The Core Strategic Decision

Two Paths to Agentic AI Automation

Option A: The "Smart Consultant" (General Agents)

Example: Claude Code, Goose.

Focus: High-level reasoning, autonomy, and flexibility.

Analogy: Hiring a senior employee who figures out how to solve the problem.

Option B: The "Assembly Line" (Custom Agents)

Example: OpenAI Agents SDK.

Focus: Reliability, process control, and specific workflows.

Analogy: Building a factory machine that performs a specific task perfectly every time.

General Agents (Claude Code, Goose)

The Power of Generalization

What it is: An autonomous agent living in your terminal/environment

Key Features

Zero-Shot Planning: You state the goal; it determines the steps.

Deep Integration: Accesses local files, git history, and command line directly.

Enhanced by MCP: Uses "Model Context Protocol" to plug in external systems (e.g., database access) instantly.

Enhanced by Agent Skill: Modular capabilities (organized folders containing instructions, scripts, and resources)

Best For:

Complex debugging & coding.

Ad-hoc analysis (e.g., "Why are sales down in Q3?").

Tasks requiring human-like judgment.
It is optimized for the 'loop' of thinking. It reads, thinks, acts, and corrects itself.

It is expensive per task but invaluable for non-routine work."

Custom Agents (OpenAI Agents SDK, Claude Agent SDK)

The Power of Specialization

What it is: A framework for building AI workflows

Key Features

Guardrails: Strict control over what the agent can and cannot do.

Orchestration: Define exact hand-offs between multiple agents (e.g., Triage Agent → Support Agent).

UI/UX Flexibility: Can be embedded in web apps, Slack, or internal dashboards.

Enhanced by MCP and Agent Skills: To plug in external systems and modular capabilities (organized folders containing instructions, scripts, and resources)

Best For:

Standard Operating Procedures (SOPs).

High-volume tasks (e.g., processing 5,000 invoices).

Customer-facing interactions (requires strict safety).

"Here, you are the architect. The AI is just a component. You don't want a customer service bot 'getting creative' with your return policy—you want it to follow the script."

Decision Matrix: How to Choose?

Requirement	Choose General Agent (Claude Code, Goose)	Choose Custom Agent (OpenAI SDK, Claude Agent SDK)
Task Type	Novel, Problem-Solving	Repetitive, Standardized
End User	Developers / Technical Staff	Non-Technical / Customers
Error Tolerance	High (Human in the loop)	Low (Must be reliable)
Cost Sensitivity	Low (High value per task)	High (Volume optimization needed)
Implementation	Instant (Install & Run)	Weeks (Design & Build)

The "Trojan Horse" of AI

Why "Claude Code" is Actually a General Agent. Moving Beyond the Name: Code as a Mechanism, Not a Constraint

- **The Misconception:** The name implies it is a tool strictly for software engineers—just a syntax highlighter or autocomplete.
- **The Reality:** It is an **Autonomous Problem Solver** that happens to speak the language of code.
- **The Distinction:**
  - **Coding Agents (e.g., Cursor):** Role-bound to software development workflows.
  - **General Agents (Claude Code):** Solve problems across any domain using code as the universal interface

"Don't let the name fool you. Calling it a 'Coding Agent' is like calling a CEO an 'Email Writer' just because they use email to do their job. Code is simply the tool it uses to exert power over the computer."

Authority Defines the Scope of Action

Where does the Agent live and what problems can it solve?

Feature	Coding Agent (e.g., Cursor)	General Agent (Claude Code)
Scope	Software development	**Any business domain**
Identity	Developer's pair programmer	Digital employee
Habitat	Embedded in developer tooling	Operates across system-level tools
Built For	Developers	**Anyone solving problems**
Example Tasks	"Implement feature, Refactor this module", "Write tests"	"Plan your 2026", "Draft emails", "Why did sales drop?"

Coding agents like Cursor are powerful, software-native agents optimized for development workflows.
General agents like Claude Code are goal-native agents trusted with broader objectives, able to choose tools, cross domains, and act at the system level using code as the universal interface.

The Cognitive Leap

From "Prediction" to "Reasoning"

- **Early Traditional Coding Agents (Predictive-centric):**
    - **Logic:** "Based on the last 10 lines, what is the most likely next line?"
    - **Limit:** They struggle to recover from errors without explicit guidance.
- **General Agents (Reasoning Loop):**
    - **Logic:** Uses an **OODA Loop** (Observe, Orient, Decide, Act).
    - **The Workflow:**
        1. **Observe:** "I see an error in the logs."
        2. **Decide:** "I will check if the Docker container is running."
        3. **Act:** *Executes Docker command.*
        4. **Correct:** "That didn't work, let me try a different flag."

"Early coding agents are primarily prediction-centric. General agents reason through problems. They enter a loop of thinking—acting, checking the result, and self-correcting. This is what allows them to handle complex tasks without you holding their hand."

Code is the Universal Interface

Why Business Questions Get Code Answers

- **The Paradigm Shift:** We don't just write code to build software; we use code to **interrogate reality**.
- **Example: "Why did sales drop in Q3?"**
  - **A Coding Agent:** Would confuse this for a code comment.
  - **A General Agent:**
    1. Writes a **SQL query** to fetch sales data.
    2. Writes a **Python script** to visualize the trend.
    3. Analyzes the chart.
    4. **Answer:** "Sales dropped because of 40% churn in the Enterprise sector."

"This is the core of our strategy. The General Agent acts as a Business Analyst. It translates a human question into a code execution to get a factual answer."

Infinite Extensibility via MCP and Agent Skills

Breaking the "Coder" Stereotype

- **The Enabler: Model Context Protocol (MCP) and Agent Skills**.
- **How it expands the role:**
    - Plug in **Slack MCP and Skill** → Becomes a **Communications Manager**.
    - Plug in **Salesforce MCP and Skill** → Becomes a **RevOps Specialist**.
    - Plug in **Xero/QuickBooks MCP and Skill** → Becomes a **Financial Auditor**.
- **The Verdict:** A strictly defined "Coding Agent" cannot send messages or audit finances. A General Agent can, provided you give it the right "hands" (MCP) and "procedures" (Agent Skill).

"This is why we classify it as 'General.' If you plug in a Finance MCP and Skill, it's a Finance Agent. If you plug in a Sales MCP and Skill, it's a Sales Agent. Code is just the glue that holds it all together."

Skills and MCP Combination

Skills are "Expertise Packs" (instructions + logic) while MCP is the "Data Pipe" (connectivity)

Agent Skills

The "How-To"

These are modular folders (with a SKILL.md) that teach Claude a specific, repeatable workflow (e.g., "Analyze this financial statement according to our company's Q4 risk framework"). It’s about **standardizing expertise**.

MCP

The "With-What"

This is the protocol that connects those skills to your live data (e.g., your SQL database or Jira)

Agent Factory

"Welcome. We are entering the era of the 'Agent Factory.' While the industry talks about General Agents and Custom SDKs as separate trends, we are going to look at how they converge. As you'll see that **Code is the Universal Interface** that allows a General Agent to transform your knowledge into a deployed **Custom Skill or Custom Agent**."