Scaling SWE Agents

Manage Agent Primitives with Agent Package Manager (APM)

Sergio Sisternes

Director, Technology Solutions, EPAM
Microsoft MVP - Developer Technologies
APM contributor

Microsoft® Most Valuable Professional

<epam>

LINKEDIN.COM/IN/SESISPLA | GITHUB.COM/SERGIO-SISTERNES-EPAM

QUICK AUDIENCE PULSE

Before We Dive In — Raise Your Hand If You Have...

Agentic Harnesses

Uses GitHub Copilot > Claude Code > Codex > Cursor in your day-to-day?

Custom Agent, Skills, Instructions...

Written any custom files to shape how your agent behaves on a specific project?

Shared Agent Config

Tried to share or standardise agent configuration across multiple repos, teams, or squads in your organisation?

Copy-Paste Chaos

Ended up copying markdown files between repos because there was no structured, repeatable way to distribute them?

That last card — copy-paste chaos — is exactly why we're here today. If you've ever duplicated Markdowfiles to move your agent from one project to another and thought "there has to be a better way," there is.

THE PROBLEM

Agent Configuration is the New Technical Debt

Today, teams manage prompts mostly by hand — no versioning, no sharing, no governance. And no self-improvement loops.

Inefficient

Building the right prompts is difficult and time-consuming. Engineers are left alone, doing the same work over and over, wasting time and creating fragmented developer experiences across organisations

Inconsistent Approach

Every team configures agents differently. The same learnings and the same mistakes are repeated across the organisation. There is no mechanism for shared knowledge to accumulate.

Zero Reusability

Instructions, skills, and prompts are trapped in individual repos. And copy/pasted from internet. There's no way to version them, share them, or compose them. Every project starts from scratch.

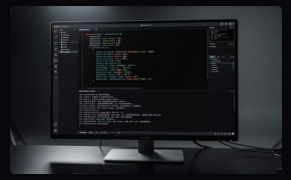
Security Blind Spots

Shared prompt files can carry hidden Unicode characters (Glassworm attacks), prompt injection, and much more. Nobody is auditing agent config today — and that's a risk.

THE INSIGHT

We've Solved This Problem Before

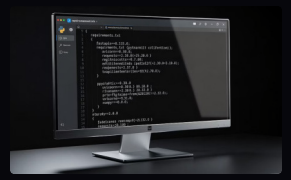
Software teams solved dependency management for application code decades ago. The patterns are proven, the tooling is mature, and developers rely on it every day. Agent configuration deserves exactly the same treatment.



package.json · npm

JavaScript dependencies — declared, resolved, locked, and reproducible across every machine on the team.

```
"dependencies": {
  "react": "^18.2.0",
  "axios": "^1.4.0"
}
```



requirements.txt · pip

Python dependencies — declared, resolved, locked, and reproducible. `pip install -r requirements.txt` and you're done.

```
requests==2.31.0
fastapi==0.103.1
pydantic==2.3.0
```

apm.yml · apm

Agent primitives — declared, resolved, locked, and reproducible. The same pattern, applied to the configuration surfaces of AI coding tools.

```
github/awesome-
copilot/plugins/context-
engineering#v1.0
epam/infrastructure/azure#v0.5
```

"Agent configuration is software. It deserves a dependency manager." — The same patterns that made npm and pip indispensable apply directly to the agent configuration problem.

THE SOLUTION

Introducing Agent Package Manager (APM)

An open-source dependency manager for AI agent configuration. Think `npm install`, but for agent primitives — instructions, skills, prompts, agents, hooks, plugins, and MCP servers. APM provides a comprehensive suite of tools to streamline the development, deployment, and governance of AI coding agents across your organisation.

MIT LICENSED | GITHUB.COM/MICROSOFT/APM | V0.21 | ★ 3K

Dependency Management

Declare, resolve, and manage agent primitive dependencies. Ensures reproducible installations across all environments.

```
apm init
apm install <package>
apm outdated
```

Multi-Tool Compilation

Write primitives once, then compile and deploy to various AI coding tools like Copilot, Claude, Codex, Cursor and more

```
apm compile
apm install -t claude,copilot
```

Content Security

Automated scanning prevents prompt injection and 'Glassworm' attacks by blocking hidden Unicode characters.

```
apm audit
apm audit --ci
```

Plugin Authoring & Distribution

Effortlessly build and distribute plugins for various AI coding tools, leveraging APM's dependency management.

```
apm pack
apm pack --format plugin
```

Marketplaces

Discover, install and author trusted agent primitives from curated marketplaces, with upcoming semver support.

```
apm marketplace add <repo>
apm marketplace browse
apm marketplace publish
```

Runtime & Scripts

Define and execute reusable scripts, preview compiled prompts, and set up AI runtimes directly via APM.

```
apm run <script>
apm preview <script>
```

Enterprise Governance

Implement policy-as-code with `apm-policy.yaml` and enforce rules through CI/CD pipelines.

```
apm-policy.yaml
apm audit --ci
```

Cross-tooling

Technology agnostic solution to pack once, deploy anywhere:

GitHub Copilot · Cursor · Claude · Codex · Gemini · OpenCode · Hermes · OpenClaw · Copilot Cowork ...

Support multiple Primitives

Agents are not only skills. Pack, reuse and deploy. All surfaces managed as first-class citizens:

Instructions · Skills · Prompts · Agents · Hooks · Plugins · MCP Servers

THE SCALE CHALLENGE

The Cost Compounds with Every Team

> "A 5-person team with manual setup has 5 divergent agent configurations. A 50-person team has 50. A 500-person team has chaos."

❌ Without a Package Manager

- Each developer configures agents manually from scratch
- Instructions drift across machines and time
- No way to share or reuse prompts across teams
- MCP servers configured per-developer, inconsistently
- Onboarding requires tribal knowledge transfer
- No audit trail for any agent configuration

✅ With APM

- `apm install` sets up everything in one command
- `apm.lock.yaml` pins exact versions for every machine
- Publish and install from any git host — instantly shareable
- MCP servers declared in manifest, installed consistently everywhere
- Clone → `apm install` → done. That's onboarding.
- Lock file tracks every dependency with full provenance

COMPATIBILITY

One Config. Every Agent. Zero Lock-In.

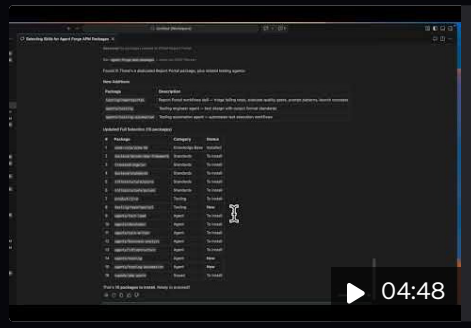
Write your agent configuration once. APM compiles it into the native format of every supported AI coding tool. If you stop using APM tomorrow, every generated file still works — unchanged — with its respective tool.

Harness	Instructions	Agents	Skills	Hooks	Commands /Prompts	Compile	Global Scope
GitHub Copilot	✅	✅	✅	✅	✅	AGENTS.md	🟢 Full
Claude Code	✅	✅	✅	✅	✅	CLAUDE.md	🟢 Full
Codex CLI	N/A	✅	✅	✅	N/A	AGENTS.md	🔴 No
Cursor	✅	✅	✅	✅	N/A	rules/	🟡 Partial
OpenCode	N/A	✅	✅	N/A	✅	AGENTS.md	🟡 Partial
Codex CLI	N/A	✅	✅	✅	N/A	AGENTS.md	🔴 No

Zero lock-in, by design. Every generated file still works natively with its respective tool. APM adds a dependency management layer — not a runtime dependency. You own your output.

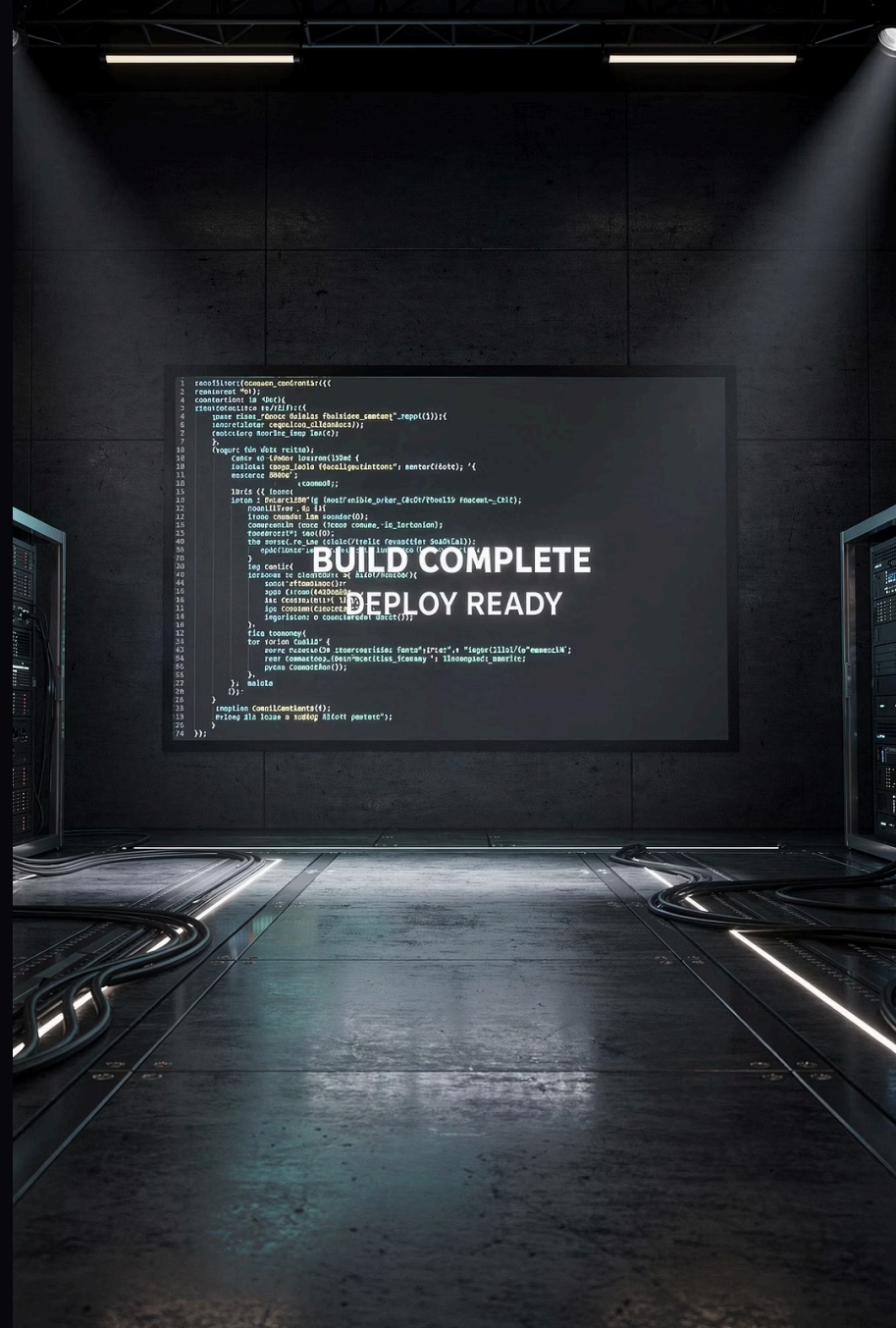
N/A: Feature not available in the AI Tool

Don't tell. Show.



YouTube

Deploy Agent Framework



GET STARTED

Start Managing Agent Primitives Today

One command to install. Three commands to get productive. An open-source community already building packages and primitives for every major use case.

⚡ Install APM

Pick your platform — one command and you're running.

```
# macOS / Linux
curl -sSL https://aka.ms/apm-unix | sh

# Windows (PowerShell)
irm https://aka.ms/apm-windows | iex

# Homebrew
brew install microsoft/apm/apm
```

Try It Now

Initialise a project, install a sample package, and compile to native tool formats in under two minutes.

```
apm init my-project
apm install <organisation>/<package-name>
apm compile
```

Explore the Ecosystem

- Docs: microsoft.github.io/apm
- Code: github.com/microsoft/apm
- Roadmap: github.com/microsoft/apm/discussions/116

Open Standards

Built on AGENTS.md • agentskills.io • MCP (modelcontextprotocol.io) — not proprietary abstractions.

MIT Licensed

Open source from Microsoft. Free to use, fork, and contribute to. No licensing surprises at enterprise scale.

Contribute

Project built for the community. By the community. Submit feature requests, raise bugs, send your PRs.

Questions?

Reach us out on GitHub: **github.com/microsoft/apm**

THE APM PROMISE

The APM promise

Portable by Manifest

One apm.yaml defines your entire agent config. Reproduce it anywhere — any machine, any team, any tool.

Secure by Default

Every install is scanned before it touches your filesystem. No hidden Unicode, no bidi overrides, no prompt injection.

Governed by Policy

Enforce standards org-wide via apm-policy.yaml. Block non-compliant installs in CI/CD before they ship.

Free. Open source. Community-driven. Vendor-neutral.

Join the community in https://www.github.com/microsoft/apm

Key Takeaways

APM: Build by the community. For the community.

1 Agent config is technical debt

Manual, siloed, ungoverned agent configuration compounds with every team added. The cost is real and it scales badly.

2 The pattern is proven

npm, pip, and cargo solved this for code. APM applies the same dependency management principles to agent primitives.

3 Security is non-negotiable

The prompt supply chain has no gap between install and execution. APM scans first — before anything lands in an agent-readable directory.

4 Zero lock-in, open standards

APM outputs the native config format of every major AI coding tool. You own your files. Stop using APM and nothing breaks.

Scan the QR Code to access this deck.