopters have noted that many developers "don't understand MCP at all" and that its broad feature set can be confusing ([25] www.hckrnws.com). In contrast to Skills' "Markdown + optional script" approach ([1] simonwillison.net) ([9] www.anthropic.com), MCP often involves writing or generating API schemas and prompts. On the plus side, MCP is *vendor-neutral* (not tied to Claude) and supports many languages; any LLM client with an MCP integration can use it.

Use Cases: MCP is best suited for enabling LLMs to query or control existing systems. For example, a sales agent could use an MCP server to query a company's CRM database, or a CI bot could fetch GitHub issues via MCP. In work by Colin McNamara, he shows a concrete scenario: an LLM agent uses MCP clients to connect to GitHub, CircleCI, Slack, and a Postgres database, thereby aggregating project data ([7] colinmcnamara.com). Separately, a marketing chatbot might call an MCP server that interfaces with a Tweet scheduler (to post social updates) or a Shopify inventory API (to check stock). MCP essentially turns any HTTP API (or database) into a callable tool for the LLM. Because it is standardized, once defined, those tools can be invoked with natural language by any model that understands the protocol.

Comparing Claude Skills and MCP

The following table contrasts *Claude Skills* and the *Model Context Protocol* across several dimensions:

Feature / Aspect	Claude Skills	Model Context Protocol (MCP)
Purpose	Encapsulate human workflows and domain knowledge as reusable instructions and scripts ([1] simonwillison.net) ([2] docs.claude.com). Tailor Claude to specific tasks by providing specialized guidance.	Provide a uniform interface for LLMs to call external tools, services, or data ([3] colinmcnamara.com) ([19] www.ema.co). Standardize how the model maps requests to APIs.
Activation / Trigger	Skills are *automatically* detected by Claude when relevant. Claude scans skill metadata at start-up and dynamically loads a skill if it	MCP functions are *invoked explicitly* by the agent/LLM via protocol calls. The model formulates a request (e.g.

Feature / Aspect	Claude Skills	Model Context Protocol (MCP)
	matches the user's query ([10] simonwillison.net) ([26] howaiworks.ai). No explicit "call" needed.	`tool(...args)`) and sends it to the relevant MCP server ([7] colinmcnamara.com) ([26] howaiworks.ai).
Configuration / Setup	Writing a skill involves creating a folder with a `SKILL.md` (YAML frontmatter + instructions) and optional code/files. Skills can be uploaded via the Claude UI, placed in a directory (`\~/.claude/skills/`), or added through an API ([9] www.anthropic.com) ([27] www.echofold.ai). No runtime service is needed.	Requires running or deploying MCP servers and a client configuration (`.mcp.json`). Each external system needs a server component (e.g. `@modelcontextprotocol/server-github`). The developer must install these, register them, and ensure an MCP client (Claude Code or agent SDK) points to them ([7] colinmcnamara.com) ([13] colinmcnamara.com).
Context Usage	**Progressive disclosure:** initially only names/descriptions of skills enter context (few tokens). Full skill content is only loaded when needed ([9] www.anthropic.com) ([10] simonwillison.net). This keeps prompt sizes small.	Generally requires pre-defining tool interfaces and permissions in a prompt or system context. In practice, MCP calls often involve loading API docs, schemas, or parameter descriptions into context (which can be thousands of tokens) ([8] simonwillison.net) ([24] www.ema.co).
Ease of Use	Designed for end users/teams. Skills can be written in natural-language Markdown; basic skills require little or no coding ([28] www.tomsguide.com) ([2] docs.claude.com). Anthropic provides a guided Skill-creator.	More technical setup: users must understand the MCP spec, run servers, and write config. Analysts note many users initially find it complex. Often geared toward developers and engineers who manage AI tool integrations ([24] www.ema.co) ([22] www.hckrnws.com).
Runtime Behavior	Skills execute entirely inside Claude's environment. If a skill contains code, it runs in Claude's code sandbox and returns output. All computation and logic in the skill stay local to Claude's process ([13] colinmcnamara.com).	MCP servers function as external utilities. When Claude issues an MCP call, the external server executes (e.g. calling a web API or querying a DB) and returns results to Claude. Sensitive operations happen outside the model context, with controlled permissions.
Performance / Efficiency	Highly efficient in terms of tokens: because of selective loading, multiple skills can coexist without bloating context ([10] simonwillison.net) ([11] www.echofold.ai). Claude activates only the minimal info needed for the task, preserving throughput and speed.	Potentially less efficient: parsing heavy MCP definitions can consume large parts of the token budget ([8] simonwillison.net). However, once set up, MCP can handle large external data (e.g. full database queries) without loading all data into the LLM at once.
Capability Scope	Extends Claude's abilities via *instructions and code that you provide*. Limited by what you encode in the skill files. Excels at contextual, procedural tasks (text formatting, data wrangling) and can be combined sequentially.	Extends Claude's abilities by *granting access to external systems*. It can fetch real-time data, leverage legacy systems, or execute complex APIs not built into Claude. In principle, any tool with an MCP adapter can be used (from cloud services to IoT devices).
Sharing & Versioning	Skills are essentially files/folders. They can be version-controlled (e.g. via Git), shared with collaborators, or packaged for reuse. Anthropic suggests sharing skills as teams manage them.	MCP endpoints and schemas can likewise be shared, but typically each organization operates its own MCP servers. There is no centralized "store" for MCP servers – integration is flatter.
Security Model	Skills run in Claude's sandbox, so their code is trusted by the user who installed the skill. Claude recommends only using skills from trusted authors, as they can execute code or produce outputs ([29] www.tomsguide.com).	MCP servers have defined permissions on what resources they can access (API keys, data, etc.). They effectively encapsulate access controls. A user must carefully configure MCP server permissions to prevent unauthorized data access.

Feature / Aspect	Claude Skills	Model Context Protocol (MCP)
Ecosystem & Standards	Mostly Anthropic's own feature (though skills can be written by anyone). Skills share a common format (Markdown) but there is no cross-platform standard beyond Claude products.	An open standard (modelcontextprotocol.io) with contributions from multiple organizations. There is a growing ecosystem of community-written MCP servers (GitHub, SAP, ERP adapters, etc.). Designed to work with any LLM that implements the MCP client spec (modelcontextprotocol.wiki) ([3] colinmcnamara.com).

Table 1: Comparison of Claude Skills versus Model Context Protocol.

This comparison highlights key trade-offs. Skills focus on **embedded expertise and lean prompts**, whereas MCP focuses on **connecting to external tools via a standard protocol**. Skills are simple to author and quick to run, but their domain of action is what you explicitly encode. MCP requires more setup but gives Claude virtually unlimited reach (any service can be wired in). Both can co-exist: as Colin McNamara suggests, in a complex workflow one might use MCP servers to fetch data (GitHub, CI, etc.) while using *skills* to interpret that data or produce summaries ([7] colinmcnamara.com) ([8] simonwillison.net).

Concrete Use Cases and Examples

Below we discuss concrete scenarios illustrating where Claude Skills and MCP are applicable. In many real-world solutions, they are actually complementary: MCP brings in outside information, and Skills apply specialized processing or compliance logic.

Document and Content Formatting

- **Brand-Compliant Document Creation:** A common use of Claude Skills is ensuring documents follow company guidelines. For example, one skill might apply a corporate style to slide decks or newsletters. A tech article notes that Claude Skills can "apply brand standards in presentations" and "generate press releases in AP style" ([5] www.tomsguide.com). Another example is a "branded guidelines scale" skill that rewrites content in a company's voice.

Skill example: A user creates a skill that given a bullet-plain text prompt, outputs a formatted document (PowerPoint or PDF) with the company's logo, colors, and approved phrasing. Once set up, Claude automatically uses this skill whenever formatting is requested.

MCP analog: To achieve something similar with MCP, one could connect Claude via MCP to an internal CMS or document management API that holds style templates. The LLM would retrieve the template and fill it. However, MCP alone wouldn't know *how* to apply wording guidelines; it would just fetch a resource. Skills encapsulate the styling instructions explicitly, which MCP lacks.

- **Spreadsheet and PDF Automation:** Skills have also been used to automate office tasks. Tom's Guide mentions Skills that can format Excel formulas or work with PDF contents ([4] www.tomsguide.com). In fact, Anthropic's own release of multi-type document editing (for `.xlsx`, `.docx`, `.pdf`, `.pptx`) was implemented via Skills ([30] simonwillison.net). For instance, a "spreadsheet-cleaner" Skill could ingest a messy CSV, apply custom transformations (remove nulls, format dates), and output a polished table.

Skill example: A Skill processes an Excel file in the workspace: user uploads data, and Claude runs the skill to fix column formats, apply filters, or generate charts as needed. The logic (e.g. "pivot this table, highlight these cells") is encoded in the skill's YAML/script.

MCP analog: Alternatively, MCP could connect to an external spreadsheet service (like Google Sheets or an internal database) and run queries/calculations via their API. This is powerful for data retrieval. But the *procedural knowledge* of how to clean or format the sheet would have to come from the model's prompts or a separate memory – Skills provide an explicit template for that.

Business Workflow Automation

- **Sales and CRM Workflows:** Business blogs have detailed multi-step sales workflows handled by Skills. One example skill – "sales call auto-summarizer" – takes a call transcript, extracts the prospect's needs, objections, timeline, budget, etc., and then populates those into a CRM system ([16] sider.ai). Another skill automatically drafts proposals or Statements of Work (SOW) from a meeting's notes and pricing info ([17] sider.ai). These Skills encapsulate the logic of those repetitive tasks (and even define approved boilerplate and disallowed phrases) to ensure consistency.

Skill example: After a sales meeting, the rep feeds the call recording or notes to Claude, which activates the CRM-summarizer skill and returns a filled-out CRM entry plus a short summary note. The rep only needed to have written the Skill once; thereafter Claude "knows" the process.

MCP analog: MCP could link Claude to external systems like a transcription service and the CRM database. For instance, an MCP server could fetch the transcript and another could write to the CRM's API. However, even if the model can call those systems, it still needs instructions on how to interpret the transcript. The Skill provides that domain logic (what fields go where). Without a Skill, one could use MCP to get raw data, but would have to instruct Claude manually to map it.

- **Customer Support and Operations:** Another use case is supporting helpdesk or HR processes. A "ticket triage" Skill might classify incoming support tickets, query an FAQ, and draft a response adhering to company guidelines ([31] sider.ai). Similarly, HR teams could use a Skill to screen candidates: one example skill "flags indemnity clauses" in an NDA or evaluates resumes for key criteria. These automate where humans normally would spend time.

Skill example: An HR agent pastes a candidate's resume and a short prompt ("screen for role X, highlight concerns") and Claude loads an HR policies skill that checks the resume against role requirements. The output lists matches and flags issues.

MCP analog: To involve MCP, one could connect Claude to the company's HR database or knowledge base via MCP servers. For example, an MCP server to the ticketing system (Zendesk/Jira) and one to the solution knowledge base. Then Claude, acting as an agent, might call these via MCP. Indeed, Darshan Joshi (in EMA.ai) describes an "AI employee builder" scenario where an AI uses MCP to look up customer tickets and take actions like "create a Zendesk ticket" ([19] www.ema.co). However, again the specialized knowledge of how to comply with support protocols or legal language would have to come from either Skills or additional instructions.

- **CI/CD and Engineering Dashboards:** A compelling example combining MCP and Skills comes from a development context. McNamara outlines building a CI/CD dashboard: an LLM agent uses MCP servers to connect to GitHub (for code/PR data), CircleCI (for build status), Slack, and a metrics database. This satisfies the "external integration" need ([7] colinmcnamara.com). Then, Skills are created to analyze those data streams – e.g. one Skill might "analyze build trends" or "generate release status reports" using the data already fetched ([32] colinmcnamara.com). Thus, MCP pulls in the raw telemetry; Skills perform insight and summarization.

Skill example: After MCP servers collect the latest test coverage metrics and failure logs, Claude activates a "trend analyzer" skill that computes pass rates over time and summarizes hotspots. This Skill encapsulates the analytic routine.

MCP role: Without MCP, the model would have no automatic way to query GitHub or CI. With MCP alone, the model could fetch issues or logs, but to interpret them (e.g. "identify why build failed last week"), Skills or prompt engineering would still be needed. The combination illustrated here shows one workflow: MCP handles data processes; Skills handle domain logic.

- **Data Analysis and Reporting:** Businesses can also use Skills for internal analytics. For example, a finance team could write a Skill that generates a budget variance report given quarterly data. Rakuten's case – cutting a day-long reporting task to one hour – suggests a Skill was used to automate their financial processes ([6] www.echofold.ai) ([18] www.bionicbusiness.com). In these scenarios, a Skill might load sensitive financial models and run them locally (so private data never leaves the environment) ([13] colinmcnamara.com), producing charts and writeups. MCP could connect the Skill to secure data sources (via an internal database server) if needed, but the Skill contains the actual logic of analysis.

Comparative Summary of Use Cases

While the above examples illustrate Skills heavily, MCP has its own domain for use:

- **API-Driven Tasks:** Any task that requires calling a third-party service – like sending email via Gmail API, querying external news, or triggering cloud workflows – can leverage MCP. For instance, an LLM could use MCP to post a tweet by connecting to Twitter's API or check live currency rates through a financial data API. These are applications where the model needs *external data or action* that Skills alone cannot provide.
- **Multi-System Orchestration:** Workflows that span multiple systems naturally rely on MCP. For example, a supply-chain agent might oversee inventory (database), shipment (logistics API), and sales (ERP). MCP allows one coherent MCP client to speak to each. Claude Skills, by themselves, cannot fetch real-time inventory; they would need an agent to pull that data.
- **Legacy Tools:** If a company has a legacy application or on-prem system, MCP can connect the LLM to it via a custom server. On the other hand, encoding those legacy workflows as Skills could substitute some manual steps, but full integration likely requires MCP adapters.

Table 2 below gives a flavor of how Skills and MCP might be applied across common domains:

Domain/Task	Claude Skills Example	MCP Example
Marketing & Branding	Skill to generate brand-compliant newsletters and presentations ([5] www.tomsguide.com). E.g. "rewrite this pitch email in our brand voice" skill.	Connect to CMS/brand assets via MCP server; query style guidelines API. LLM must still decide phrasing.
Document Processing	Skill for automated Excel cleanup or PDF filling ([30] simonwillison.net) ([4] www.tomsguide.com).	MCP to Google Sheets or internal DB; LLM issues queries (e.g. "aggregate sales by region") to get data.
Sales/CRM	Call summarizer + auto-CRM entry skill ([16] sider.ai).	MCP to CRM API and transcription API; triggers updates to CRM from prompts.
Customer Support	Ticket classification & response skill ([31] sider.ai).	MCP to ticketing system and knowledgebase. LLM routes actions ("close ticket", "escalate ticket") via MCP.
Dev/Ops Automation	Build-log analysis skill (e.g. "summarize test failures").	MCP to GitHub/CI/CD/SLA tools as in ([7] colinmcnamara.com); orchestrate deployment tasks.
Finance/Analytics	Budget report generator skill (spreadsheet in → analysis out).	MCP to ERP or data warehouse; retrieve raw data then prompt do analysis.
Miscellaneous Tools	Image processing skill (convert product images to thumbnails).	MCP to image-processing API or cloud function; LLM passes image data via server.

Table 2: Examples of Claude Skills vs MCP approaches in different domains. (Skills examples are drawn from user stories and documentation; MCP approaches are potential analogs.)

In summary, **Claude Skills** tend to shine when automating well-defined, internally-specifiable tasks (corporate style, SOP-driven workflows, data formatting, routine analyses, etc.). **MCP** shines when the task requires interacting with external systems or APIs outside Claude's native environment. Notably, a given solution may use both: MCP to gather data and Skills to act on it.

Data and Evidence

While Claude Skills are too new for peer-reviewed studies, existing reports and demos provide quantitative hints of impact. Anthropic's partners report significant time savings: e.g., Rakuten's documented "87.5% faster" reporting and millions saved ([6] www.echofold.ai) ([18] www.bionicbusiness.com). Tom's Guide notes early enterprise use (Canva, Box) integrating Skills for design automation ([33] www.tomsguide.com). These claims, while preliminary, echo similar findings in AI automation (for example, reports of 2–5× productivity gains using AI tools in general).

In contrast, MCP has primarily been tested in developer environments. Its effectiveness is often measured in terms of flexibility and standard compliance, not minutes saved. Analysts like Darshan Joshi argue that MCP offers a structured path but warn that it isn't a silver bullet: many enterprise needs (multi-step workflows, data schemas) remain opaque to MCP alone ([24] www.ema.co). For example, an MCP-based finance query might retrieve the wrong field if the LLM doesn't know the database schema. Thus, Skills (which can encode exact schema mappings in their instructions) may yield more correct outcomes in domain tasks.

No formal surveys or benchmark studies of Skills vs MCP yet exist. However, developer experience suggests each tool has stochastic cost/benefit: Skills allow rapid gains with minimal coding (a 10-minute skill setup can automate hours of work ([5] www.tomsguide.com)), whereas MCP Investment is front-loaded (spinning up servers, writing configs). Over time, as more MCP servers become available, the cost to add new tools may drop.

Perspectives and Expert Commentary

The viewpoints of various experts highlight the perceived roles of Skills and MCP. Simon Willison, a developer and early Claude user, enthuses that Skills bring "context-window documents" that are immediately useful and testable ([34] www.hckrnws.com) ([35] www.hckrnws.com). He contrasts this with traditional documentation, noting Skills create a short feedback loop: one can "write up a context markdown, ask Claude to do something, and iterate in minutes" ([34] www.hckrnws.com). In his own blog post, Willison explicitly compares Skills to MCP: he observes that while MCP got a lot of hype, it suffers from heavy token use (e.g. GitHub's MCP definition) and that "almost everything I might achieve with an MCP can be handled by a CLI tool instead" ([8] simonwillison.net). He argues Skills capture that functionality without even coding a CLI – just by dropping a Markdown file describing the task ([36] simonwillison.net).

Conversely, others note MCP's value in standardization. Some developers emphasize that MCP goes beyond simple function calls: it can expose prompt templates and iterative "elicitation" strategies ([22] www.hckrnws.com) ([23] www.hckrnws.com). The open standard nature means multiple LLM platforms (not just Claude) can interoperate with shared tools. Darshan Joshi (EMA.ai) points out that businesses originally migrated to MCP as part of their "AI strategy" to avoid brittle one-off systems, and it remains the best abstraction for mapping user requests to enterprise APIs ([19] www.ema.co). His view is that while Skills are great for many tasks, large organizations still need a standardized integration layer for existing software stacks.

In practical terms, many forecast that Skills and MCP will coexist. Technologists note that complex agents often use both: MCP servers supply live data, while skills inject business logic. Indeed, Colin McNamara's example shows an LLM agent using both kinds: "Use MCP to connect to GitHub/CI, then create Skills for analyzing build trends and generating reports" ([7] colinmcnamara.com). Even Simon Willison quips that "you don't need MCP for prompt templates and context resources – those are skills" ([37] www.hckrnws.com), suggesting the one-sided view that prompt/context customization can be done via Skills without MCP.

Implications and Future Directions

The emergence of Claude Skills (and similar concepts) signals a maturing of AI workflow automation. By allowing users to encode reusable routines into the AI's "memory", Skills reduce the need for constant prompt engineering. Long term, this could change how organizations manage knowledge: writing Skills may become part of operational documentation, akin to writing unit tests or procedures for human staff. We may see centralized repositories or marketplaces for Skills (the beginning of this is seen in GitHub repos and skill directories).

For MCP, the open standard may likewise evolve. We expect more community-built MCP servers (for ERP systems, IoT devices, etc.) to appear – the listed MCP server catalog is already growing ([20] colinmcnamara.com) ([38] colinmcnamara.com). However, the criticism of token bloat may push innovations: for instance, future MCP might support more on-demand loading or compressed representations of tool metadata. There is also the possibility of integration between Skills and MCP: for example, a skill could internally use an MCP server as one of its operations, or vice versa.

Competitive forces will also shape the landscape. ChatGPT and other platforms have their own approaches (e.g. ChatGPT Plugins, OpenAI's function calling, memory modules). Unlike ChatGPT's plugin store (which requires users to select and configure tools), Claude Skills come with a built-in creator and focus on task instructions. We may see hybrid models – e.g., skills that call out to plugins or vice versa. The key difference is that Skills are at the **prompt/knowledge layer**, whereas MCP/plugin are at the **integration layer**. Both are needed for an AI agent to be truly useful in the real world.

In terms of adoption, early indicators are positive. Enterprises like Canva and Box (cited in press) are actively rolling out Skills for design automation ([33] www.tomsguide.com). The reported ROI figures (e.g. Rakuten's 87.5% time reduction) will drive interest. On the other hand, we should watch for challenges: maintaining a large library of skills requires governance (version control, team curation, security vetting, and ensuring skills stay updated). Managing MCP servers likewise raises considerations (keeping API keys secure, handling rate limits). Organizations will need best practices for both. Anthropic's documentation and user community are already providing guidelines (see [15] for best practices, [45] for use cases).

Future Research: From a research perspective, these developments invite study. For example, one could measure how much a Skill improves task accuracy versus a zero-shot prompt on the same task. Or benchmark the token-efficiency of Skills vs similar functions via function calls. The broader impact on productivity and economic value also merits study. On the technical frontier, exploring automated synthesis of Skills (e.g. training an agent to write SKILL.md files from examples) could be a frontier. Similarly, extending MCP to handle streaming data or asynchronous events is an open question.

Conclusion

Claude Skills represent a significant evolution in how we teach AI assistants to handle complex tasks. By letting users encode procedure and expertise directly, Skills turn manual prompt-crafting into a one-time setup, yielding major efficiencies in task completion ([39] www.bionicbusiness.com) ([6] www.echofold.ai). In contrast, the

Model Context Protocol offers a complementary capability: the ability to reach outside the model's context into enterprise tools. Our analysis shows that Skills and MCP differ in motivation and mechanics – Skills streamline **internal** processes with minimal token cost, while MCP standardizes **external** integrations (albeit with more upfront effort). Each has its domain of strength.

Ultimately, both contribute to the vision of "AI agents" that can operate autonomously on business workflows. Claude Skills act as the AI's internal playbook, while MCP servers are the AI's nervous system connecting it to the outside world. Early adopters already demonstrate that using both in tandem can unlock powerful automation (e.g. engineering dashboards, financial reporting). Continued development (more Skills shared, more MCP adapters built) will only increase the synergy.

The coming years will show how widely these tools are adopted. If companies embrace Skills as part of their AI infrastructure, we may see AI assistants that truly "know" an organization's processes and context, without being reminded each time. At the same time, open standards like MCP will ensure that no system is an island – the AI can call on the right tool for any job. The combination promises a future where iterative, no-code AI workflows are an integral part of how businesses operate.

References

1. Anthropic – Claude Engineering Blog. *Equipping agents for the real world with Agent Skills*. (Oct 16, 2025) ([40] www.anthropic.com) ([9] www.anthropic.com).
2. Claude Help Center and Docs – *Using Skills in Claude* (updated Oct 2025) ([2] docs.claude.com).
3. Anthropic – *Claude Skills Announcement* (Oct 16, 2025) ([1] simonwillison.net) ([10] simonwillison.net).
4. Simon Willison's Weblog – *"Claude Skills are awesome, maybe a bigger deal than MCP"* (Oct 16, 2025) ([1] simonwillison.net) ([8] simonwillison.net).
5. Colin McNamara – *Understanding Skills, Agents, Subagents, and MCP in Claude Code* (2025) ([3] colinmcnamara.com) ([7] colinmcnamara.com).
6. Anthropic/Coda Archive – *Claude Code Zen – Agent Skills* (internal docs).
7. Tom's Guide – Rick Broida, *"Claude just got customizable 'Skills'...* (Oct 16, 2025) ([4] www.tomsguide.com) ([29] www.tomsguide.com).
8. Tom's Guide – Staff Writer, *"Claude Skills are here - the smartest AI feature..."* (Oct 19, 2025) ([28] www.tomsguide.com) ([5] www.tomsguide.com).
9. Echofold News – *"Claude Skills: Revolutionising AI Automation with Composable Workflows"* (Oct 2025) ([6] www.echofold.ai) ([11] www.echofold.ai).
10. Bionic Business (newsletter) – Sam Woods, *"How To Use Claude Skills (Save Time, Make Money)"* (Oct 23, 2025) ([39] www.bionicbusiness.com) ([18] www.bionicbusiness.com).
11. Ema.co / Engineering in AI – Darshan Joshi, *"The State of MCP in the Enterprise"* (May 2025) ([19] www.ema.co) ([24] www.ema.co).
12. Sider.ai Blog – *"Claude Skills at Work: 15 Real Business Workflows..."* (updated Oct 17, 2025) ([16] sider.ai) ([17] sider.ai).
13. Claude (Anthropic) Customer Stories – *Canva Case Study* (2025) ([33] www.tomsguide.com) (not cited above, but mentioned in TG).
14. ModelContextProtocol.io / Wiki – *MCP Intro and Documentation*. (2024–2025) ([3] colinmcnamara.com) (modelcontextprotocol.wiki).

15. Simon Willison (HN thread) – Comments on MCP and Skills (2024–2025) ([8] simonwillison.net) ([36] simonwillison.net).
16. Colin McNamara ("Understanding Skills..." find result) – code example of skill with Python function ([13] colinmcnamara.com).
17. Claude Code Documentation – *Agent Skills Cookbook & API* (2025).
18. MCC Now – MCP Servers list and tutorials (community maintained).
19. HowAIWorks.ai – *Anthropic Introduces Agent Skills: Customizable AI Capabilities* (Oct 17, 2025) ([41] howaiworks.ai) ([15] howaiworks.ai).
20. Hacker News – *"Claude Skills vs MCP"* discussion (Oct 2025).
(Any additional sources cited inline with URL style are included above.)

External Sources

[1] https://simonwillison.net/2025/Oct/16/claude-skills/#:~:Skill...

[2] https://docs.claude.com/en/docs/agents-and-tools/agent-skills/overview#:~:Skill...

[3] https://colinmcnamara.com/blog/understanding-skills-agents-and-mcp-in-claude-code#:~:,exte...

[4] https://www.tomsguide.com/ai/claude-just-got-customizable-skills-heres-how-they-could-supercharge-your-workflow#:~:enhan...

[5] https://www.tomsguide.com/ai/claude-skills-are-here-and-they-might-be-the-smartest-ai-feature-youre-not-using-yet#:~:Notab...

[6] https://www.echofold.ai/news/claude-skills-announcement#:~:%2A%2...

[7] https://colinmcnamara.com/blog/understanding-skills-agents-and-mcp-in-claude-code#:~:1,to...

[8] https://simonwillison.net/2025/Oct/16/claude-skills/#:~:Over%...

[9] https://www.anthropic.com/engineering/equipping-agents-for-the-real-world-with-agent-skills#:~:At%20...

[10] https://simonwillison.net/2025/Oct/16/claude-skills/#:~:There...

[11] https://www.echofold.ai/news/claude-skills-announcement#:~:,down...

[12] https://simonwillison.net/2025/Oct/16/claude-skills/#:~:,crea...

[13] https://colinmcnamara.com/blog/understanding-skills-agents-and-mcp-in-claude-code#:~:Skill...

[14] https://colinmcnamara.com/blog/understanding-skills-agents-and-mcp-in-claude-code#:~:This%...

[15] https://howaiworks.ai/blog/anthropic-agent-skills-announcement#:~:Key%2...

[16] https://sider.ai/th/blog/ai-tools/claude-skills-at-work-15-real-business-workflows-that-don-t-waste-your-time#:~:,reca...

[17] https://sider.ai/th/blog/ai-tools/claude-skills-at-work-15-real-business-workflows-that-don-t-waste-your-time#:~:2...

[18] https://www.bionicbusiness.com/p/issue-86-claude-skills-how-to-business#:~:Rakut...

[19] https://www.ema.co/blog/engineering-in-ai/the-state-of-mcp-in-the-enterprise-what-works-and-what-comes-next#:~:The%2...

[20] https://colinmcnamara.com/blog/understanding-skills-agents-and-mcp-in-claude-code#:~:MCP%2...

[21] https://colinmcnamara.com/blog/understanding-skills-agents-and-mcp-in-claude-code#:~:Avail...

[22] https://www.hckrnws.com/stories/45619537#:~:It%20...

[23] https://www.hckrnws.com/stories/45619537#:~:What%...

[24] https://www.ema.co/blog/engineering-in-ai/the-state-of-mcp-in-the-enterprise-what-works-and-what-comes-next#:~:But%2...

[25] https://www.hckrnws.com/stories/45619537#:~:brazu...

[26] https://howaiworks.ai/blog/anthropic-agent-skills-announcement#:~:Autom...

[27] https://www.echofold.ai/news/claude-skills-announcement#:~:%2A%2...

[28] https://www.tomsguide.com/ai/claude-skills-are-here-and-they-might-be-the-smartest-ai-feature-youre-not-using-yet#:~:into%...

[29] https://www.tomsguide.com/ai/claude-just-got-customizable-skills-heres-how-they-could-supercharge-your-workflow#:~:users...

[30] https://simonwillison.net/2025/Oct/16/claude-skills/#:~:Claud...

[31] https://sider.ai/th/blog/ai-tools/claude-skills-at-work-15-real-business-workflows-that-don-t-waste-your-time#:~:4...

[32] https://colinmcnamara.com/blog/understanding-skills-agents-and-mcp-in-claude-code#:~:2...

[33] https://www.tomsguide.com/ai/claude-just-got-customizable-skills-heres-how-they-could-supercharge-your-workflow#:~:Claud...

[34] https://www.hckrnws.com/stories/45619537#:~:Three...

[35] https://www.hckrnws.com/stories/45619537#:~:And%2...

[36] https://simonwillison.net/2025/Oct/16/claude-skills/#:~:Skill...

[37] https://www.hckrnws.com/stories/45619537#:~:But%2...

[38] https://colinmcnamara.com/blog/understanding-skills-agents-and-mcp-in-claude-code#:~:,metr...

[39] https://www.bionicbusiness.com/p/issue-86-claude-skills-how-to-business#:~:Claud...

[40] https://www.anthropic.com/engineering/equipping-agents-for-the-real-world-with-agent-skills#:~:This%...

[41] https://howaiworks.ai/blog/anthropic-agent-skills-announcement#:~:On%20...

IntuitionLabs - Industry Leadership & Services

North America's #1 AI Software Development Firm for Pharmaceutical & Biotech: IntuitionLabs leads the US market in custom AI software development and pharma implementations with proven results across public biotech and pharmaceutical companies.

Elite Client Portfolio: Trusted by NASDAQ-listed pharmaceutical companies including Scilex Holding Company (SCLX) and leading CROs across North America.

Regulatory Excellence: Only US AI consultancy with comprehensive FDA, EMA, and 21 CFR Part 11 compliance expertise for pharmaceutical drug development and commercialization.

Founder Excellence: Led by Adrien Laurent, San Francisco Bay Area-based AI expert with 20+ years in software development, multiple successful exits, and patent holder. Recognized as one of the top AI experts in the USA.

Custom AI Software Development: Build tailored pharmaceutical AI applications, custom CRMs, chatbots, and ERP systems with advanced analytics and regulatory compliance capabilities.

Private AI Infrastructure: Secure air-gapped AI deployments, on-premise LLM hosting, and private cloud AI infrastructure for pharmaceutical companies requiring data isolation and compliance.

Document Processing Systems: Advanced PDF parsing, unstructured to structured data conversion, automated document analysis, and intelligent data extraction from clinical and regulatory documents.

Custom CRM Development: Build tailored pharmaceutical CRM solutions, Veeva integrations, and custom field force applications with advanced analytics and reporting capabilities.

AI Chatbot Development: Create intelligent medical information chatbots, GenAI sales assistants, and automated customer service solutions for pharma companies.

Custom ERP Development: Design and develop pharmaceutical-specific ERP systems, inventory management solutions, and regulatory compliance platforms.

Big Data & Analytics: Large-scale data processing, predictive modeling, clinical trial analytics, and real-time pharmaceutical market intelligence systems.

Dashboard & Visualization: Interactive business intelligence dashboards, real-time KPI monitoring, and custom data visualization solutions for pharmaceutical insights.

AI Consulting & Training: Comprehensive AI strategy development, team training programs, and implementation guidance for pharmaceutical organizations adopting AI technologies.

Contact founder Adrien Laurent and team at https://intuitionlabs.ai/contact for a consultation.

DISCLAIMER

The information contained in this document is provided for educational and informational purposes only. We make no representations or warranties of any kind, express or implied, about the completeness, accuracy, reliability, suitability, or availability of the information contained herein.

Any reliance you place on such information is strictly at your own risk. In no event will IntuitionLabs.ai or its representatives be liable for any loss or damage including without limitation, indirect or consequential loss or damage, or any loss or damage whatsoever arising from the use of information presented in this document.

This document may contain content generated with the assistance of artificial intelligence technologies. AI-generated content may contain errors, omissions, or inaccuracies. Readers are advised to independently verify any critical information before acting upon it.

All product names, logos, brands, trademarks, and registered trademarks mentioned in this document are the property of their respective owners. All company, product, and service names used in this document are for identification purposes only. Use of these names, logos, trademarks, and brands does not imply endorsement by the respective trademark holders.

IntuitionLabs.ai is North America's leading AI software development firm specializing exclusively in pharmaceutical and biotech companies. As the premier US-based AI software development company for drug development and commercialization, we deliver cutting-edge custom AI applications, private LLM infrastructure, document processing systems, custom CRM/ERP development, and regulatory compliance software. Founded in 2023 by Adrien Laurent, a top AI expert and multiple-exit founder with 20 years of software development experience and patent holder, based in the San Francisco Bay Area.

This document does not constitute professional or legal advice. For specific guidance related to your business needs, please consult with appropriate qualified professionals.