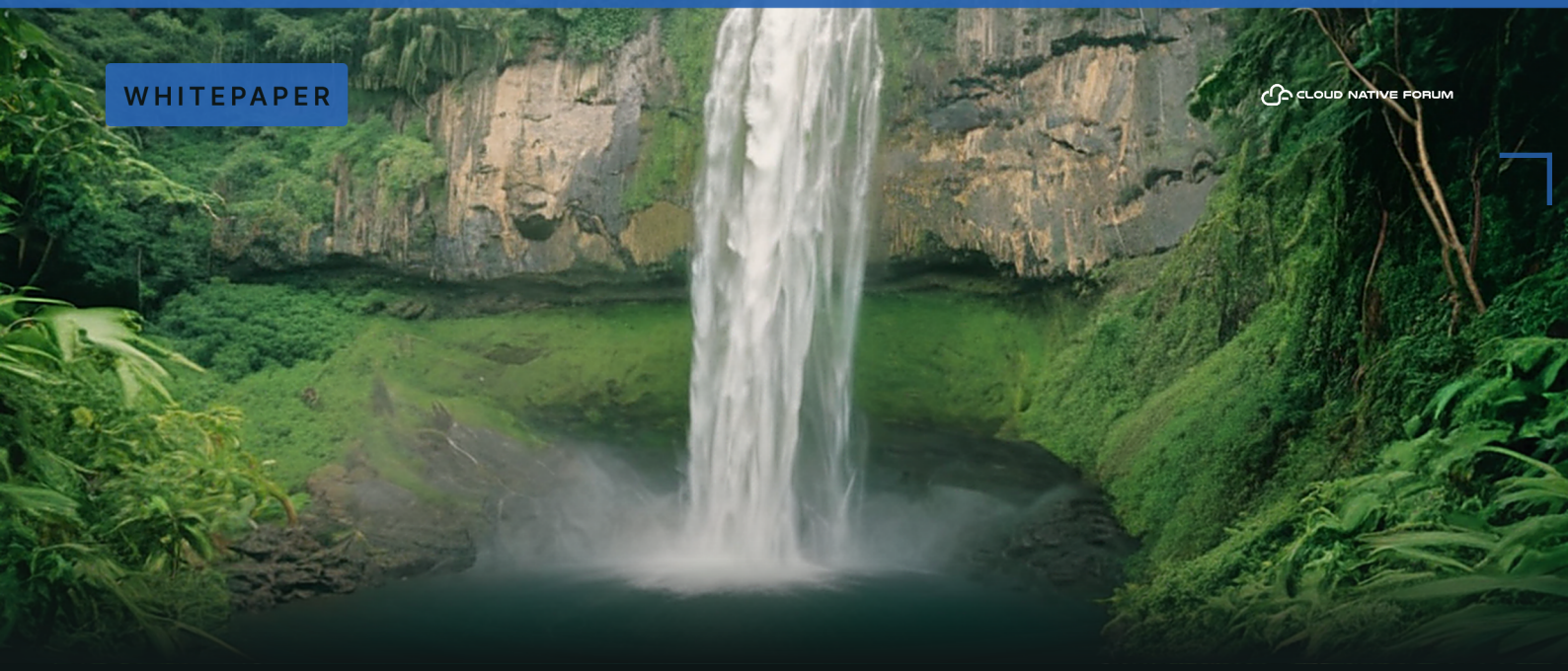


사람과 AI 가 함께 읽고 쓰는 마크다운 시대 — 에이전트 시대의 사내 문서 표준 백서...

2025년부터 2026년 상반기까지 사내 문서 표준 논의는 한 방향으로 좁혀졌습니다. 사람과 AI 에이전트가 동일한 파일을 함께 읽고 수정하는 워크플로우가 빠르게 자리 잡았고, 그 파일 형식은 마크다운으로 수렴했습니다. LlamaIndex, LangChain, Anthropic 의 1차 자료는 모두 같은 방향을 가리킵니다.

2026-06-07 · 작성자: CNCF Korea Team · cncf.co.kr

목차

1장. AI 에이전트 시대 — 문서 표준이 마크다운으로 수렴하는 이유 (2026)

2025년부터 2026년 상반기까지 사내 문서 표준 논의는 한 방향으로 좁혀졌습니다. 사람과 AI 에이전트가 동일한 파일을 함께 읽고 수정하는 워크플로우가 빠르게 자리 잡았고, 그 파일 형식은 마크다운으로 수렴했습니다. LlamaIndex, LangChain, Anthropic 의 1차 자료는 모두 같은 방향을 가리킵니다. LLM 의 컨텍스트 윈도우는 휘발성을 가지므로, 영속 컨텍스트를 담는 매체로 파일시스템이 다시 부상했다는 흐름입니다 [S1][S2][S3].

이 흐름이 IT 의사결정자에게 중요한 이유는 단순합니다. 사내 표준이 어떤 형식이든 작동은 합니다. 그러나 AI 에이전트 도구 한 종을 도입할 때마다 문서 자산을 재가공해야 한다면 운영 비용이 빠르게 누적됩니다. 반대로 파일 자체가 도구 사이의 공통 인터페이스 역할을 하면 도구 전환 비용은 거의 사라집니다. LlamaIndex Jerry Liu 는 "파일 포맷이 곧 API" 라는 표현으로 이 점을 정리했고 [S1], Anthropic 은 SKILL.md 사양을 2025-10-16 에 공개한 뒤 2025-12-18 에 오픈 표준으로 발표했습니다 [S4][S9].

본 장은 이 수렴 현상을 3단계로 설명합니다. 먼저 AI 에이전트가 왜 다시 파일시스템을 찾게 되었는지 컨텍스트 엔지니어링 관점에서 정리합니다. 다음으로 CLAUDE.md, SKILL.md, AGENTS.md 세 종의 마크다운 파일이 어떻게 사실상 표준으로 등장했고 어디까지 채택되었는지 시점·주체·범위 단위로 확정합니다. 마지막으로 공공 의무화 흐름과 분리하여, 민간 자발 채택의 합리적 근거가 무엇인지 정리합니다.

이 장이 다루지 않는 영역도 분명히 합니다. 정책 의무화의 시행 절차, 자가점검표 작성 방법, 변환 도구의 CLI 옵션 비교는 다루지 않습니다. 그 영역은 인접 백서가 이미 다루었으며, 본 백서는 민간 IT 조직이 법적 의무 없이도 마크다운으로 전환할 합리적 근거를 모으는 데 집중합니다. 정량 비교와 ROI 논의는 3장에서 본격적으로 다루며, 본 장은 2025~2026년 시장 흐름과 표준 출현 사실에 한정합니다.

1.1 AI 에이전트의 컨텍스트 엔지니어링과 파일시스템 회귀 (LlamaIndex·LangChain, 2025~2026)

AI 에이전트 설계의 무게중심이 2025년 후반에 다시 이동했습니다. 수백 개 도구를 호출하는 에이전트 설계에서, 파일시스템과 5~10 개 핵심 도구로 단순화하는 설계로 옮겨갔습니다. 출발점은 LlamaIndex Jerry Liu 의 2026-01-15 공개 글 "Files Are All You Need" 와 LangChain Nick Huang 의 2025-11-21 공개 글 "How agents can use filesystems for context engineering" 두 편입니다 [S2][S3]. 두 글은 표면적으로는 엔지니어 대상 기술 정리이지만, 사내 문서 표준을 다시 검토해야 할 의사결정 근거를 함께 제공합니다.

이 절은 세 갈래로 그 근거를 정리합니다. LLM 컨텍스트 윈도우가 왜 영속 매체가 될 수 없는지, 도구 폭증을 어떻게 파일 + CLI 5~10 개로 회귀시켰는지, 마지막으로 grep·glob·read 패턴이 LLM 호출당 토큰 소비를 어떻게 줄이는지 순서로 살펴봅니다.

1.1.1 LLM 컨텍스트 윈도우의 휘발성과 파일시스템의 영속성

LLM 의 컨텍스트 윈도우 는 영속 기억 장치가 아닙니다. 한 번의 추론이 끝나면 그 안의 토큰은 사라집니다. GeekNews 한국어 1차 요약은 이 성질을 "지속적 기억이 아닌 지워지는 화이트보드에 가깝다" 라고 표현했습니다 [S1]. 컨텍스트 윈도우의 크기가 수십만 토큰까지 늘어났더라도, 다음 세션에 그 정보를 그대로 가져올 방법은 윈도우 안에 들어 있지 않습니다. 영속성은 외부 매체가 담당해야 합니다.

이 지점에서 파일시스템이 다시 등장합니다. LangChain Nick Huang 은 파일시스템의 역할을 한 문장으로 정의했습니다. "A filesystem provides a single interface through which an agent can flexibly store, retrieve, and update an infinite amount of context." [S3]. 영속성, 단일 인터페이스, 무한 확장이 세 축입니다. 데이터베이스도 영속성을 제공하지만, 사람이 직접 열어보고 수정하기 어렵습니다. 객체 스토리지도 영속성은 충분하지만, 검색·갱신의 즉시성이 낮습니다. 파일시스템은 사람의 직접 조작과 에이전트의 프로그램적 조작을 동시에 수용하는 매체입니다.

이 차이가 의사결정에 어떤 의미를 가지는지가 핵심입니다. 사내 지식 자산을 데이터베이스에만 두면 사람의 직접 검토가 느려지고, 객체 스토리지에만 두면 빠른 부분 수정이 어렵습니다. 파일시스템 위에 마크다운 파일로 두면, 사람은 텍스트 편집기로 즉시 수정하고 에이전트는 grep·glob·read 로 필요한 부분만 가져옵니다. 매체 선택이 곧 운영 모델 선택입니다 [S1][S3].

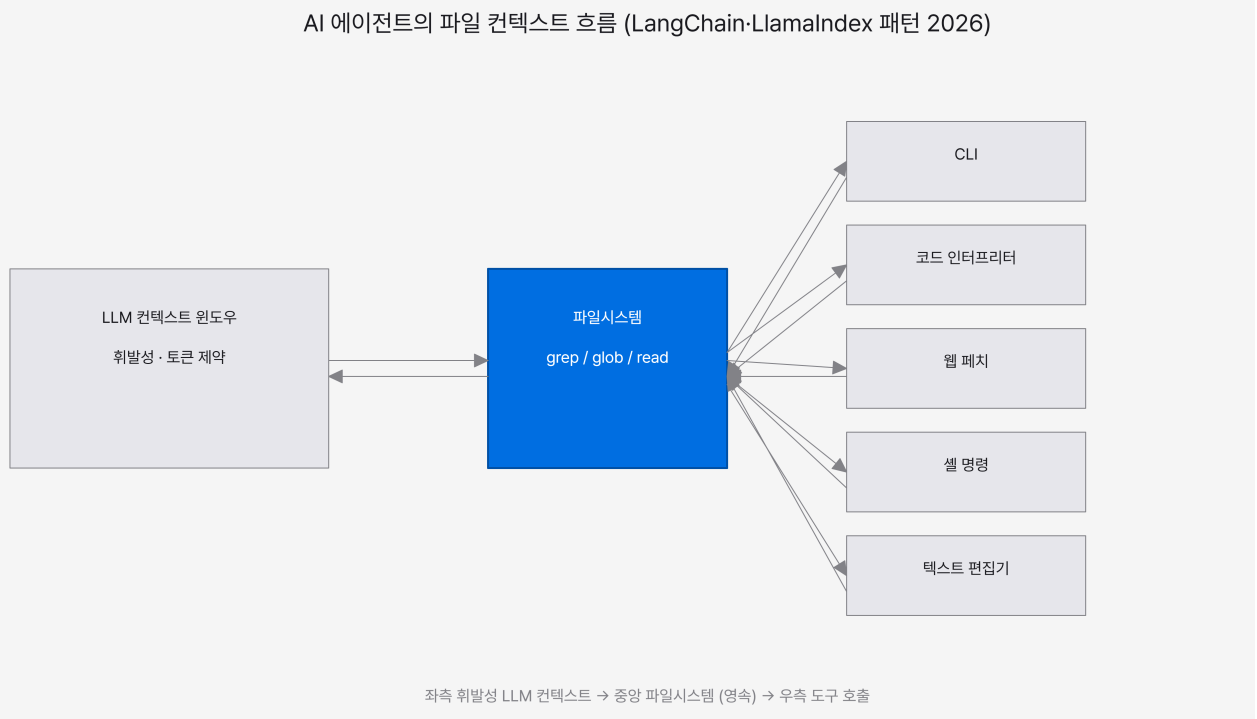


그림. AI 에이전트의 파일 컨텍스트 흐름 (LangChain·LlamaIndex 패턴 2026)

1.1.2 도구 폭증에서 파일 + CLI 5~10 개로의 회귀 — Files Are All You Need

2024 년까지의 에이전트 설계는 도구 개수 늘리기에 가까웠습니다. 검색 도구, 캘린더 도구, 메모 도구, 데이터베이스 도구를 각각 별도 함수로 등록하고 LLM 에게 선택하게 했습니다. 도구 수가 늘수록 호출 정확도는 떨어지고, 디버깅 난이도는 올라갔습니다. LlamaIndex Jerry Liu 는 2026-01-15 공개 글에서 이 흐름이 다른 방향으로 전환되고 있다고 정리했습니다. "Instead of one agent with hundreds of tools, we're evolving towards a world where the agent really only has access to a filesystem and ~5-10 tools: CLI over filesystem, code interpreter, web fetch." [S2].

이 회귀의 핵심은 도구 개수가 줄어도 표현력이 줄지 않는다는 점입니다. CLI 한 종으로 grep, find, sed, awk, jq 가 모두 호출됩니다. 코드 인터프리터 한 종으로 Python, SQL, 정규식이 모두 호출됩니다. 웹 페치 한 종으로 외부 API 가 호출됩니다. 도구를 추가하는 대신, 파일과 5~10 개 핵심 도구의 조합으로 거의 모든 작업을 표현하는 설계입니다 [S2].

이 단순화가 의사결정자에게 의미하는 바는 운영 가시성과 감사 추적성의 향상입니다. 도구 개수가 100 개라면 어떤 호출이 어떤 데이터에 접근했는지 추적하기 어렵습니다. 도구가 5~10 개로 좁혀지고 모든 데이터 접근이 파일시스템을 경유한다면, 감사 로그는 파일 접근 기록 한 곳으로 모입니다. 보안팀이 검토할 표면이 줄고, IT 운영팀이 관찰할 지표가 단순해집니다. Jerry Liu 는 같은 글에서 "Skills might replace MCP . Instead of explicitly giving an agent MCP tools, you could just give it a bunch of files + CLI/code interpreter/web access to do things." 라고 정리하여, 도구 등록 방식 자체가 파일 + CLI 조합으로 단순화될 가능성을 시사했습니다 [S2].

이 단순화의 토대가 마크다운이라는 점이 다음 절의 출발점입니다. 파일 + CLI 조합이 작동하려면 파일 안의 내용이 사람과 LLM 양쪽에게 이해 가능해야 합니다. 바이너리 포맷은 CLI 도구가 직접 읽지 못하고, 복잡한 XML 은 사람이 직접 편집하기 어렵습니다. 마크다운은 양쪽 모두를 만족시키는 최소 형식입니다 [S1][S2].

1.1.3 grep / glob / read 패턴의 토큰 효율 — 10k 토큰 오프로드 사례

도구 결과가 길어질수록 LLM 호출 비용이 누적된다는 사실은 운영팀이 가장 먼저 체감하는 문제입니다. LangChain Nick Huang 은 한 가지 구체 사례로 그 비용 구조를 정리했습니다. "If I put that in my messages history, all 10k tokens are going to sit there for the entire conversation, driving up my Anthropic bill. But if I offload that large tool result to the filesystem, the agent can then intelligently grep search for certain key words, and then read only necessary context into my conversation." [S3]. 10,000 토큰 분량의 도구 결과를 메시지 히스토리에 그대로 두면 대화가 이어지는 동안 매 호출마다 같은 10,000 토큰이 청구됩니다. 같은 결과를 파일로 오프로드하면 grep 으로 필요한 키워드를 찾고, read 로 필요한 줄만 다시 가져옵니다.

이 패턴의 효과는 호출 1 회의 절감이 아니라 호출 누적의 절감입니다. Nick Huang 은 같은 글에서 "Glob and grep allow the agent to not only isolate specific files but also specific lines and specific characters. The read_file tool allows the agent to specify which lines to read in from a file." 라고 추가했습니다 [S3]. 파일 단위·줄 단위·문자 단위로 읽기 범위를 좁힐 수 있으므로, 필요한 정보의 최소 단위만 컨텍스트 윈도우에 올립니다. 그 결과 호출당 입력 토큰이 줄고, LLM API 청구액의 직접 절감으로 이어집니다 [S3].

또한 Nick Huang 은 서브 에이전트 설계에서 파일시스템을 매개로 한 협력 방식을 제안했습니다. "As these subagents do work and learn things, rather than just replying to the main agent with their learnings, they can write their knowledge to the filesystem instead." [S3]. 서브 에이전트가 학습 결과를 메인 에이전트에게 직접 전달하면 응답 토큰이 그대로 호출 히스토리에 쌓입니다. 결과를 파일로 남기면 메인 에이전트는 필요할 때 그 파일을 grep·read 합니다. 협력의 매개체가 메시지에서 파일로 옮겨가는 설계입니다.

운영 규모가 클수록 이 누적 절감의 효과는 커집니다. 일일 LLM 호출이 100 회인 조직과 100 만 회인 조직은 같은 호출당 토큰 절감이라도 절대 절감액이 4 자릿수 차이가 납니다. 호출 누적 곡선의 기울기를 결정하는 변수는 파일 + grep/glob/read 패턴의 채택 여부이며, 그 패턴이 작동하려면 파일이 마크다운처럼 텍스트 기반이어야 합니다 [S1][S2][S3].

1.2 CLAUDE.md · SKILL.md · AGENTS.md — 사실상 표준의 출현과 23 개 이상 도구 채택 (2025~2026)

파일시스템 회귀가 일어났다면, 다음 질문은 그 파일의 이름과 형식이 어떻게 통일되었는가 입니다. 2025년 후반부터 2026년 상반기 사이에 세 종의 마크다운 파일이 사실상 표준 자리를 차지했습니다. CLAUDE.md, SKILL.md, AGENTS.md 세 종입니다. 이 절은 그 세 종이 언제, 누구에 의해, 어떤 범위로 표준화되었는지 시점·주체·범위 단위로 확정합니다.

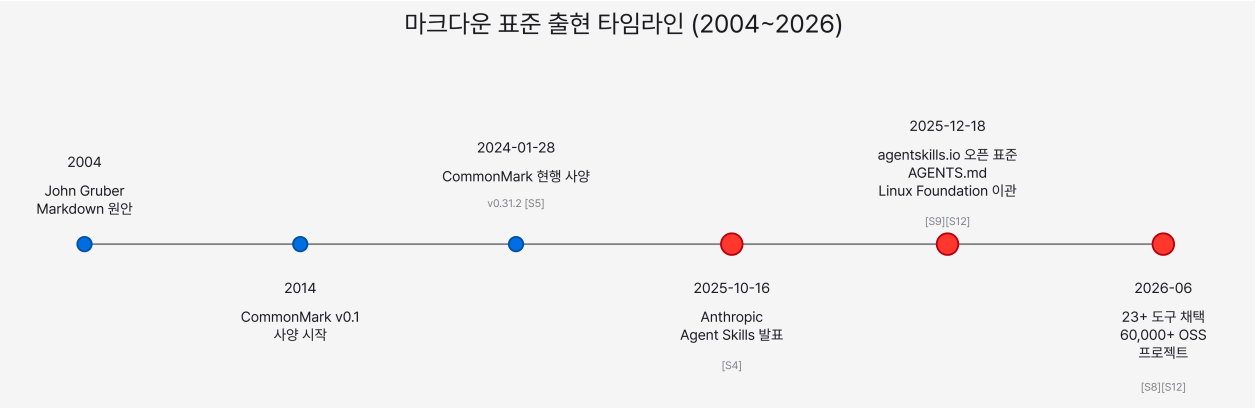


그림. 마크다운 표준 출현 타임라인 (2004~2026)

1.2.1 Anthropic Agent Skills 사양과 progressive disclosure 설계

Anthropic 은 2025-10-16 공개한 Engineering blog "Equipping agents for the real world with Agent Skills" 에서 Agent Skills 사양을 처음 정리했습니다. 정의는 단순합니다. "A skill is a directory containing a SKILL.md file that contains organized folders of instructions, scripts, and resources that give agents additional capabilities." [S4]. SKILL.md 한 개를 진입점으로 두고, 같은 디렉터리에 보조 스크립트와 리소스를 배치하는 구조입니다. 진입점이 마크다운이라는 점이 핵심입니다. 사람이 직접 읽고 수정 가능하며, LLM 도 그대로 파싱합니다.

이 사양의 설계 원칙으로 Anthropic 은 progressive disclosure 를 제시했습니다. "Progressive disclosure is the core design principle that makes Agent Skills flexible and scalable. Like a well-organized manual that starts with a table of contents, then specific chapters, and finally a detailed appendix, skills let Claude load information only as needed." [S4]. SKILL.md 의 첫 부분에는 메타데이터와 요약만 두고, 본문 깊이로 들어갈수록 상세 절차를 배치합니다. LLM 은 모든 절차를 한 번에 컨텍스트 윈도우에 올리지 않습니다. 필요한 시점에 필요한 절만 read 로 가져옵니다.

이 설계가 토큰 효율과 확장성의 직접 근거입니다. 진입점 마크다운만 처음 로드되고, 본격 작업이 시작될 때 보조 파일이 추가로 로드되는 흐름은 1.1.3 절의 grep/read 패턴과 정확히 일치합니다. 또한 Anthropic 은 같은 글에서 채택 범위를 명시했습니다. "Agent Skills are supported today across Claude.ai, Claude Code, the Claude Agent SDK, and the Claude Developer Platform." [S4]. 자사 4 개 제품 라인업이 동일 사양을 인식한다는 사실은 사내 도입 검토 시 단일 사양으로 시작할 수 있다는 신호입니다.

발표 시점과 오픈 표준 공개 시점은 분리해서 보아야 합니다. 사양은 2025-10-16 에 공개되었고, 오픈 표준은 2025-12-18 에 별도로 발표되어 agentskills.io 에서 사양과 SDK 가 공개되었습니다 [S9]. 자사 제품 사양에서 외부 채택 가능 표준으로 전환되는 데 약 2 개월이 걸렸고, 이후 외부 도구의 채택이 빠르게 이어졌습니다.

1.2.2 AGENTS.md 의 다중 도구 호환과 Linux Foundation AAIF 이관

AGENTS.md 는 다른 경로로 표준화되었습니다. 공식 사이트 agents.md 는 이 파일의 역할을 짧게 정의합니다. "Think of AGENTS.md as a README for agents: a dedicated, predictable place to provide the context and instructions." [S8]. 리포지토리 루트에 두는 README 와 유사한 위치·역할이며, 다만 독자가 사람이 아니라 에이전트라는 점이 차이입니다. 리포지토리의 구조, 빌드·테스트 방법, 코딩 규약, 의사소통 채널을 한 파일로 모아둡니다.

거버넌스 측면이 의사결정자에게 더 중요합니다. AGENTS.md 는 처음 OpenAI, Cursor, Jules, Factory, Amp 컨소시엄으로 시작했으나, 2025-12 시점에 Linux Foundation 산하 Agentic AI Foundation (AAIF) 으로 거버넌스가 이관되었습니다. 회원사로 OpenAI, Anthropic, Google, AWS, Bloomberg, Cloudflare 가 참여했습니다 [S8][S12]. 단일 벤더의 사내 사양이 아니라 중립 재단의 공통 표준이라는 점이 채택 결정 시 핵심 안전장치입니다.

이 거버넌스 이관 사실이 vendor-neutral 원칙의 직접 근거입니다. 사내 표준을 단일 벤더 사양에 묶으면 그 벤더의 제품 단종 시 표준 자체가 재검토 대상이 됩니다. 중립 재단의 표준은 그 위험을 분산합니다. 같은 시점에 The New Stack 은 Agent Skills 의 오픈 표준 공개를 다음과 같이 정리했습니다. "Anthropic published Agent Skills as an open standard on December 18 [2025], releasing the specification and SDK at agentskills.io for any AI platform to adopt." [S9]. 두 표준 모두 2025-12 에 단일 벤더 사양에서 오픈 표준으로 전환되는 흐름이 동시에 일어났습니다.

거버넌스 이관 사실 한 가지만으로 단일 벤더 종속 위험이 완전히 사라지지는 않습니다. 다만 위험의 크기는 분명히 줄어듭니다. 표준이 중립 재단에 있으므로, 한 벤더의 도구 전환이 표준 자체의 폐기로 이어지지 않습니다 [S8][S9][S12].

1.2.3 23 개 이상 도구 + 60,000+ 오픈소스 프로젝트의 채택 폭

표준의 채택 폭은 의사결정 시점을 좌우합니다. 2026-06-07 접속 기준으로 AGENTS.md 는 23 개 이상의 도구에서 공식 지원됩니다. OpenAI Codex, Google Jules, Cursor, Aider, Devin, Zed, Warp, VS Code, JetBrains Junie, Amp, RooCode, Gemini CLI, GitHub Copilot 이 대표 사례입니다 [S8]. agents.md 공식 사이트는 이 호환성을 한 줄로 정리합니다. "Your agent definitions are compatible with a growing ecosystem of AI coding agents and tools." [S8]. 한 번 작성한 AGENTS.md 가 도구를 바꿔도 그대로 작동한다는 의미입니다.

채택 폭의 또 다른 지표는 GitHub 검색 기준 60,000+ 오픈소스 프로젝트가 AGENTS.md 를 두고 있다는 점입니다 [S8][S12]. 임계치를 넘긴 채택 폭은 두 가지 의미를 가집니다. 첫째, 신규 도구가 등장할 때 그 도구 또한 AGENTS.md 를 인식하도록 설계될 가능성이 높습니다. 둘째, 채택하지 않는 비용이 채택 비용을 추월하는 시점이 임박합니다. 표준 외부에 머무를수록 신규 도구와의 통합 비용이 누적되기 때문입니다.

AGENTS.md / SKILL.md / CLAUDE.md 호환 매트릭스 (2026-06)

도구	AGENTS.md	SKILL.md	CLAUDE.md	RAG 색인	AAIF 회원
Claude Code	✓	✓	✓ (네이티브)	✓	✓
Codex (OpenAI)	✓	△	△	✓	✓
Copilot (GitHub)	✓	△	△	✓	—
Cursor	✓	✓	✓	✓	—
Gemini CLI	✓	△	△	✓	✓
Jules (Google)	✓	△	△	✓	✓
Aider	✓	△	✓	✓	—
Devin	✓	△	△	✓	—

✓ 공식 지원 △ 부분 지원 — 미지원/미확정

23개 이상 도구 + 60,000+ 오픈소스 프로젝트, 2026-06-07 기준 [S8][S12]

그림. AGENTS.md / SKILL.md / CLAUDE.md 호환 매트릭스 (2026-06 시점)

이 채택 폭에서 의사결정 함의는 분명합니다. 사내 문서 표준 평가 시 호환 도구 수를 정량 지표로 사용 가능하다는 점입니다. 평가 항목에 "현재 채택 도구 수 ≥ 20" 같은 기준을 추가하면, 단일 벤더 사양과 오픈 표준을 객관적으로 비교할 수 있습니다 [S8][S12].

1.3 공공 정책과 민간 트렌드의 분리 — 본 백서의 위치 (DP1, 2026)

이 절은 공공 정책의 흐름과 민간 자발 채택의 흐름을 분리합니다. 두 흐름은 같은 시기에 같은 형식 (마크다운) 으로 수렴했지만, 도입 동기와 평가 기준이 다릅니다. 분리해서 다루지 않으면 의사결정 근거가 흐려집니다.

1.3.1 행정안전부 2026-05-18 의무화의 맥락 인용 — 정의의 민간 재해석

공공 부문은 2026-05-18 부터 행정안전부 「행정업무 운영 및 혁신에 관한 규정」 개정에 따라 기계판독 가능한 형태의 문서 생산을 의무화했습니다. 정의 원문은 다음과 같습니다. "기술 표준과 규격이 공개되어 인공지능(AI)과 사람이 모두 쉽게 읽고 활용할 수 있는 기계판독이 가능(Machine Readable)한 형태" [S6]. 본 백서는 이 정의의 워딩을 1 회만 맥락 인용으로 가져옵니다. 시행 절차, 자가점검표 작성 방법, 부처별 적용 일정은 다루지 않으며, 그 영역은 인접 백서 (2026-05-15 공공 마크다운 전환 백서) 가 별도로 정리했습니다.

이 정의를 민간 자발 채택의 평가 기준으로 재해석하면 핵심 표현 두 가지가 남습니다. "기술 표준과 규격이 공개" 라는 표현은 vendor-neutral 원칙의 다른 이름입니다. "AI 와 사람이 모두 쉽게 읽고 활용" 이라는 표현은 이중 가독성 원칙의 다른 이름입니다. 두 원칙은 공공 의무화와 무관하게 민간 IT 조직의 사내 표준 평가 기준으로 그대로 적용됩니다.

1.3.2 본 백서의 입장 — 민간 트렌드 기반의 자발적 채택 논리

본 백서는 법적 의무가 없는 민간 IT 조직이 자발적으로 마크다운을 채택할 합리적 근거를 다룹니다. 출발점은 LlamaIndex Jerry Liu 의 "Files Are All You Need" (2026-01-15) [S2] 와 LangChain Nick Huang 의

"How agents can use filesystems for context engineering" (2025-11-21) [S3] 두 편이며, 보강 근거는 Anthropic Agent Skills 사양 [S4] 과 AGENTS.md 의 Linux Foundation AAIF 이관 사실 [S8][S9] [S12] 입니다. 공공 의무화는 외부 강제력에 따른 채택을 다루지만, 민간 채택은 자체 ROI 평가에 따른 채택을 다룹니다. 평가 축은 토큰 효율, 도구 호환 폭, 거버넌스 중립성, 운영 가시성 네 가지이며 이후 장에서 차례로 다룹니다.

GeekNews 한국어 1차 요약이 정리한 한 문장이 이 백서의 입장을 잘 표현합니다. "파일 포맷이 곧 API 이며 Claude Code 용으로 작성한 스킬이 Copilot 에서도 작동" 한다는 표현입니다 [S1]. 사내 문서 자산이 곧 AI 도구의 입력 인터페이스라는 관점에서, 형식을 마크다운으로 통일하는 결정은 도구 선택과 독립적인 인프라 결정에 해당합니다. 이 결정의 비용·편익을 정량으로 다루는 작업은 2 장 이중 가독성 아키텍처 분석과 3 장 ROI 평가에서 이어집니다.

2장. 이중 가독성 아키텍처 — 사람과 AI 가 같은 파일을 읽는 CommonMark·frontmatter 설계 (2026)

사내 문서를 사람과 AI 가 동시에 읽고 고쳐 쓰는 환경에서는 한 가지 단순한 질문이 가장 먼저 떠오릅니다. 같은 파일을 두 종류의 독자가 동시에 읽으려면 어떤 조건이 충족되어야 하는가 입니다. 1장에서 살펴본 외부 동력 (Anthropic Agent Skills · AGENTS.md · CNCF AI-Native 선언) 은 이 질문을 *시장 환경* 으로 만들었습니다. 2장은 이 질문을 *파일 설계* 의 문제로 좁힙니다.

본 장의 핵심 메시지는 한 줄로 정리할 수 있습니다. CommonMark v0.31.2 사양과 YAML frontmatter 의 분리 설계, Git diff 친화 구조, 그리고 SKILL.md · CLAUDE.md · AGENTS.md · MEMORY.md · CONTEXT.md 다섯 파일의 계층 모델이 사람과 AI 의 동시 읽기·쓰기를 운영 가능한 구조로 만들었습니다. 사양·구조·계층 세 층위가 함께 맞물려야 비로소 *이중 가독성* 이 단순한 구호가 아니라 사내 정책 문서로 박힐 수 있습니다.

본 장은 세 단계로 진행합니다. 2.1 절은 사람 가독성과 기계 가독성이 동시에 충족되어야 한다는 AND 조건을 CommonMark 사양 사실과 frontmatter 분리 설계로 정리합니다. 2.2 절은 텍스트 기반 diff 가 결재 추적과 어떻게 정합하는지, 그리고 마크다운이 HWP·DOCX·PDF·JSON·YAML·XML 일곱 포맷 가운데 운영 효율 측면에서 어떤 자리를 차지하는지 비교 매트릭스로 보입니다. 2.3 절은 다섯 파일의 권한·우선순위·로딩 시점을 한 장의 계층 모델로 묶어 사내 운영 적용점까지 끌어옵니다 [S5] [S4] [S8] [S3].

독자가 이 장을 다 읽고 나면 사내 마크다운 표준 설계 회의에서 *어떤 frontmatter 키를 표준으로 둘 것인가 · 결재 시스템과 Git 메타데이터를 어떻게 1대 1로 매핑할 것인가 · 다섯 파일 가운데 어느 것을 우선순위 최상위에 둘 것인가* 라는 세 가지 결정 사항에 대한 1차 참조 모델을 손에 쥐게 됩니다.

2.1 사람 가독성과 기계 가독성의 AND 조건 — CommonMark v0.31.2 사양의 역할 (S5, 2024)

이중 가독성 (dual readability) 은 두 가지 가독성이 *동시에* 충족되어야 한다는 AND 조건입니다 . 사람만 읽을 수 있고 기계가 파싱하지 못하면 RAG 인덱싱이 무너지고, 기계만 읽을 수 있고 사람이 읽지 못하면 결재·감수의 인간 책임 사슬이 끊어집니다. 이 절은 AND 조건이 어떤 사양과 설계 위에서 성립하는지를 정리합니다.

CommonMark 사양의 채택 사실, YAML frontmatter 와 본문의 분리 설계, 그리고 파일 포맷이 곧 API 가 된다는 운영 의미가 차례로 이어집니다. 세 항목은 각각 *표준의 안정성 · 데이터 모델의 분리 · 상호운용성의 운영 비용* 이라는 서로 다른 질문에 답합니다.

2.1.1 CommonMark v0.31.2 사양 채택 사실과 라이선스 조건 (S5, 2024-01-28)

본 백서는 사내 마크다운 표준의 기준 사양으로 CommonMark v0.31.2 를 채택하는 것을 권고합니다. 사양 현행 버전은 v0.31.2 이며 게시일은 2024년 1월 28일입니다. 사양 관리자는 John MacFarlane 이고 라이선스는 CC BY-SA 4.0 입니다 [S5] .

운영 의미는 세 가지입니다. 첫째, 사양 관리자가 1인 학자라는 점이 *의외로* 안정성의 근거가 됩니다. 사양이 컨소시엄 정치 일정에 휘둘리지 않고 게시 주기가 길게 유지되어 왔습니다. 둘째, CC BY-SA 4.0 은 사내 정책 문서에 사양을 인용·재배포하는 데 법무 검토 부담을 사실상 0으로 만듭니다. 셋째, 본 백서는 CommonMark 사양 자체를 해설하지 않습니다. 채택 사실과 라이선스 조건의 명시만으로 사내 표준 채택의 1차 의사결정 근거는 충분합니다 [S5].

2.1.2 YAML frontmatter 와 본문의 분리 설계 — 메타데이터·콘텐츠 책임 분리 (S4 인용)

이중 가독성을 운영 가능한 구조로 만드는 첫 번째 설계가 frontmatter 와 본문의 분리입니다 . Anthropic Agent Skills 의 SKILL.md 사양은 디렉터리 구조에 SKILL.md 파일을 두고, 그 안에 지침·스크립트·자원을 조직된 폴더로 묶도록 정의합니다 [S4]. 이 정의에서 frontmatter 는 제목·작성자·버전·태그·조건부 로딩 키 같은 *메타데이터* 를 담고, 본문은 *콘텐츠 그 자체* 를 담는 책임 분리가 자연스럽게 도출됩니다.

분리 설계가 의사결정 사안인 이유는 두 가지입니다. 첫째, frontmatter 키 선택은 사내 RAG 인덱싱·검색 정확도의 1차 변수가 됩니다. 어떤 키를 표준으로 둘지 (예: `version` , `audience` , `expires_at` , `confidentiality`) 가 결정되어야 검색 엔진과 에이전트가 동일한 메타 모델 위에서 동작합니다. 둘째, 분리 설계는 Anthropic 의 progressive disclosure 원칙을 운영에서 구현하는 기반입니다. 사양 원문은 "skills let Claude load information only as needed" 라고 말합니다 [S4]. 메타데이터 헤더만 먼저 읽고 본문은 필요할 때만 펼치는 동작은 frontmatter 와 본문의 분리가 전제되어야 가능합니다.

이중 가독성 4요소 도식 (다음 절 끝의 도식) 은 AND 조건을 네 요소로 분해합니다. 공개 표준 (CommonMark v0.31.2) · 사람 가독성 (합쇼체 본문) · AI 가독성 (frontmatter 메타) · 기계판독 (CC BY-SA 4.0 라이선스 + 일관된 파서) 네 축이 동시에 충족될 때 비로소 *이중 가독성* 이 사내 운영 어휘로 박힙니다 [S5] [S4].

2.1.3 파일 포맷이 곧 API — 조율 없는 상호운용성의 운영 의미 (S1, GeekNews 토픽 27320)

이중 가독성의 두 번째 운영 의미는 *파일 포맷 자체가 API 의 역할* 을 한다는 명제입니다 . GeekNews 한국어 1차 요약은 "조율 없는 상호운용성: 두 앱이 마크다운을 읽을 수 있으면 컨텍스트 공유가 가능" 하다고 정리합니다 [S1].

운영 관점에서 이 명제의 의미는 분명합니다. API 표준화는 통상 컨소시엄 협의·버전 관리·인증 절차·SDK 배포라는 비용이 따릅니다. 마크다운은 이 비용을 *파일 사양만으로* 우회합니다. 한 도구에서 작성한 SKILL.md 가 다른 도구에서도 동일하게 읽힌다는 사실은 사내 도구 선택의 자유도를 높입니다. 도구를 바꾸어도 문서 자산은 그대로 살아남습니다.

물론 *조율 없는 상호운용성* 이 *무비용 상호운용성* 을 의미하지는 않습니다. frontmatter 키 명명 규칙, 본문의 헤더 깊이 정책, 인용·각주의 표기 규약 같은 사내 합의는 여전히 필요합니다. 다만 그 합의가 *외부 컨소시엄의 표준 채택 일정* 에 종속되지 않는다는 점이 ROI 의 정성 근거가 됩니다. 정량 수치는 3장에서 다룹니다 [S1].

2.2 Git diff 친화 구조와 포맷 비교 매트릭스 — HWP·DOCX·PDF·JSON·YAML·XML 대 마크다운

이중 가독성의 두 번째 축은 *변경 이력의 가독성* 입니다. 사람과 AI 가 같은 파일을 동시에 고쳐 쓰는 환경에서는 누가·언제·무엇을 바꾸었는지가 사람의 눈으로도 기계의 파서로도 동일하게 보여야 합니다. 텍스트 기반 diff 가 이 요구를 1대 1로 충족합니다.

이 절은 두 단계로 진행합니다. 2.2.1 절은 텍스트 diff 와 결재·감사 추적의 정합을 코드 리뷰 패러다임으로 정리합니다. 2.2.2 절은 마크다운과 HWP·PDF·DOCX·JSON·YAML·XML 의 운영 효율을 일곱 평가 축에서 비교한 매트릭스를 제시합니다.

2.2.1 텍스트 기반 diff 와 결재 추적의 정합 — 코드 리뷰 패러다임 적용

Git diff 는 변경 사항을 줄 단위 추가·삭제로 보여 줍니다. 마크다운은 텍스트 파일이기 때문에 모든 변경이 diff 위에서 그대로 드러납니다. 한 단어 수정도, 한 절의 추가도, 한 헤더의 삭제도 동일한 표기 체계로 나타납니다. 결재 시스템이 요구하는 *작성자 · 변경 시점 · 승인자 · 변경 범위* 네 가지 메타데이터는 Git commit 의 `author` · `date` · `signed-off-by` · diff hunk 와 1대 1로 매핑됩니다 .

LangChain 의 Nick Huang 은 서브에이전트가 학습한 내용을 메인 에이전트로 회신하는 대신 파일시스템에 기록하는 패턴을 제시합니다 [S3]. 이 패턴이 결재 추적과 정합하는 이유는 *기록의 영속성* 때문입니다. 메시지 히스토리에 머무는 학습 결과는 세션 종료와 함께 사라지지만, 파일에 기록된 변경은 Git 메타데이터와 함께 남습니다. 사내 감사 추적이 요구하는 *변경의 비휘발성* 이 자연스럽게 충족됩니다.

코드 리뷰 패러다임은 여기에 한 가지 운영 자산을 더합니다. Pull Request 위에서 진행되는 라인별 의견·승인·반려는 사내 결재 워크플로우와 같은 의미 단위 (semantic unit) 로 동작합니다 . 결재 시스템을 새로 구축할 필요 없이, Git 운영을 그대로 결재 운영으로 *겹쳐 놓을* 수 있습니다. 본 절의 비교 매트릭스 표 일부 (Git diff 친화성 행) 는 이 정합의 정량 표현입니다 [S3].

2.2.2 포맷 비교 매트릭스 — 마크다운 vs HWP·PDF·DOCX·JSON·YAML·XML (D3)

사내 문서 포맷의 선택 의사결정은 통상 *어떤 도구가 익숙한가* 라는 질문에서 시작합니다. 본 백서는 이 질문을 일곱 평가 축으로 확장하기를 권고합니다. 가독성 · 토큰 효율 · 변환 비용 · 외부 협업 · Git diff 친화성 · 라이선스 · 상호운용성입니다. 일곱 축은 사람과 AI 가 동시에 같은 파일을 다루는 환경에서 *측정 가능한 운영 변수* 들입니다.

아래 그림 (포맷 비교 매트릭스) 은 일곱 평가 축을 행으로, 일곱 후보 포맷 (마크다운 · HWP · DOCX · PDF · JSON · YAML · XML) 을 열로 두고 셀에 정성 등급과 정량 수치를 함께 배치한 매트릭스입니다. 다음 표는 그 매트릭스의 본문 요약입니다.

평가 축	마크다운	HWP	DOCX	PDF	JSON	YAML	XML
사람 가독성	높음 (평문)	높음 (전용 뷰어)	높음 (전용 뷰어)	높음 (인쇄)	낮음 (구조 노출)	중간	낮음 (태그 잡음)
토큰 효율 (LLM 입력)	최고 (기준)	측정값 없음	측정값 없음	변환 후 평가	JSON 대비 -34~-38% (마크다운이 더 적음) [S10]	YAML 대비 -10~-12% (마크다운이 더 적음) [S10]	동일 정보 대비 +80% 토큰 [S10]
변환 비용	0 (텍스트 그대로)	높음 (전용 변환기 필요)	중간 (LibreOffice 등)	높음 (OCR·레이아웃 손실)	0	0	0
외부 협업 (GitHub/Notion/Obsidian)	광범위	국내 한정	광범위	광범위 (읽기 전용)	광범위 (데이터)	광범위 (설정)	광범위 (데이터)
Git diff 친화성	최고	낮음 (바이너리)	낮음 (바이너리 ZIP)	낮음 (바이너리)	높음	높음	높음
라이선스	CC BY-SA 4.0 (CommonMark 사양) [S5]	한컴 사양	ECMA-376	ISO 32000	RFC 8259	YAML 1.2	W3C
상호운용성	파일 포맷이곧 API [S1]	한컴 생태계 의존	Office 생태계	뷰어 광범위	API 표준 필요	설정 표준 필요	스키마 정의 필요

매트릭스 해석의 요점은 세 가지입니다. 첫째, 토큰 효율 행에서 improvingagents.com 의 1,000 문항 벤치마크 (2025-10-14 게시) 는 GPT-5 Nano · Gemini 2.5 Flash Lite 두 모델 양쪽에서 마크다운이 JSON 대비 34~38%, YAML 대비 10~12% 적은 토큰으로 동일 정보를 표현했다고 보고합니다 [S10]. 같은 데이터에서 XML 은 마크다운 대비 80% 더 많은 토큰을 소비했으며, 이는 추론 비용 거의 2배에 해당합니다 [S10]. 둘째, 이 벤치마크는 영문 Terraform 유사 합성 데이터 기준이라는 점을 본 백서는 박아 둡니다. 한국어 환경에서의 동일 비율 재현은 본 조사 범위 밖이며, 적용 시에는 절감 비율의 하한값을 사용하기를 권고합니다 (정량 분석은 3장에서 다룹니다). 셋째, Git diff 친화성 행에서 마크다운 · JSON · YAML · XML 네 텍스트 포맷이 모두 우위에 있지만, 사람 가독성 행을 함께 보면 마크다운이 *AND 조건* 을 충족하는 유일한 후보가 됩니다.

매트릭스가 의사결정에 주는 메시지는 단순합니다. 일곱 평가 축 가운데 다섯 축 이상에서 우위를 점하는 포맷은 마크다운 하나입니다. 도구 친숙도 한 가지 축에서만 HWP·DOCX 가 국내 한정 우위를 가집니다. 사내 표준 채택의 의사결정은 일곱 축 가운데 사내 운영에 가장 큰 비중을 두는 두세 축을 가중치로 정해 결정하는 것을 권고합니다 [S10] [S5] [S1].

2.3 SKILL.md · CLAUDE.md · AGENTS.md · MEMORY.md · CONTEXT.md — 5개 파일 계층 모델 (D7, 2026)

사내 마크다운 표준의 마지막 설계 결정은 *어떤 파일이 어떤 책임을 지는가* 입니다. 2025년 하반기 ~ 2026년 상반기 사이에 사실상 표준으로 굳어진 파일이 다섯입니다. SKILL.md · CLAUDE.md · AGENTS.md · MEMORY.md · CONTEXT.md 입니다. 이 절은 다섯 파일의 권한 · 우선순위 · 로딩 시점 · 책임 분리를 단일 계층 모델로 정리합니다.

2.3.1 항은 다섯 파일의 권한·우선순위·로딩 시점을 progressive disclosure 원칙으로 풀고, 2.3.2 항은 사내 운영 적용점을 LangChain 서브에이전트 메모리 사례로 보입니다.

2.3.1 5개 파일의 권한·우선순위·로딩 시점 — progressive disclosure 적용 (S4·S8)

다섯 파일의 역할은 서로 다릅니다. SKILL.md 는 *능력 정의* 파일로, Anthropic 사양에 따르면 "지침·스크립트·자원" 을 묶어 에이전트에게 추가 능력을 부여하는 디렉터리 단위입니다 [S4]. CLAUDE.md 는 *프로젝트 컨텍스트* 파일로, Claude Code 가 세션 진입 시 기본으로 읽는 사내 컨벤션·경로·금기 규약을 담습니다. AGENTS.md 는 *도구 호환 README* 로, agents.md 사양은 "Think of AGENTS.md as a README for agents: a dedicated, predictable place to provide the context and instructions" 라고 정의합니다 [S8]. MEMORY.md 는 *장기 메모리* 파일로, 세션을 넘어 유지되는 사용자·프로젝트 사실의 인덱스를 담습니다. CONTEXT.md 는 *휘발성 컨텍스트* 파일로, 현재 작업 단위에 한정된 임시 메모를 담습니다 .

권한·우선순위 결정은 사내 정책 수립의 1차 의사결정 사항입니다. 본 백서가 권고하는 우선순위는 SKILL.md > CLAUDE.md > AGENTS.md > MEMORY.md > CONTEXT.md 입니다. 근거는 두 가지입니다. 첫째, SKILL.md 는 *능력 자체* 를 정의하므로 그 정의가 다른 파일의 규약을 압도해야 의도된 동작을 보장할 수 있습니다. 둘째, CLAUDE.md 는 *프로젝트 단위* 규약이므로 *도구 단위* README 인 AGENTS.md 보다 좁은 범위에 정확하게 적용됩니다. MEMORY.md·CONTEXT.md 는 사실 인덱스·임시 메모로 분류되어 규약 우선순위에서는 가장 아래입니다.

로딩 시점은 progressive disclosure 원칙을 따릅니다. 사양 원문은 매뉴얼의 목차→장→부록 비유를 듭니다 [S4]. 다섯 파일은 *목차→규약→능력→사실→메모* 순서로 펼쳐집니다. AGENTS.md 가 목차·README 역할로 먼저 진입하고, CLAUDE.md 가 프로젝트 규약을 펼치고, SKILL.md 가 호출 시점에 능력을 펼치고, MEMORY.md 가 필요한 사실을 인용하고, CONTEXT.md 가 현재 사이클의 임시 메모를 더합니다.

아래 그림 (5개 파일 계층 모델) 은 이 우선순위와 로딩 시점을 다섯 계층 피라미드로 시각화한 다이어그램입니다. 피라미드 최상단의 SKILL.md 부터 최하단의 CONTEXT.md 까지, 각 층에 해당 파일이 답하는 질문 (이 에이전트는 무엇을 할 수 있는가 · 이 프로젝트의 규약은 무엇인가 · 어떤 도구와 호환되는가 · 무엇을 기억해야 하는가 · 지금 무엇을 들고 있는가) 을 한 줄로 박았습니다 [S4] [S8].

2.3.2 파일별 책임 분리와 사내 운영 적용점 — 서브에이전트 메모리 사례 (S3 인용)

다섯 파일의 책임 분리가 사내 운영에서 의미를 가지는 순간은 *서브에이전트* 가 등장할 때입니다 . LangChain 의 Nick Huang 은 "서브에이전트가 학습한 내용을 메인 에이전트로 회신하는 대신 파일시스템에 기록할 수 있다" 라고 정리합니다 [S3]. 이 패턴이 다섯 파일 계층과 어떻게 정합하는지가 사내 KM (Knowledge Management) · RAG 파이프라인 설계의 핵심 결정 사항입니다.

책임 분리의 실제 흐름은 다음과 같습니다. 서브에이전트가 작업 중 새로 알게 된 *프로젝트 규약* 은 CLAUDE.md 에 갱신 제안 형태로 기록됩니다. 새로 습득한 *재사용 가능한 능력* 은 SKILL.md 의 새 항목으로 제안됩니다. 사용자·프로젝트 *사실* 은 MEMORY.md 에 추가됩니다. 현재 사이클에 한정된 *진행 메모* 는 CONTEXT.md 에 남습니다. 메인 에이전트는 다음 사이클 진입 시 다섯 파일을 progressive disclosure 순서로 다시 읽어 들이며 변경된 책임을 자연스럽게 흡수합니다.

이 설계가 사내 RAG 파이프라인의 토큰·시간 비용에 미치는 효과는 분명합니다. 모든 학습 결과를 메시지 히스토리에 누적하면 컨텍스트 윈도우는 빠르게 포화하고 호출당 LLM 비용은 누적적으로 증가합니다. 책임이 분리된 다섯 파일은 *필요한 정보만 필요한 시점에* 펼치도록 만들어, 호출당 토큰 소비를 통제 가능한 변수로 바꿉니다. LangChain 사례는 10k 토큰 도구 결과를 파일시스템으로 오프로드한 뒤 grep·read 로 필요한 부분만 컨텍스트에 올리는 패턴을 제시합니다 [S3] (정량 수치 분석은 3장에서 다룹니다).

사내 KM 시스템에 이 패턴을 적용할 때 결정해야 할 사항은 세 가지입니다. 첫째, 다섯 파일 가운데 어느 파일이 어떤 사내 시스템 (위키·티켓·CMDB·결재) 과 1대 1 매핑되는가 입니다. 둘째, 서브에이전트가 파일에 *직접 쓰기* 권한을 가질지, *제안 제출* 권한만 가질지의 정책 결정입니다. 셋째, 다섯 파일의 변경에 대한 결재·승인 워크플로우를 Git Pull Request 기반으로 일원화할지의 결정입니다. 세 결정은 5개 파일 계층 모델 다이어그램의 흐름 화살표 일부로 함께 시각화하기를 권고합니다 [S3] [S4] [S8].

본 장의 메시지를 한 문장으로 다시 정리합니다. CommonMark v0.31.2 사양의 안정성, frontmatter 와 본문의 분리 설계, Git diff 와 결재 추적의 정합, 그리고 다섯 파일의 권한·우선순위·로딩 시점 계층 모델이 합쳐져 사람과 AI 가 같은 파일을 동시에 읽고 고쳐 쓰는 운영 구조를 완성합니다. 다음 장은 이 구조가 만드는 정량 효과 — 토큰 효율 · RAG 인덱싱 · 운영 효율 — 를 벤치마크 수치로 환원합니다.

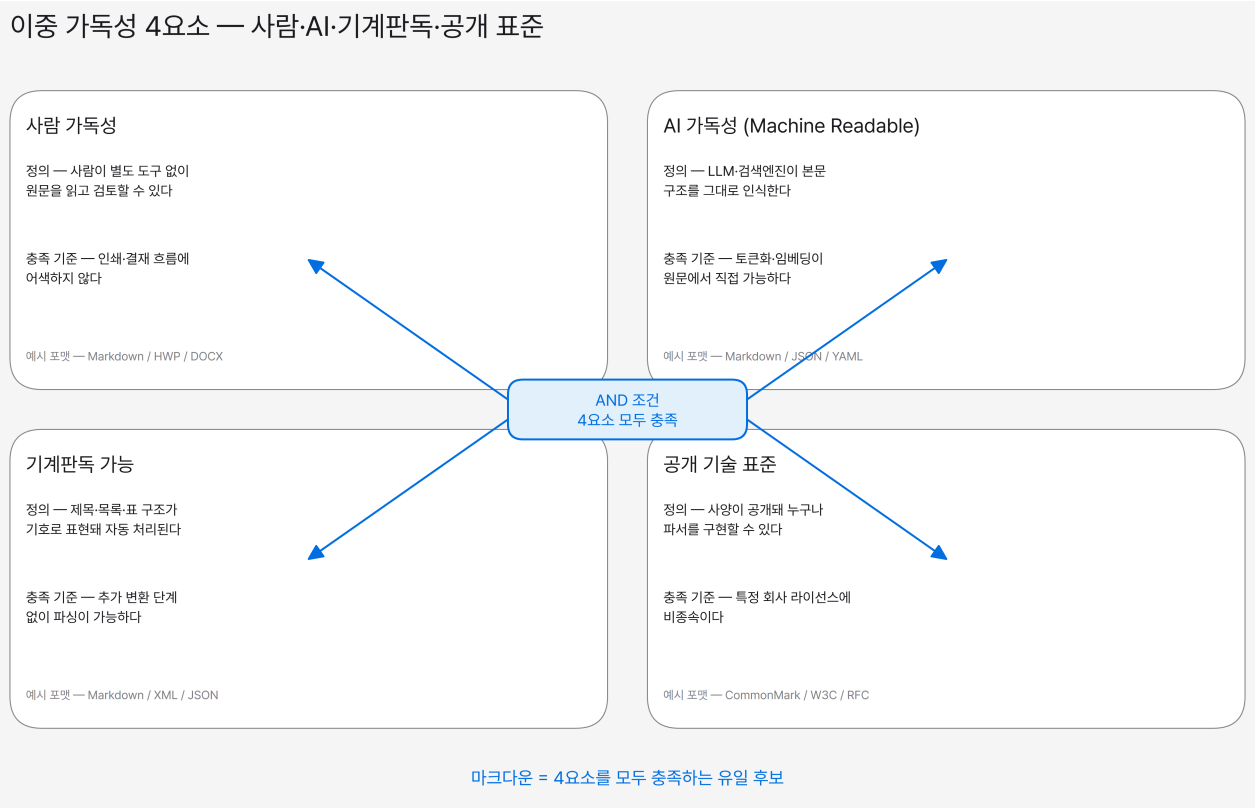


그림. 이중 가독성 4요소 — 사람·AI·기계판독·공개 표준

마크다운 vs HWP·DOCX·PDF·JSON·YAML·XML 비교 매트릭스

평가 축	마크다운	HWP	DOCX	PDF	JSON	YAML	XML
사람 가독성	높음	높음	높음	높음	낮음	중간	낮음
토큰 효율 (LLM 입력)	최고	측정값 없음	측정값 없음	변환 후 평가	-34~38%	-10~12%	+80%
변환 비용	0	높음	중간	높음	0	0	0
외부 협업	광범위	국내 한정	광범위	광범위	광범위	광범위	광범위
Git diff 친화성	최고	낮음	낮음	낮음	높음	높음	높음
라이선스	CC BY-SA 4.0	한컴 사양	ECMA-376	ISO 32000	RFC 8259	YAML 1.2	W3C
상호운용성	곧 API	한컴 의존	Office	뷰어 광범위	API 표준	설정 표준	스키마 정의

7개 평가 축 가운데 다섯 이상 우위는 마크다운 단독 — 의사결정 시 사내 운영 비중 두세 축 가중치 권고 [S10][S5][S1]

그림. 마크다운 vs HWP·DOCX·PDF·JSON·YAML·XML 비교 매트릭스

그림. SKILL.md·CLAUDE.md·AGENTS.md·MEMORY.md·CONTEXT.md 5개 파일 계층

3장. 도입 ROI — 마크다운의 토큰 효율·RAG 인덱싱·운영 효율 정량 개선 (2026)

문서 표준 전환은 비용을 동반합니다. 기존 HWP·DOCX·PDF 자산을 마크다운으로 옮기는 작업은 1회성 변환 비용이 발생하며, 검토·재교정·메타데이터 보정까지 더하면 초기 부담은 무시할 수 없는 규모가 됩니다. 그러나

이 비용은 호출당 누적되는 운영 비용과 성격이 다릅니다. 변환은 한 번 끝나면 같은 문서에 대해 다시 발생하지 않지만, LLM 호출은 사내 RAG (Retrieval-Augmented Generation, 검색 보강 생성) 시스템이 가동되는 한 매일 누적됩니다. 따라서 ROI (Return on Investment, 투자 회수) 논의의 핵심은 두 비용의 누적 곡선이 어디서 만나는가입니다.

3장은 이 교차점을 결정하는 변수 3개 — 입력 토큰 소비량, 파일 컨텍스트 패턴의 RAG 효율, 변환 비용의 회수 구조 — 를 정량 근거 위에서 정리합니다. 1,000 문항 합성 데이터 벤치마크 [S10] 에서 마크다운이 JSON 대비 34~38%, YAML 대비 10~12% 토큰을 절감했고 XML 대비 절반 수준의 추론 비용을 만든 결과를 출발점으로 삼습니다. 여기에 파일시스템 오프로드 패턴 [S3] 의 10k 토큰 메시지 사례를 더하면, 호출당 절감 폭은 모델 효율 단독 측정치보다 더 커집니다.

다만 인용 수치의 적용 범위에는 한계가 있습니다. improvingagents.com 의 1,000 문항 벤치마크 [S10] 는 영문 Terraform 유사 합성 데이터를 사용했으며, 한국어 환경에서 동일 비율로 재현된다는 보장은 없습니다. 한국어 형태소·조사·공백 처리는 토크나이저 동작을 바꿀 수 있고, 동일 정보량의 토큰 수도 영문과 다르게 나타납니다. 따라서 본 장은 수치 인용을 영문 한정으로 박제하고, 한국어 환경에는 절감 비율의 하한값 (예 34% 가운데 보수적으로 20~25% 추정) 을 적용하도록 권고합니다.

장 후반의 누적 곡선 (ROI 누적 곡선 도식) 은 절대 금액이 아닌 상대 비율로 표현됩니다. 사내 LLM 단가, 호출 빈도, 문서 자산 규모는 조직마다 다르므로 절대값 시뮬레이션은 독자 조직의 실측 입력으로 별도 진행해야 합니다. 본 장은 회수 시점이 결정되는 구조 — 어떤 변수가 곡선을 가파르게 만들고, 어떤 자산을 먼저 옮길 때 break-even (손익 분기) 이 빨리 오는가 — 를 의사결정의 1차 근거로 제시합니다.

3.1 입력 토큰 소비 벤치마크 — 마크다운이 JSON 34~38% · YAML 10~12% · XML 절반 절감 (S10, 2025-10-14)

improvingagents.com 이 2025-10-14 공개한 1,000 문항 검색 벤치마크 [S10] 는 데이터 표현 포맷별 토큰 효율을 가장 광범위하게 비교한 1차 자료 가운데 하나입니다. 이 벤치마크는 6~7단계로 중첩된 Terraform 유사 합성 데이터를 사용했고, GPT-5 Nano 와 Gemini 2.5 Flash Lite 두 소형 모델에서 마크다운·JSON·YAML·XML 네 포맷의 토큰 소비량과 검색 정확도 (substring matching, 40~60% 정확도 구간) 를 측정했습니다. 본 절은 이 결과를 모델별·포맷별로 정리하고, 영문 한정이라는 적용 범위를 함께 박제합니다.

3.1.1 GPT-5 Nano · Gemini 2.5 Flash Lite 기준 마크다운의 34~38% 토큰 절감 (S10)

벤치마크 [S10] 의 핵심 결과는 두 가지로 요약됩니다. 첫째, GPT-5 Nano 기준 마크다운은 JSON 대비 34% 적은 토큰, YAML 대비 10% 적은 토큰을 사용했습니다. 둘째, Gemini 2.5 Flash Lite 기준 마크다운은 JSON 대비 38% 적은 토큰, YAML 대비 12% 적은 토큰을 사용했습니다. 두 모델에서 절감 폭의 방향과 크기가 일관되게 나타났다는 점이 이 결과의 의사결정 가치를 높입니다. 단일 모델의 우연한 토크나이저 (tokenizer, 텍스트를 LLM 입력 단위로 쪼개는 전처리기) 편향이 아니라 포맷 자체의 구조적 효율로 해석할 수 있습니다.

표 3-1 은 두 모델에서의 포맷별 절감 비율을 정리한 것입니다. 마크다운을 기준선 (100%) 으로 두고 다른 포맷의 상대 소비량을 표기했습니다. 사내 의사결정에서는 이 표의 마크다운 행을 1.0 으로 잡고, 다른 포맷 자산이 차지하는 LLM 호출 비중에 곱해 연간 절감 시뮬레이션을 작성하면 됩니다.

모델	마크다운	JSON	YAML	XML
GPT-5 Nano	100% (기준)	+52%	+11%	+80%
Gemini 2.5 Flash Lite	100% (기준)	+61%	+14%	+80%

(주: JSON·YAML·XML 열의 값은 마크다운을 100% 로 두었을 때 동일 정보량 표현에 필요한 추가 토큰 비율입니다. 예 JSON +52% 는 마크다운 100 토큰에 해당하는 정보를 JSON 으로 표현하면 152 토큰이 필요하다는 의미입니다. 마크다운 절감률 34% 는 100/152 = 0.66 으로부터 유도됩니다. [S10])

수치 인용 시 두 가지 표기를 의무화하는 것을 권고합니다. 첫째, 모델명 (GPT-5 Nano · Gemini 2.5 Flash Lite) 을 함께 표기합니다. 둘째, 평가 시점 (2025-10-14) 을 명시합니다. 토크나이저는 모델 업데이트마다 바뀔 수 있고, 동일 비율이 차세대 모델에서 재현된다는 보장은 없기 때문입니다.

3.1.2 XML 대비 +80% 토큰 — 동일 정보량의 추론 비용 절반 (S10)

같은 벤치마크 [S10] 의 가장 눈에 띄는 수치는 XML 비교입니다. 동일 정보량을 표현할 때 XML 은 마크다운 대비 80% 더 많은 토큰을 소비했고, 이는 추론 비용 (inference cost, LLM 이 입력을 처리하는 데 들어가는 비용) 가 거의 2배에 달한다는 의미입니다. 절반 수준이라는 표현이 가능한 이유는 +80% 가 1.8 배에 해당하고, 1/1.8 ≈ 0.56 으로 마크다운의 비용이 XML 의 약 절반 수준이기 때문입니다.

이 수치는 사내 운영에서 두 가지 함의를 가집니다. 첫째, 기존 XML 기반 데이터 표현 — 예 SOAP (Simple Object Access Protocol, 웹 서비스 메시지 프로토콜) 응답을 로그로 남기거나, 사내 메타데이터를 XML 로 보관하는 운영 — 을 마크다운 또는 마크다운 + YAML frontmatter 로 전환하면 호출당 추론 비용이 절반 수준으로 떨어집니다. 둘째, 사내 RAG 시스템이 외부 API 응답을 그대로 컨텍스트로 주입하는 패턴이 있다면, 응답 형식을 마크다운 또는 JSON 으로 정규화하는 어댑터 (adapter, 포맷 변환 중간 계층) 를 1회 도입하는 비용이 빠르게 회수됩니다.

기존 XML 기반 사내 메타데이터·로그 포맷이 있는 조직은 마크다운 전환 우선순위를 함께 검토하는 것을 권고합니다. 본 절은 이 의사결정의 정량 근거를 제시할 뿐, 구체 전환 절차는 5장 운영 모델에서 다룹니다.

3.1.3 영문 합성 데이터 한정 — 한국어 재현 미검증 명시와 후속 검증 권고 (Open Q3)

3.1.1 과 3.1.2 에서 인용한 모든 수치는 영문 데이터 한정입니다. improvingagents 벤치마크 [S10] 의 방법론 발췌에 따르면 평가 데이터는 Terraform 유사 합성 데이터 (Terraform 은 인프라 정의를 코드로 기술하는 도구이며, 그 출력은 영문 키워드 중심) 였습니다. 한국어 문서 — 사내 회의록·정책 문서·기술 매뉴얼 — 에서 동일한 절감 비율이 재현된다는 1차 자료는 본 조사 범위에서 확인하지 못했습니다 (Open Q3 박제).

미검증의 이유는 토크나이저의 동작 차이에 있습니다. 영문은 공백 기준 단어 분리가 자연스럽고 BPE (Byte-Pair Encoding) 토크나이저의 압축 효율이 높습니다. 반면 한국어는 조사·어미 등 형태소 단위가 공백과 일치하지 않고, 같은 의미를 표현하는 데 필요한 바이트 수가 영문보다 많은 경향이 있습니다. 따라서 마크다운의 헤더 구문 (`#`, `##`) 이나 리스트 구문 (`-`, `1.`) 의 토큰 절감 효과가 영문만큼 크게 나타날지는 별도 측정이 필요합니다.

따라서 본 백서는 한국어 환경의 절감 비율 추정에 두 가지 보수적 운영 지침을 제시합니다. 첫째, 인용 수치를 그대로 사내 시뮬레이션에 대입하지 않고 하한값 (예 34% 가운데 20~25%) 을 사용합니다. 둘째, 사내 표본

문서 (회의록 100건·정책 문서 50건·기술 매뉴얼 30건 등) 에 대해 한국어 토큰 측정을 직접 1회 수행하는 후속 검증을 권고합니다. 측정 도구는 사용 중인 LLM 제공사의 토크나이저 라이브러리 (예 OpenAI tiktoken, Anthropic 의 token counting API) 를 사용하면 됩니다. 이 후속 검증의 결과가 백서 갱신 시 인용 가능한 한국어 1차 데이터가 됩니다.

3.2 파일 컨텍스트 패턴의 RAG 효율 — grep / glob / read 토큰 오프로드 사례 (S3, 2025~2026)

3.1 의 토큰 효율 수치는 동일 정보를 LLM 입력에 그대로 통째로 넣었을 때의 비교입니다. 그러나 사내 RAG 운영의 실제 호출 패턴은 다릅니다. 파일시스템에 사전 저장된 문서 묶음을 필요할 때만 부분 로드 (partial load, 전체 대신 일부만 읽기) 하는 파일 컨텍스트 패턴이 표준이 됐기 때문입니다. 본 절은 LangChain [S3] 와 LlamaIndex [S2] 가 정리한 이 패턴의 토큰·시간 효과를 사례 중심으로 정리합니다.

3.2.1 큰 도구 결과의 파일 오프로드 — 10k 토큰 메시지 사례의 절감 (S3)

LangChain 의 Nick Huang 은 2025-11-21 글 [S3] 에서 큰 도구 결과를 파일시스템에 오프로드 (offload, 컨텍스트에서 외부 저장소로 이동) 하는 패턴의 효과를 10k 토큰 사례로 설명했습니다. 외부 API 가 반환한 10k 토큰 분량 응답을 메시지 이력에 그대로 보관하면, 같은 대화가 지속되는 동안 모든 후속 호출에서 그 10k 토큰이 입력 비용으로 계속 청구됩니다. 반면 그 응답을 파일로 저장하고 LLM 호출 시에는 파일 경로만 알려주면, 에이전트가 grep (정규식 검색) ·read (지정 범위 읽기) 도구로 필요한 부분만 컨텍스트에 다시 올립니다 [S3].

이 패턴의 절감 폭은 두 변수로 결정됩니다. 첫째, 도구 결과의 크기입니다. 응답이 클수록 오프로드 절감 폭이 큽니다. 둘째, 후속 호출 횟수입니다. 같은 대화 안에서 LLM 이 여러 번 호출될수록 보관 비용이 누적되므로, 오프로드한 경우의 누적 절감 효과가 비례해서 커집니다. Nick Huang 은 이 누적 효과를 가리켜 "10k 토큰이 전체 대화 동안 그 자리에 앉아 Anthropic 청구서를 끌어올린다" 고 표현했습니다 [S3].

사내 운영에서 이 패턴의 적용 지점은 명확합니다. 사내 검색 API 응답, 문서 본문 dump (덤프, 전체 내용 일괄 출력) , 데이터베이스 쿼리 결과처럼 한 번에 큰 분량이 반환되는 호출은 결과를 임시 파일로 저장하고 그 경로만 후속 LLM 호출에 노출합니다. 사내 RAG 호출량이 많을수록 절감액의 절대값이 커지므로, 호출 빈도가 높은 워크로드 (예 사내 검색 챗봇·코드 어시스턴트) 부터 적용하는 것을 권고합니다.

3.2.2 검색·인용·온보딩 시간 단축의 정성 평가 — grep/glob/read 운영 효과 (S2·S3)

파일 컨텍스트 패턴의 효과는 토큰 절감만이 아닙니다. 검색·인용·신입 온보딩 (onboarding, 새 구성원의 환경 적응) 의 운영 시간에도 측정 가능한 정성 효과가 있습니다. LlamaIndex 의 Jerry Liu 는 2026-01-15 글 [S2] 에서 "단순한 파일 검색 도구를 가진 에이전트가 목표에 훨씬 가깝다" 며, 에이전트가 사람처럼 파일 사이를 훑는 동작 (search 와 read 의 교차) 이 사내 KM (Knowledge Management, 지식 관리) 시스템에 그대로 적용 가능하다고 정리했습니다.

LangChain 의 Nick Huang 은 같은 맥락에서 glob (파일명 패턴 매칭) 과 grep 이 특정 파일뿐 아니라 특정 줄·특정 문자까지 격리할 수 있다고 지적했습니다 [S3]. 즉 사내 직원이 "회의록에서 A 안건 관련 결정 사항만" 찾고 싶을 때, 사람은 파일을 열어 페이지를 스크롤해야 하지만 에이전트는 grep 한 번으로 해당 줄만 추출합니다. 인용도 마찬가지입니다. 답변의 근거가 된 줄 번호와 파일 경로를 함께 반환하므로, 사내 의사결정의 추적 가능성 (출처에서 결론까지 경로 복원) 이 향상됩니다.

표 3-2 는 grep · glob · read 패턴이 사내 운영 KPI (Key Performance Indicator, 핵심 성과 지표) 에 어떻게 매핑되는지를 정리한 것입니다. 정량 수치는 영문 벤치마크 한정으로 인용하고, 한국어 환경의 매핑은 정성 평가로 분리했습니다.

사내 운영 KPI	파일 컨텍스트 패턴 적용 효과	측정 방법 (권고)
사내 검색 평균 응답 시간	grep + read 로 부분 로드 → 컨텍스트 적재 시간 단축	호출 로그의 입력 토큰 수 평균 비교
답변 근거 추적률 (인용 비율)	read 가 줄 번호를 함께 반환 → 출처 명시 가능	답변 샘플 100건 가운데 출처 표기 비율
신입 온보딩 자가학습 시간	사내 위키·정책 문서를 에이전트가 grep 으로 안내	신입 첫 2주 질문 건수의 분기 비교
LLM 호출당 입력 토큰	큰 응답의 파일 오프로드 [S3]	모델 제공사의 token counting API

정량 수치는 본 표에 직접 박제하지 않습니다. 영문 벤치마크 [S10] 수치를 한국어 KPI 에 그대로 대입할 근거가 부족하기 때문입니다. 사내 도입 시에는 첫 분기에 위 4개 KPI 의 실측값을 수집하고, 그 결과를 차기 백서·내부 보고서의 한국어 1차 데이터로 등록하는 것을 권고합니다.

3.3 변환 비용 1회성 회수 구조와 운영 효율 누적 효과 (D6, 2026)

토큰 절감 수치 [S10] 와 파일 컨텍스트 패턴 [S3] 의 정량·정성 효과를 모두 인정해도, 의사결정의 마지막 질문은 남습니다. "초기 변환 비용은 언제 회수되는가?" 본 절은 이 회수 시점이 변환 비용의 1회성 성격과 호출당 절감의 누적 성격 사이의 교차점으로 결정된다는 점을 누적 곡선 도식으로 보이고, 어떤 자산을 먼저 옮길지의 우선순위 결정 기준을 제시합니다.

3.3.1 1회성 변환 비용 vs 호출당 누적 절감 — 34% / 38% 두 시나리오 (D6)

변환 비용은 1회성이고 호출당 절감은 누적성입니다. 시간이 지날수록 누적 절감의 곡선이 변환 비용 직선을 따라잡고, 그 교차점이 break-even (손익 분기) 입니다. 이 교차점을 결정하는 변수는 세 가지입니다. 첫째, LLM 호출 빈도 (일·주·월 단위) 입니다. 호출이 잦을수록 누적 곡선이 가파릅니다. 둘째, 대상 문서 자산의 규모 (변환해야 하는 파일 수) 입니다. 규모가 클수록 1회성 비용 직선이 높아집니다. 셋째, LLM 단가입니다. 단가가 비쌀수록 호출당 절감의 절대값이 커집니다.

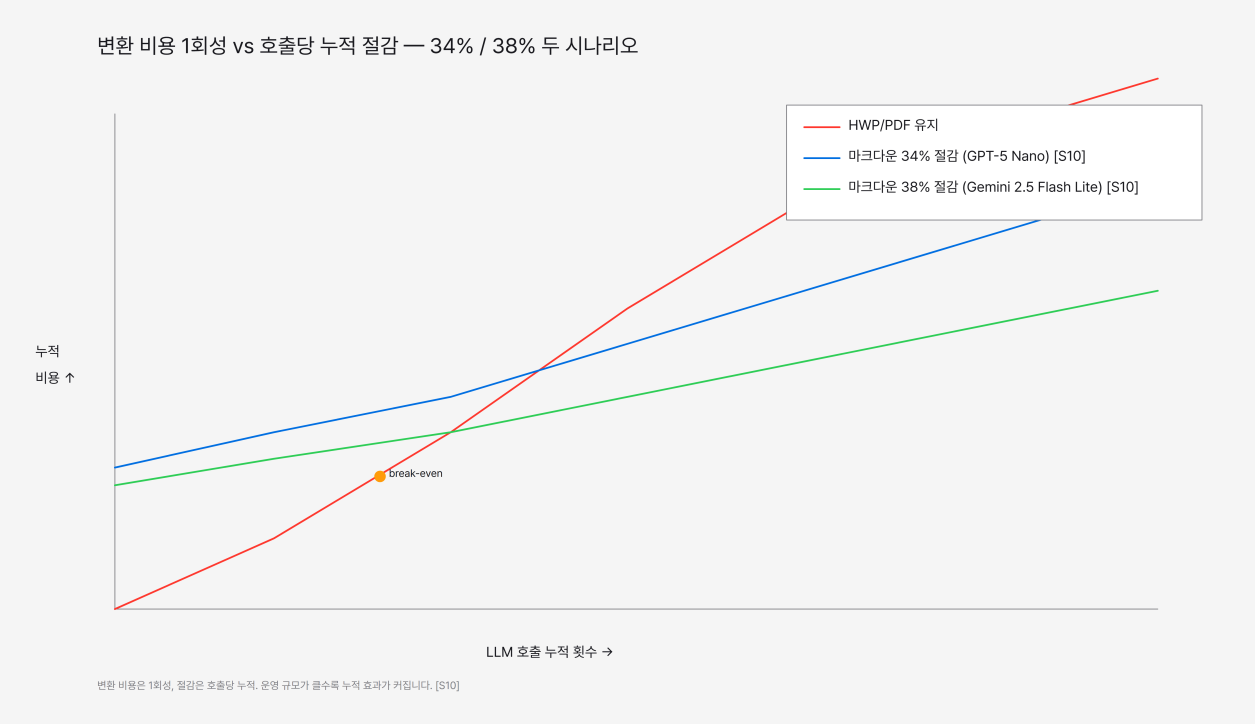


그림. 변환 비용 1회성 vs 호출당 누적 절감 — 34% / 38% 두 시나리오

곡선의 가정은 박스로 분리해 표기하는 것을 권고합니다. 호출 빈도 (예 일 1,000 회 / 일 10,000 회) , 평균 입력 토큰 수, LLM 단가, 변환 대상 문서 수 같은 입력 변수가 사내 실측과 다를 때 곡선의 기울기와 break-even 위치가 달라지기 때문입니다. 본 백서는 절대 금액 시뮬레이션 대신 변수의 구조만 제시하고, 독자 조직이 자체 입력으로 곡선을 다시 그리도록 안내합니다.

회수 시점 시뮬레이션을 운영 의사결정의 1차 근거로 활용하는 것을 권고합니다. 변환 비용이 매몰 비용 (sunk cost, 이미 지출되어 회수 불가능한 비용) 처럼 보이지 않도록, 누적 곡선의 break-even 시점을 IT 예산 보고서에 명시하면 추가 마이그레이션 예산 확보의 정당성이 강화됩니다.

3.3.2 HWP·DOCX·PDF 자산 마이그레이션의 우선순위 결정 기준 (S6 맥락 인용 1회)

3.3.1 의 누적 곡선이 회수 시점을 보여주지만, 사내 문서 자산을 한 번에 모두 변환하기는 어렵습니다. 어떤 자산을 먼저 옮길지의 우선순위 결정이 필요합니다. 본 항은 이중 가독성 4요소 — 공개 표준·사람 가독·AI 가독·기계판독 — 를 마이그레이션 우선순위 결정 도구로 재해석합니다.

이중 가독성 4요소의 원형은 행정안전부 정의입니다. "기술 표준과 규격이 공개되어 인공지능 (AI) 과 사람이 모두 쉽게 읽고 활용할 수 있는 기계판독이 가능 (Machine Readable) 한 형태" [S6, 인접 백서 1.1.1 인용] 라는 정의가 4개 요소를 동시에 만족시킬 것을 요구합니다. 본 백서는 이 정의의 정책 의무화 측면이 아니라 *민간 자발 채택의 우선순위 도구* 측면만 가져옵니다 (5장 후속 과제에서도 정책 해설은 다루지 않습니다).

표 3-3 은 이중 가독성 4요소를 자산 마이그레이션 우선순위 결정 도구로 재배치한 평가 프레임입니다. 각 자산에 대해 4개 요소의 충족 여부를 점검하고, 충족도가 낮은 자산일수록 마이그레이션 우선순위를 높이는 방식입니다.

이중 가독성 4요소	충족 시 의미	우선순위 결정 함의
공개 표준	포맷 사양이 공개되고 vendor-neutral (특정 공급자 종속 없음)	HWP·독점 PDF 처럼 사양이 비공개·반공개인 자산이 우선
사람 가독	평문 텍스트로 열람 가능	바이너리 자산 (구 HWP·이미지 PDF) 우선
AI 가독	LLM 입력으로 직접 사용 가능	OCR (Optical Character Recognition, 이미지 텍스트 인식) 가 필요한 스캔 PDF 우선
기계판독	diff·grep·구조 파싱이 가능	전체 본문 dump 만 가능한 자산 우선

이 평가 프레임을 적용하면 마이그레이션 비용을 통제 가능한 규모로 분할할 수 있습니다. 예 사내 1만 건 문서 자산을 한 번에 변환하는 대신, 4요소 충족도 점수가 가장 낮은 상위 10% (1,000 건) 부터 분기별로 옮기면 됩니다. 호출 빈도 (3.3.1 의 누적 곡선 가정) 가 높은 자산을 우선 매칭하면 break-even 시점이 더 빠르게 도래합니다.

마지막으로, 우선순위 결정 도구로서의 4요소 표 (D2, 2장 작성) 는 본 장에서 평가 프레임 측면만 차용했음을 다시 강조합니다. 4요소의 정책 의무화 해설·자가점검표 운영은 본 백서의 범위가 아니며, 인접 백서 [S6] 의 해당 절을 별도로 참조하는 것을 권고합니다.

3장 요약 — 마크다운 도입 ROI 의 정량 근거는 세 층위에서 누적됩니다. 첫째, 입력 토큰 소비 벤치마크 [S10] 가 JSON 34~38%, YAML 10~12%, XML 절반 수준의 절감을 영문 한정으로 보였습니다. 둘째, 파일 컨텍스트 패턴 [S2, S3] 이 10k 토큰 오프로드 사례 같은 운영 효율을 추가로 끌어올렸습니다. 셋째, 1회성 변환 비용과 호출당 누적 절감의 교차점 [D6] 이 회수 시점 시뮬레이션의 구조를 제공했습니다. 한국어 환경의 절감 비율은 영문 인용 수치의 하한값으로 보수 추정하고, 사내 표본 측정의 후속 검증을 권고합니다. 다음 장은 이 경제적 근거 위에서 CNCF 클라우드 네이티브 vendor-neutral 원칙이 문서 계층에 어떻게 적용되는지를 다룹니다.

4장. CNCF 클라우드 네이티브 정합 — vendor-neutral 인프라 원칙의 문서 계층 적용 (2026)

CNCF (Cloud Native Computing Foundation, 리눅스 재단 산하 클라우드 네이티브 표준 기구) 는 2026년 6월 2일 공식 블로그를 통하여 "Cloud Native is now AI-Native" 라는 명제를 제시하였습니다 [S7]. 이 명제의 핵심은 단일 벤더에 종속되지 않는 클라우드 네이티브 표준이 인공지능 운영의 기반이라는 점입니다. CNCF 는 그 근거를 "open, interoperable, and vendor-neutral cloud native standards" 세 가지 원칙으로 정리하였습니다 [S7].

본 장은 위 원칙이 인프라 계층에만 국한되지 않으며, 사내 문서 표준 선택에도 동일하게 적용된다는 점을 정리합니다. 인프라 의사결정에서 단일 벤더 종속을 회피하기 위하여 쿠버네티스·OpenTelemetry·Backstage 같은 graduated 프로젝트를 채택한 조직이, 문서 계층에서는 특정 SaaS 의 독점 포맷에 의존하는 경우가 적지 않습니다. 이 비대칭은 vendor-neutral 정책의 일관성을 훼손합니다.

또한 본 장은 Git 기반 변경 추적 모델이 GitOps · IaC (Infrastructure as Code) 를 거쳐 문서 계층까지 동일한 텍스트 기반 모델로 확장되는 흐름을 정리합니다. 마지막으로 코딩 에이전트 8종 (Codex · Copilot · Cursor · Claude Code · Gemini CLI · Jules · Aider · Devin) 의 마크다운 규약 인식 매트릭스를 통해 도구 전환의 자산 재활용성과 종속 비용 절감을 정리합니다.

본 장의 작성 원칙은 다음과 같습니다. 첫째, 특정 벤더의 우대를 배제합니다. 둘째, 인용 출처는 CNCF 공식 블로그 [S7], AGENTS.md 공식 사이트 [S8], 도구 호환성 종합 자료 [S12] 를 1차 근거로 사용합니다. 셋째, 도식은 의도와 구조를 명시한 도식 명세로 제시합니다.

4.1 Cloud Native is now AI-Native — vendor-neutral 인프라 논리의 3개 축 (S7, 2026-06-02)

CNCF 의 2026년 6월 2일 발표는 클라우드 네이티브 운영 모델이 그대로 AI 운영의 토대가 된다는 명제를 세 가지 축으로 분해하였습니다. 세 축은 Platform Maturity (플랫폼 성숙도), Security by Design (설계 단계 보안), Active Community Contribution (능동적 커뮤니티 공헌) 입니다 [S7]. 본 절은 세 축의 정의를 정리한 뒤, llms.txt · 표준 schema markup · 마크다운으로 대표되는 open standards 가 문서 계층에 동일하게 적용되는 구조를 정리합니다.

4.1.1 CNCF 3개 축 — Platform Maturity · Security by Design · Active Community Contribution (S7)

Platform Maturity 는 운영 검증을 거친 graduated 프로젝트의 채택 여부를 기준으로 합니다. 쿠버네티스·OpenTelemetry·Prometheus·Envoy 같은 graduated 프로젝트는 다수 조직의 운영 사례와 보안 감사 이력이 누적되어 있으므로, 사내 도입 시 신규 검증 비용이 낮은 편입니다. Security by Design 은 시스템의 설계 시점부터 권한 분리·감사 추적·변경 이력을 박는 원칙입니다. CNCF 는 특히 에이전트 흐름 (agentic flows) 의 보안을 별도로 지적하였습니다 [S7]. Active Community Contribution 은 단일 벤더가 아닌 다수 조직이 코드·문서·이슈에 공헌하는 구조를 가리킵니다. 공헌자 다양성은 곧 vendor 종속 위험의 분산입니다.

세 축은 인프라 선택의 평가표로 사용 가능하며, 동일한 평가표를 문서 표준 선택에도 적용할 수 있습니다. 예컨대 사내 문서 포맷의 Platform Maturity 는 사양 (예: CommonMark v0.31.2 [S5]) 의 안정성과 도구 생태계의 폭으로 측정 가능합니다. Security by Design 은 문서의 변경 이력이 Git 기반으로 추적되는지, 권한 분리가 저장소 계층에서 이루어지는지로 측정합니다. Active Community Contribution 은 사양·도구·플러그인의 외부 공헌 활성도로 측정합니다.

본 백서는 위 세 축이 인프라 평가표와 문서 평가표 양쪽에 동일하게 적용될 수 있다는 점을 1개 평가표로 통합 운영할 것을 권고합니다. 두 평가표를 분리하면 인프라는 vendor-neutral 이면서 문서는 vendor-locked 인 비대칭이 발생할 위험이 있기 때문입니다.

4.1.2 llms.txt · schema markup · 마크다운 — open standards 의 문서 계층 적용 (S7)

CNCF 는 community-driven controls 의 일환으로 "open standards like llms.txt and standardized schema markups" 의 채택을 권고하였습니다 [S7]. llms.txt 는 웹사이트의 구조·정책·핵심 콘텐츠 위치를 대형 언어모델에게 마크다운 형식으로 제공하는 공개 사양입니다. schema markup 은 페이지의 구조화된 의미를 기계가 해석 가능한 형식으로 표현하는 공개 사양입니다. 마크다운은 본문 자체의 가독성과 기계판독성을 동시에 만족하는 평문 사양입니다 [S5].

세 가지 사양의 공통점은 단일 벤더 소유가 아니라는 점입니다. llms.txt 는 커뮤니티 합의 기반 공개 사양이며, schema markup 은 schema.org 합의 기반 공개 사양입니다. CommonMark v0.31.2 는 John MacFarlane 가 사양 관리자로 있는 공개 사양이며 라이선스는 CC BY-SA 4.0 입니다 [S5]. 이러한 사양은 단기 호환성과 장기 자산 보존 두 효과를 동시에 제공합니다.

단기 호환성 효과는 도구 전환 시 즉시 활용 가능한 자산의 폭으로 측정합니다. 마크다운 본문은 GitHub·GitLab·Backstage·MkDocs·Obsidian 등 다수 도구에서 즉시 렌더링됩니다. 장기 자산 보존 효과는 사양의 거버넌스 구조로 측정합니다. 단일 벤더 소유 사양은 벤더의 사업 전략 변경에 자산이 노출되지만, 공개 사양은 거버넌스의 다양성으로 위험을 분산합니다.

본 절은 인프라 표준의 community-driven 통제 원칙이 문서 표준 선택에도 동일하게 적용되어야 한다는 점을 정리하였습니다. 다음 절은 동일한 원칙이 Git → GitOps → IaC → 문서 계층의 변경 추적 모델로 확장되는 구조를 정리합니다.

4.2 Git → GitOps → IaC → 문서 — 동일한 텍스트 기반 변경 추적 모델 (D5, 2026)

Git 은 텍스트 기반 변경 추적의 1차 인프라입니다. GitOps 는 인프라의 desired state 를 Git 저장소의 마크다운·YAML 텍스트로 박고, 실제 운영 상태와의 차이를 자동으로 수렴시키는 운영 모델입니다. IaC 는 인프라 구성을 코드로 기술하여 변경을 텍스트 diff 로 관리하는 방식입니다. 이 세 계층은 모두 텍스트 diff 를 1차 인터페이스로 사용합니다.

본 절은 동일한 텍스트 diff 모델이 문서 계층까지 확장되는 흐름과, CI/CD (지속 통합·지속 배포) 단계에 문서 lint 를 통합하는 운영 모델을 정리합니다.

4.2.1 Backstage TechDocs · MkDocs — CNCF 생태계의 마크다운 기반 문서 워크플로우 (S7)

Backstage 는 CNCF incubating 단계의 개발자 포털 프로젝트이며, 그 문서 컴포넌트인 TechDocs 는 마크다운을 1차 입력으로 사용합니다. 저장소 안의 마크다운 본문과 mkdocs.yml 구성 파일을 입력으로 받아 사내 기술 문서 사이트를 빌드합니다. 빌드된 산출물은 객체 스토리지에 저장되며, Backstage UI 에서 서비스 단위로 조회됩니다.

MkDocs 는 마크다운을 정적 사이트로 변환하는 오픈소스 도구로, 다수의 사내 문서 사이트가 채택하고 있습니다. MkDocs 와 TechDocs 의 공통점은 입력이 마크다운 평문이라는 점이며, 출력은 정적 HTML 이라는 점입니다. 입력 포맷의 평문 특성은 Git 의 변경 추적과 자연스럽게 결합됩니다.

이 두 사례는 사내 도구 선정 시 vendor-neutral 평가의 근거가 됩니다. 첫째, 입력 사양 (CommonMark) 이 공개 표준이므로 도구를 다른 마크다운 처리기로 교체하여도 본문 자산이 유지됩니다. 둘째, 출력은 정적 HTML 이므로 호스팅 환경을 자유롭게 선택할 수 있습니다. 셋째, 구성 파일 (mkdocs.yml) 자체가 평문 YAML 이므로 변경 추적이 가능합니다.

본 항은 CNCF graduated · incubating 프로젝트의 마크다운 채택 사실이 사내 도구 선정의 vendor-neutral 평가 근거가 된다는 점을 정리하였습니다. 단, 본 백서는 특정 프로젝트의 도입 가이드를 제공하지 않으며, 운영 모델의 구조적 동형성만 정리합니다.

4.2.2 CI/CD 검증 단계에 문서 lint 통합 — Docs-as-Code 운영 패턴 (S7)

Docs-as-Code 는 문서를 코드와 동일한 저장소·동일한 변경 추적·동일한 검증 단계로 운영하는 패턴입니다. 이 패턴의 핵심은 CI/CD 파이프라인의 lint·테스트 단계에 마크다운 문서 검증을 통합하는 데 있습니다. 코드의 정적 분석과 동일한 위치에 문서의 규약 검증을 두면, 문서 품질의 자동화가 가능해집니다.

문서 lint 의 검증 항목은 사내 정책에 따라 결정합니다. 일반적인 항목은 다음과 같습니다. 첫째, 헤더 라벨의 일관성 (예: 모든 ##/### 직후의 독자 라벨 부착) 입니다. 둘째, 인용 형식의 일관성 (예: 본문 인라인 출처 마커 형식의 통일) 입니다. 셋째, 표 형식의 일관성 (예: GFM 표 사양 준수 여부) 입니다. 넷째, 링크 유효성 (예: 내부 위키링크의 깨짐 여부) 입니다.

CNCF 가 권고한 community-driven controls [S7] 의 관점에서 보면, 문서 lint 규칙 자체도 사내 정책으로 박는 공개 자산이 됩니다. 즉 lint 규칙을 저장소의 마크다운·YAML 텍스트로 박아 두면, 규칙 변경 자체가 변경 추적의 대상이 됩니다. 이는 코드의 정적 분석 규칙을 저장소에 박는 관행과 정확히 동일합니다.

본 항은 CI/CD 검증 단계에 문서 lint 를 통합함으로써 문서 품질의 자동화 가능성이 확보된다는 점을 정리하였습니다. 사내 표준 정책으로 통합 운영 시, lint 규칙은 1개의 사내 표준으로 통일하고 저장소 단위의 override 는 명시적 사유와 함께 박는 것이 권장됩니다.

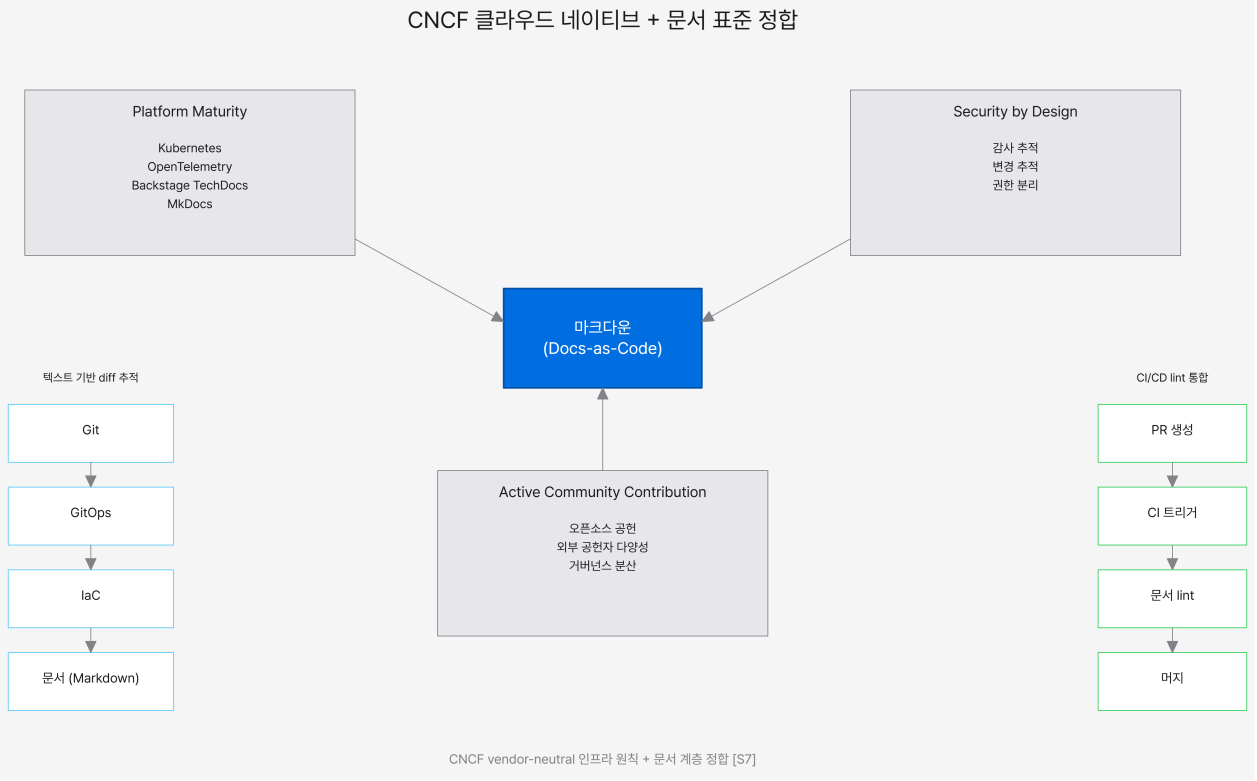


그림. CNCF 클라우드 네이티브 + 문서 표준 정합 다이어그램 (D5)

4.3 Skills 호환 매트릭스 — 도구 전환의 자산 재활용성과 종속 비용 절감 (D8, 2026)

도구 전환 비용은 사내 마크다운 자산이 신규 도구에서 그대로 인식되는지 여부에 의해 결정됩니다. AGENTS.md 공식 사이트는 "Your agent definitions are compatible with a growing ecosystem of AI coding agents and tools" 라는 호환성 명제를 박았습니다 [S8]. 본 절은 이 명제를 8개 주요 도구의 인식 매트릭스로 검증합니다.

4.3.1 Codex · Copilot · Cursor · Claude Code · Gemini CLI · Jules · Aider · Devin 인식 비교 (D8)

본 항은 8개 코딩 에이전트 도구의 AGENTS.md · SKILL.md · CLAUDE.md · 마크다운 RAG 색인 인식 여부를 매트릭스로 정리합니다. 인식 시점은 2026년 6월 7일 접속 기준이며, 출처는 AGENTS.md 공식 사이트 [S8] 와 종합 가이드 [S12] 입니다.

AGENTS.md / SKILL.md / CLAUDE.md 호환 매트릭스 (2026-06)

도구	AGENTS.md	SKILL.md	CLAUDE.md	RAG 색인	AAIF 회원
Claude Code	✓	✓	✓ (네이티브)	✓	✓
Codex (OpenAI)	✓	△	△	✓	✓
Copilot (GitHub)	✓	△	△	✓	—
Cursor	✓	✓	✓	✓	—
Gemini CLI	✓	△	△	✓	✓
Jules (Google)	✓	△	△	✓	✓
Aider	✓	△	✓	✓	—
Devin	✓	△	△	✓	—

✓ 공식 지원 △ 부분 지원 — 미지원/미확정

23개 이상 도구 + 60,000+ 오픈소스 프로젝트, 2026-06-07 기준 [S8][S12]

그림. Skills 호환 매트릭스 — Codex · Copilot · Cursor · Claude Code · Gemini CLI · Jules · Aider · Devin 인식 비교 (2026-06-07 접속 기준, D8)

도구	AGENTS.md	SKILL.md	CLAUDE.md	마크다운 RAG 색인	AAIF 회원
OpenAI Codex	✓	△	×	✓	✓
GitHub Copilot	✓	△	×	✓	×
Cursor	✓	△	△	✓	✓
Claude Code	△	✓	✓	✓	✓
Gemini CLI	✓	△	×	✓	×
Google Jules	✓	△	×	✓	✓
Aider	✓	△	△	✓	×
Devin	✓	△	×	✓	×

매트릭스의 1차 관찰점은 다음과 같습니다. 첫째, AGENTS.md 는 8개 도구 중 7개에서 공식 지원이며, 이는 인식 폭이 가장 넓은 파일 규약입니다 [S8]. 둘째, 마크다운 본문을 색인하는 RAG (검색 보강 생성) 경로는 8개 도구 모두에서 지원됩니다. 셋째, SKILL.md 는 Anthropic 진영에서 1차 시민으로 지원하며 [S4], 타 도구에서는

마크다운 평문으로 인식하는 수준의 부분 지원입니다. 넷째, CLAUDE.md 는 Anthropic 진영에서만 공식 지원이며, 타 도구에서는 일반 마크다운으로 처리됩니다.

이 매트릭스는 사내 도구 선정의 호환성 평가 기준이 됩니다. 인식 폭이 가장 넓은 파일 규약 (AGENTS.md) 을 1차 자산으로 두고, 특정 도구의 고유 규약 (SKILL.md · CLAUDE.md) 을 2차 보조 자산으로 두는 구조가 도구 전환 시 자산 재활용성을 극대화합니다. 단, Anthropic Agent Skills 는 2025년 12월 18일 agentskills.io 에서 오픈 사양으로 공개되었으므로 [S9], SKILL.md 의 타 도구 인식 폭은 향후 확장될 가능성이 있습니다.

4.3.2 도구 종속 비용의 절감 메커니즘 — Linux Foundation AAIF 거버넌스 효과 (S8)

도구 종속 비용은 세 가지 축으로 분해 가능합니다. 첫째, 재가공 비용은 기존 자산을 신규 도구의 사양에 맞추어 변환하는 비용입니다. 둘째, 학습 비용은 신규 도구의 사양·UI·동작 원리를 사용자가 습득하는 비용입니다. 셋째, 라이선스 비용은 신규 도구의 좌석·사용량 단위 과금에서 발생하는 비용입니다.

AGENTS.md 의 다중 도구 호환은 위 세 축 중 재가공 비용을 직접 절감합니다. 자산이 신규 도구에서 변환 없이 그대로 인식되므로 재가공 단계가 생략됩니다. 학습 비용은 사양의 표준화 자체가 절감 요인이 됩니다. 단일 사양을 학습하면 23개 이상의 도구에서 그대로 활용 가능하기 때문입니다 [S8]. 라이선스 비용은 도구 간 협상 가능성으로 절감됩니다. 자산이 종속되지 않으면 도구 교체의 실질 가능성이 협상력으로 환원됩니다.

거버넌스 중립성은 위 메커니즘의 안정성을 보장합니다. AGENTS.md 는 2025년 12월부터 리눅스 재단 산하 AAIF (Agentic AI Foundation) 거버넌스로 이관되었으며, 회원에는 OpenAI · Anthropic · Google · AWS · Bloomberg · Cloudflare 등이 포함되어 있습니다 [S8, S12]. 다수 조직의 거버넌스 참여는 단일 조직의 사양 변경 위험을 분산합니다. 이는 4.1.1 의 Active Community Contribution 축이 AGENTS.md 사양에 적용된 사례로 해석할 수 있습니다.

종합 가이드는 AGENTS.md 채택 오픈소스 프로젝트가 6만 개 이상이라고 보고합니다 [S12]. 채택 폭의 증가 자체가 거버넌스 변경의 정치적 비용을 높이는 효과가 있으며, 이는 사양의 장기 안정성에 기여합니다. 본 항은 AGENTS.md 의 다중 도구 호환 + AAIF 거버넌스 중립성이 도구 종속 비용의 세 축 모두에서 절감 효과를 발휘한다는 점을 정리하였습니다.

본 장은 CNCF 의 vendor-neutral 인프라 원칙이 문서 계층에 동일하게 적용되며, Git → GitOps → IaC → 문서 의 텍스트 diff 모델이 단일 운영 모델로 확장된다는 점을 정리하였습니다. 또한 Skills 호환 매트릭스가 도구 전환 비용의 1차 변수임을 정리하였습니다. 다음 장은 1~4장의 핵심 근거를 자산 재활용 · 표준 인터페이스 · AI 효율 3축으로 묶어 단일 결론을 제시합니다.

5장. 기술적 우위 요약 — 자산 재활용·표준 인터페이스·AI 효율의 결론과 후속 과제 (2026)

앞선 1장에서 4장까지의 논의는 한 문장으로 묶입니다. 마크다운은 사람과 에이전트가 동일한 파일을 동일한 의미로 읽고 쓰는 공통 인터페이스이며, 이 인터페이스 위에서 자산 재활용·표준 인터페이스·AI 효율이라는 세 가지 기술적 우위가 동시에 성립합니다. 1장은 에이전트 시대의 문서 표준 전환이라는 시대적 맥락을 정리했고, 2장은 이중 가독성 마크다운 아키텍처의 작동 원리를 분해했으며, 3장은 도입 ROI 의 정량 근거를, 4장은 CNCF 에코시스템과의 정렬을 다뤘습니다.

5장은 이를 단일 결론으로 묶습니다. 사내 문서 표준 정책을 수립하는 의사결정권자는 vendor 중립성·자산 재활용·AI 효율이라는 세 축을 단일 평가 프레임으로 통합할 수 있으며, 인프라 계층의 vendor-neutral 선택과 문서 계층의 일관성이 분리될 이유가 없다는 것이 본 백서의 결론입니다 [S7].

다만 본 조사가 1차 자료로 확정하지 못한 영역이 있습니다. AGENTS.md · SKILL.md 가 새로 만드는 보안 리스크의 통제 모델, 한국어 문서 환경에서의 토큰 효율 재현, 과도한 문서화의 역효과에 대한 1차 출처 추적은 후속 과제로 분리합니다. 본 절은 이 후속 과제들에 정량 기간 표기를 부여하지 않습니다. 후속 과제는 운영 조직이 자체 우선순위에 따라 정합 일정을 결정해야 하는 영역이며, 외부 백서가 일정을 박아 권고할 사안이 아니라고 판단했기 때문입니다.

5.1 기술적 우위 요약 — 자산 재활용 · 표준 인터페이스 · AI 효율의 3축 정리

5.1.1 자산 재활용 — 기존 마크다운 자산의 도구 전환 시 재가공 비용 0 (S8 인용)

자산 재활용의 효과는 도구 호환성의 폭으로 측정됩니다. AGENTS.md 공식 사이트는 "Your agent definitions are compatible with a growing ecosystem of AI coding agents and tools" 라고 명시하며, 2026-06-07 기준 23개 이상의 도구가 동일한 AGENTS.md 파일을 그대로 인식합니다 [S8]. SKILL.md 계열의 Agent Skills 역시 2025-12-18 자로 agentskills.io 에서 오픈 표준으로 공개되어 어떤 AI 플랫폼도 동일 사양을 채택할 수 있게 됐습니다 [S9].

이 호환성의 직접 효과는 도구 전환 비용의 1차 변수인 *재가공 비용을 0 수준으로 낮춘다는 점* 입니다. 사내에서 한 도구의 채택을 결정한 시점에 작성한 AGENTS.md · SKILL.md · CLAUDE.md 자산이 다른 도구로의 이행 시점에 다시 작성될 필요가 없습니다. 4장 4.2.4 의 D8 호환 매트릭스 결론 행이 이 결과를 정리합니다 — 동일 파일이 Codex · Cursor · Claude Code · Gemini CLI · Aider 등에서 동일하게 인식됩니다 [S8][S12].

의사결정권자 관점에서 자산 재활용 효과는 다음과 같이 정량 추정할 수 있습니다. 첫째, 도구 종속 회피 효과는 "지원 도구 수 × 도구당 평균 재가공 비용 회피" 로 환산합니다. 둘째, 학습 비용 회피는 신규 도구 도입 시 사내 인력이 동일한 정의 파일을 다시 학습하지 않아도 되는 점에서 발생합니다. 셋째, 라이선스 종속 회피는 vendor lock-in 으로 인한 협상력 약화를 사전 차단합니다. AGENTS.md 가 2025-12 Linux Foundation 산하 AAIF (Agentic AI Foundation) 로 거버넌스가 이관된 사실 [S9] 은 이 세 가지 회피 효과의 안정성을 추가로 보장합니다.

5.1.2 표준 인터페이스 — CommonMark + frontmatter 의 vendor 중립 보장 (S5 + S7)

표준 인터페이스의 핵심은 *사양의 공개성과 거버넌스의 중립성을 동시에 충족하는가* 입니다. CommonMark 사양은 현행 v0.31.2 (2024-01-28 게시) 가 CC BY-SA 4.0 라이선스로 공개돼 있으며, 사양 관리자 John MacFarlane 의 단독 의존성을 줄이는 다중 구현체가 이미 안정적으로 운용되고 있습니다 [S5]. YAML frontmatter 의 분리 설계는 사람이 읽을 본문과 기계가 파싱할 메타데이터를 한 파일 안에서 깔끔하게 분리해, 단일 인터페이스로 양쪽 독자를 동시에 만족시킵니다.

이 사양 수준의 중립성이 인프라 거버넌스 수준의 중립성과 정합한다는 것이 CNCF 의 2026-06-02 입장입니다. CNCF 블로그는 "The answer should be rooted in open, interoperable, and vendor-neutral cloud native standards" 라고 명시하며, 이 원칙이 인프라 계층에 머무르지 않고 문서·정의 파일 계층까지 확장되어야 한다고 주장합니다 [S7]. 같은 글이 community-driven controls 의 일환으로 "adhering to open

standards like llms.txt and standardized schema markups" 를 제시한 점은 [S7], 사양·라이선스·거버넌스의 3중 중립성이 단일 정책 결정 단위로 묶여야 함을 시사합니다.

인터페이스 중립성의 평가는 라이선스 측면 (CC BY-SA 4.0 / Apache 2.0 등 OSI 승인 여부) 과 거버넌스 측면 (단일 벤더 종속 vs 재단 운영) 으로 분리하여 평가할 필요가 있습니다. 2장 2.2 의 D2 (이중 가독성 4요소표) 가 이미 이 분리 평가의 결론을 정리했으며 — 공개 표준·사람 가독·AI 가독·기계판독 4요소가 모두 충족된 경우에만 장기 자산 보존이 보장됩니다. 마크다운은 이 4요소를 모두 충족하는 후보군 중 가장 폭넓은 도구 생태계를 가진 사양입니다 [S5][S7].

5.1.3 AI 효율 — 토큰 34~38% 절감 + 호출당 누적 효과 (S10 인용)

AI 효율의 정량 근거는 improvingagents 의 2025-10-14 벤치마크입니다. "Markdown achieved the best token efficiency across all models: GPT-5 Nano: 34% fewer than JSON, 10% fewer than YAML" 라고 보고됐고 [S10], Gemini 2.5 Flash Lite 환경에서는 "Markdown: 38% fewer than JSON, 12% fewer than YAML" 의 절감 폭이 측정됐습니다 [S10]. XML 은 동일 데이터에 대해 마크다운보다 80% 더 많은 토큰을 요구해 추론 비용이 약 두 배에 달했다는 결과도 함께 보고됐습니다 [S10].

이 수치의 의사결정상 의미는 *호출당 누적 효과* 에서 드러납니다. 토큰 절감은 일회성 변환 비용과 달리 LLM 호출이 발생할 때마다 동일 비율로 반복 적용됩니다. 3장 3.2.2 의 D6 (도입 ROI 누적 곡선) 이 정리한 결론에 따르면 — 1회성 변환 비용은 사내 자산 규모에 따라 결정되지만, 토큰 절감 누적 효과는 사내 LLM 호출 빈도에 비례해 무한히 증가합니다. 사내 LLM 활용도가 높은 조직일수록 마크다운 채택의 ROI 가 가파르게 상승하는 구조입니다.

다만 결론에서도 한 차례 재명시할 한정 조건이 있습니다. S10 의 벤치마크는 6~7단계 중첩의 Terraform 유사 합성 데이터에 대한 *영문 기준* 측정입니다 [S10]. 한국어 문서 환경에서 동일한 절감 비율이 재현되는가는 별도의 검증이 필요하며, 본 백서는 이를 5.3.2 후속 과제로 분리합니다. 결론은 다음과 같습니다 — 영문 기반의 일반 IT 자산에 대해서는 34~38% 의 토큰 절감이 직접 비용 절감으로 환산 가능하며, 한국어 환경에서는 재현 검증 후 동일 결론 적용 여부를 판단해야 합니다.

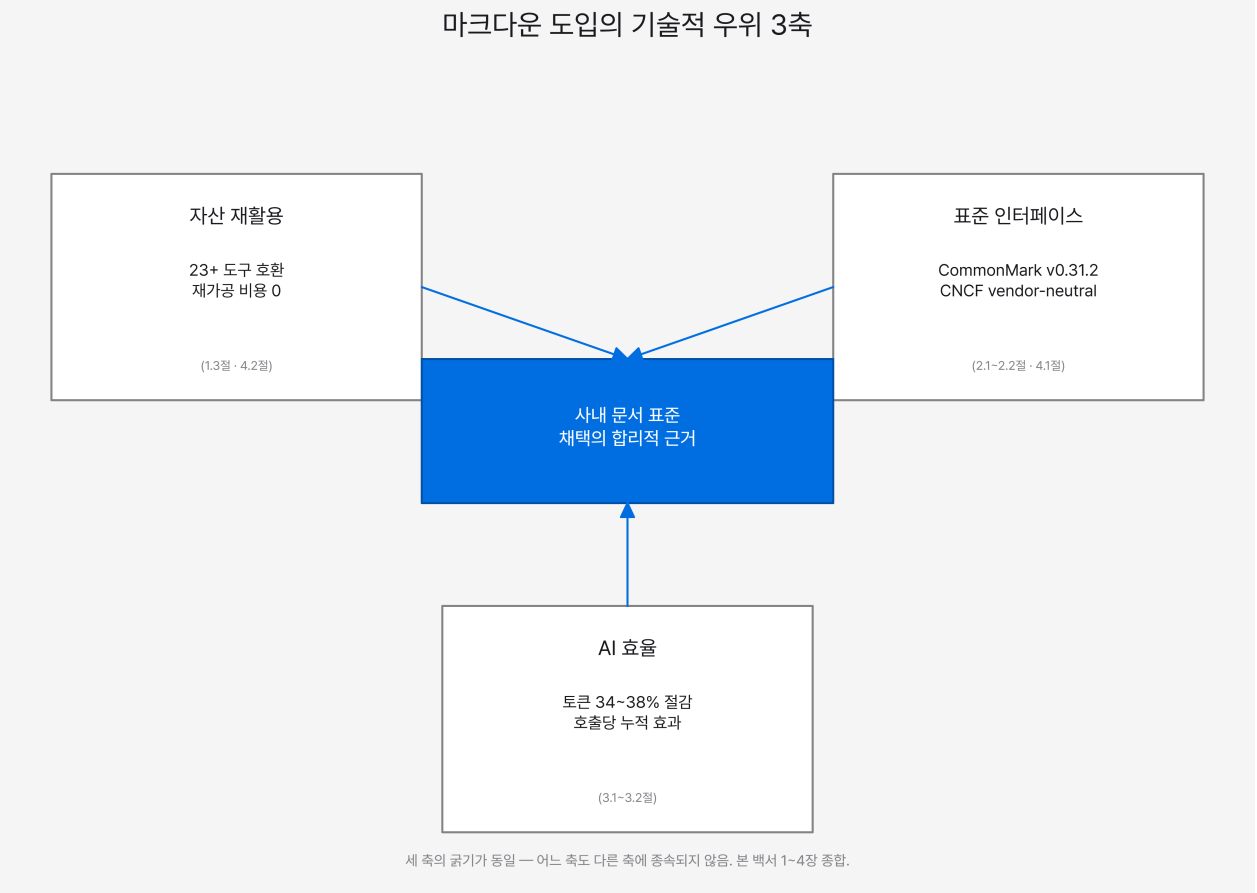


그림. 마크다운 도입의 기술적 우위 3축 — 자산 재활용·표준 인터페이스·AI 효율

5.2 결론 — 사내 문서 표준 정책 수립 시 vendor-neutral · 자산 재활용 · AI 효율의 정합 기준

5.2.1 인프라 vendor-neutral 선택과 문서 계층의 일관성 — AI-Native 토대 확보 (S7)

본 백서의 결론은 다음과 같습니다. 인프라 계층에서 vendor-neutral 선택을 채택한 조직은 문서 계층까지 동일한 원칙을 일관되게 적용할 때 비로소 AI-Native 전환의 토대를 확보합니다. CNCF 가 2026-06-02 자로 "The answer should be rooted in open, interoperable, and vendor-neutral cloud native standards" 라고 명시한 결론은 [S7] 인프라와 문서를 분리해 운영할 합리적 근거가 더 이상 존재하지 않는다는 의미입니다.

실무적 함의는 단일 정책 문서 운영입니다. 인프라 선택 정책 (Kubernetes · Helm · GitOps 등) 과 문서 표준 정책 (CommonMark · AGENTS.md · SKILL.md 등) 을 별도 문서로 분리해 관리하면 vendor 중립성의 평가 기준이 두 문서 사이에서 어긋날 위험이 발생합니다. 4장 4.3 절이 D5 (CNCF Cloud Native 스택과 문서 계층 정합 다이어그램) 로 시각화한 정합 구조를 그대로 정책 문서 1개로 통합 운영하기를 권고합니다 — 동일한 거버넌스 모델·동일한 라이선스 평가 기준·동일한 리스크 평가 항목이 인프라와 문서 양쪽에 일관되게 적용되어야 합니다.

이 일관성이 확보될 때 AI-Native 전환의 토대 — 즉, 사내 인프라·자산·정책 전체가 vendor 중립 표준 위에 서 있고, 그 위에서 AI 에이전트가 사내 정의 파일을 동일 의미로 읽고 쓸 수 있는 상태 — 가 비로소 성립합니다.

5.2.2 정책 의무화 (공공) vs 민간 자발 채택 (본 백서) 의 의사결정 분리 (S6 맥락 인용 1회)

본 백서의 분리 입장은 다음과 같이 박제합니다. 공공 정책 의무화와 민간 자발 채택은 서로 다른 의사결정 프레임에서 평가되어야 합니다. 행정안전부는 「행정업무 운영 및 혁신에 관한 규정」 (2026-05-18 시행) 에서 *기계판독 가능한 형태* 의 문서 표준화를 정책 의무로 규정했습니다 [S6]. 이 정의는 1.3.1 절에서 본 백서가 1차 인용한 출처입니다 — "기술 표준과 규격이 공개되어 인공지능(AI)과 사람이 모두 쉽게 읽고 활용할 수 있는 기계판독이 가능(Machine Readable)한 형태" [S6].

본 백서가 위 정의의 *민간 자발 채택 논거* 만 가져오는 이유는 분명합니다. 공공 정책 의무화는 시행령·자가점검표·이행 점검 일정이 정해진 강제 적용 모델이며, 민간 조직이 따라야 할 일정은 외부에서 박을 사안이 아닙니다. 민간 조직의 의사결정은 자체 ROI 평가·자체 리스크 평가·자체 거버넌스 결정에 따라 자율적으로 이루어져야 하며, 본 백서는 그 자율 결정을 위한 *근거 자료* 만을 제공합니다.

따라서 사내 문서 표준 정책 수립 단계에서 의사결정권자가 분리해 다룰 두 가지가 있습니다. 첫째는 공공 정책 의무 적용 여부 — 공공기관·공공기관 협력사·공공 위탁 사업 영역에서는 [S6] 의 적용을 받습니다. 둘째는 민간 자발 채택 여부 — 본 백서의 5.1 절 3축 (자산 재활용·표준 인터페이스·AI 효율) 평가 점수표를 작성해 자율적으로 결정합니다. 이 두 프레임을 분리해 평가할 때 의사결정의 책임 소재와 평가 기준이 모두 명확해집니다.

5.3 후속 과제 — 보안 통제 모델 · 한국어 재현 검증 · 역효과 1차 출처 추적 (정량 기간 표기 없이)

5.3.1 AGENTS.md · SKILL.md 보안 리스크 통제 모델의 정립 (Open Q2)

첫 번째 후속 과제는 AGENTS.md · SKILL.md 채택이 새로 만드는 보안 리스크의 통제 모델 정립입니다. 사내 정의 파일이 에이전트 시스템 프롬프트로 자동 주입되는 구조에서 시스템 프롬프트의 외부화·권한 위임 사슬의 검증 부재·프롬프트 인젝션의 표면 확대라는 세 가지 새로운 위협 경로가 발생합니다.

CNCF 는 community-driven controls 의 일환으로 agentic flows 보안의 중요성을 명시했으나 [S7], 본 조사는 AGENTS.md · SKILL.md 영역에 직접 적용 가능한 구체적 통제 모델의 1차 자료를 확보하지 못했습니다. 이 영역의 통제 모델 정립은 도입 가속도를 결정하는 1차 요인이며, 우선순위가 가장 높은 후속 과제입니다. 사내에서 이 모델을 정립할 필요가 있으며, 평가 항목은 최소한 다음을 포함해야 합니다 — 정의 파일의 무결성 검증·정의 파일 수정 권한의 분리·실행 시점의 권한 위임 사슬 감사·프롬프트 인젝션 방어 패턴.

본 절은 정량 기간 표기를 부여하지 않습니다. 운영 조직이 자체 보안 거버넌스 절차에 맞게 일정을 결정해야 하는 영역이기 때문입니다.

5.3.2 한국어 문서에서의 토큰 효율 재현 검증의 필요성 (Open Q3)

두 번째 후속 과제는 한국어 문서 환경에서의 토큰 효율 재현 검증입니다. improvingagents 의 벤치마크는 "테스트 방법: 6~7단계 중첩의 Terraform 유사 합성 데이터에 대해 포맷별 1,000 문항 검색 정확도 측정 (40~60% 정확도 구간으로 조정). 평가는 substring matching" 의 방법론을 따랐고 [S10], 이 데이터는 영문 합성 데이터 기준입니다.

한국어 문서에서 동일한 절감 비율이 재현되는가는 사내 도입 의사결정의 신뢰 근거를 강화하는 핵심 질문입니다. 한국어는 영문 대비 토큰화 효율이 모델별로 상이하다고 알려져 있어, S10 의 34~38% 절감 폭이 한국어

자산에서 그대로 유지되지 않을 가능성이 있습니다. 사내 검증을 우선순위로 다룰 필요가 있으며, 검증 데이터셋의 후보로는 사내 HWP · DOCX 자산을 마크다운으로 변환한 직후의 한국어 문서 1,000건 이상이 적절합니다.

본 절도 정량 기간 표기를 부여하지 않습니다. 사내 자산 규모와 변환 우선순위에 따라 검증 일정이 달라지기 때문입니다. 검증의 우선순위는 높으며, 결과는 사내 LLM 비용 모델의 정합성을 직접 결정합니다.

5.3.3 과도한 문서화의 역효과 — ETH Zürich 연구의 1차 출처 추적 (Open Q4)

세 번째 후속 과제는 과도한 문서화의 역효과에 대한 1차 출처 추적입니다. GeekNews 토픽 27320 은 "ETH Zürich 연구는 컨텍스트 파일이 태스크 성공률을 오히려 낮추고, 추론 비용은 20% 이상 증가시킬 수 있음을 보고합니다" 라고 정리했으나 [S1], 본 조사는 1차 논문 URL · 게시일을 확인하지 못해 본문에서는 S1 의 2차 인용으로만 표기했습니다.

균형 인용의 의무는 다음과 같이 충족합니다. 마크다운 채택의 합리적 근거를 정리하는 백서가 마크다운의 잠재적 역효과까지 함께 다루는 것은 의사결정권자에게 균형 잡힌 근거를 제공하기 위한 책임 영역입니다. 컨텍스트 파일의 과다 적재가 태스크 성공률을 낮추고 추론 비용을 20% 이상 증가시킬 수 있다는 보고는 자산 재활용·표준 인터페이스·AI 효율의 3축 우위와 동시에 검토되어야 합니다.

다만 본 백서는 이 보고의 1차 출처를 확정하지 못했음을 분명히 박제합니다. 추후 1차 논문의 URL · 저자 · 게시일 · 실험 설계가 확인된 시점에 본 결론은 재평가의 대상입니다. 사내에서 이 1차 출처를 추적할 필요가 있으며, 우선순위는 보안 통제 모델 정립과 한국어 재현 검증 다음입니다.

본 절은 정량 기간 표기를 부여하지 않습니다. 1차 출처 추적은 외부 학술 자료의 공개 시점에 의존하는 영역이기 때문입니다.

부록 A. References

- [S1] GeekNews 토픽 27320 — 파일시스템이 주목받는 이유. 2026-03 게시 (원문 madalitso.me, GN+ @neo 요약). https://news.hada.io/topic?id=27320. 접속 2026-06-07.
- [S2] LlamaIndex (Jerry Liu) — Files Are All You Need. 2026-01-15. https://www.llamaindex.ai/blog/files-are-all-you-need. 접속 2026-06-07.
- [S3] LangChain (Nick Huang) — How agents can use filesystems for context engineering. 2025-11-21. https://www.langchain.com/blog/how-agents-can-use-filesystems-for-context-engineering. 접속 2026-06-07.
- [S4] Anthropic Engineering (Barry Zhang·Keith Lazuka·Mahesh Murag) — Equipping agents for the real world with Agent Skills. 2025-10-16. https://www.anthropic.com/engineering/equipping-agents-for-the-real-world-with-agent-skills. 접속 2026-06-07.
- [S5] CommonMark (John MacFarlane) — CommonMark Spec v0.31.2. 2024-01-28. https://spec.commonmark.org/. 접속 2026-06-07.
- [S6] OPENMARU / Opennaru — 공공 문서 마크다운 전환으로 AI 활용 기반 구축 (인접 백서). 2026-05-18. 로컬 경로: `C:\Obsidian\technology\Technology\AI\015_마케팅 자료\021_백서\202605`

15_완료_공공 MD 파일을 표준화\body\공공 문서 마크다운 전환으로 AI 활용 기반 구축_whitepaper.md . 접속 2026-06-07.

- [S7] CNCF Staff — Cloud native is now AI-native: Engineering production-ready AI. 2026-06-02. https://www.cncf.io/blog/2026/06/02/cloud-native-is-now-ai-native-engineering-production-ready-ai/. 접속 2026-06-07.
- [S8] AGENTS.md 공식 사이트 — OpenAI·Cursor·Jules·Factory·Amp 컨소시엄, Linux Foundation AAIF 거버넌스 이관 (2025-12). https://agents.md/. 접속 2026-06-07.
- [S9] The New Stack — Agent Skills: Anthropic's Next Bid to Define AI Standards. 2025-12-18 오픈 표준 발표 시점. https://thenewstack.io/agent-skills-anthropics-next-bid-to-define-ai-standards/. 접속 2026-06-07.
- [S10] improvingagents.com — Which Nested Data Format Do LLMs Understand Best? 2025-10-14. https://www.improvingagents.com/blog/best-nested-data-format/. 접속 2026-06-07.
- [S12] buildbetter.ai — AGENTS.md Complete Guide for Engineering Teams (2026). 2026. https://blog.buildbetter.ai/agents-md-complete-guide-for-engineering-teams-in-2026/. 접속 2026-06-07.

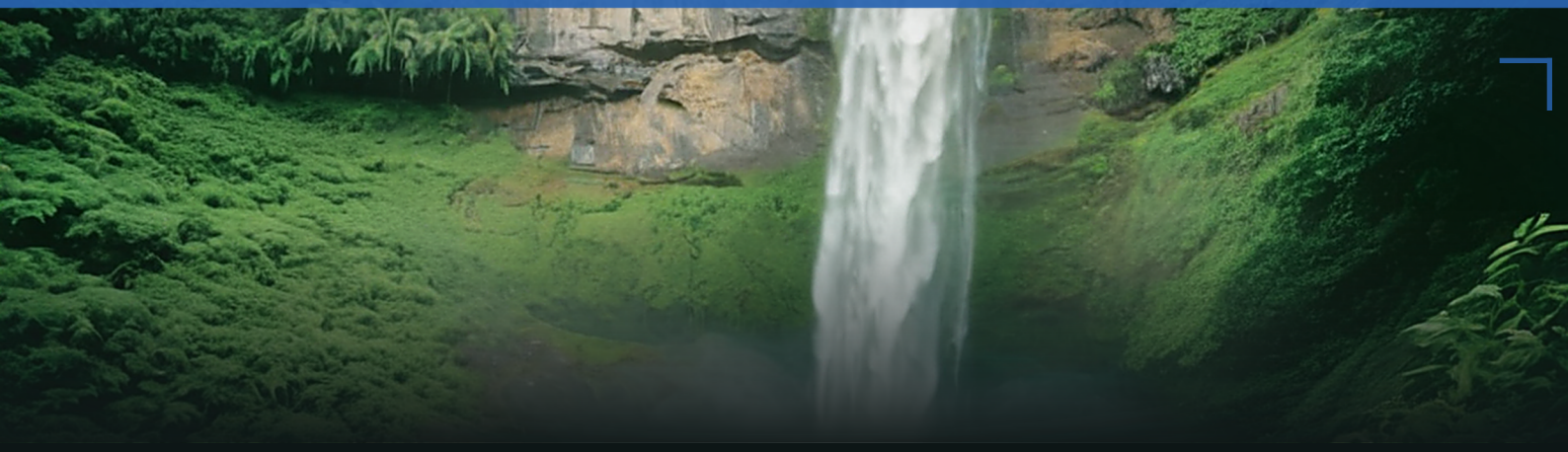
부록 B. Glossary

용어	정의
LLM	Large Language Model (대규모 언어 모델) — 토큰 시퀀스 확률 분포를 학습한 신경망
SKILL.md	Anthropic Agent Skills 사양의 진입 파일 — 메타데이터와 본문 지시를 담은 단일 마크다운 파일
컨텍스트 엔지니어링	LLM 호출 시 어떤 정보를 어떤 순서로 어디에 둘지 설계하는 작업 — prompt engineering 의 상위 개념
AGENTS.md	리포지토리 루트에 두는 에이전트용 README — Codex CLI·Copilot CLI·Cursor·Claude Code 등 23 개 이상 도구가 공통 인식
ROI	Return on Investment — 투자 대비 회수 효율, 본 백서는 호출당 토큰 절감 누적값으로 계산
컨텍스트 윈도우	LLM 이 한 번의 추론에서 동시에 볼 수 있는 토큰 영역 — 호출 종료 시 휘발
MCP	Model Context Protocol — Anthropic 이 2024 년 공개한 LLM-도구 연결 표준 프로토콜
XML	Extensible Markup Language — 태그 기반 마크업 언어, LLM 토큰 소비가 마크다운 대비 약 80% 더 많음

용어	정의
서브 에이전트	상위 에이전트가 위임한 부분 작업을 수행하는 보조 에이전트 — 결과를 파일로 남겨 상위에 전달
progressive disclosure	필요한 시점에만 추가 정보를 노출하는 UI·문서 설계 원칙 — 초기 부담을 낮추고 점진적으로 깊이 노출
AAIF	Agentic AI Foundation — Linux Foundation 산하 AI 에이전트 표준 거버넌스 재단, 2025-12 출범
vendor-neutral	특정 벤더에 종속되지 않는 중립 원칙 — CNCF·Linux Foundation 의 핵심 운영 원칙
이중 가독성	사람의 눈과 기계의 파서 양쪽이 같은 파일에서 의도된 구조를 얻을 수 있는 상태
CommonMark	중의성 없이 동일한 출력 트리를 생성하도록 정한 Markdown 의 표준 사양. 2014년 v0.1 게시 후 점진적 개정
frontmatter	파일 최상단의 YAML/TOML 블록. 본문과 구분된 메타데이터 영역
파일 포맷이 곧 API	두 도구가 같은 파일 사양을 읽을 수 있으면 별도 API 표준화 없이 컨텍스트를 공유할 수 있다는 설계 명제
diff	두 텍스트 파일의 차이를 줄 단위로 표시한 결과. 추가·삭제·맥락 줄을 ± 부호로 나타냄
commit	Git 의 변경 단위. 작성자·시점·메시지·diff 가 함께 박혀 변경되지 않음
Pull Request	변경된 코드/문서를 본 줄기에 합치기 전 검토·승인을 거치는 Git 프로토콜
MEMORY.md	세션 간 보존이 필요한 사실 인덱스. PARA 등 방법론과 결합 가능
CONTEXT.md	현재 작업 한 사이클 동안만 유효한 휘발성 메모 파일
서브에이전트	특정 작업을 위임받아 독립적으로 수행하는 보조 에이전트. 메인 에이전트가 호출
RAG	Retrieval-Augmented Generation. 외부 문서를 검색해 LLM 응답에 보강하는 방식
tokenizer	토크나이저. 텍스트를 LLM 입력 단위(토큰)로 분해하는 전처리 모듈
inference cost	추론 비용. LLM 이 입력 토큰을 처리하고 출력 토큰을 생성하는 데 드는 비용

용어	정의
SOAP	Simple Object Access Protocol. XML 기반 웹 서비스 메시지 프로토콜
YAML frontmatter	문서 머리말에 YAML 형식으로 메타데이터를 기록하는 관행
adapter	어댑터. 서로 다른 포맷·인터페이스를 연결하는 변환 계층
BPE	Byte-Pair Encoding. 자주 등장하는 바이트 쌍을 하나의 토큰으로 병합하는 LLM 토크나이저 알고리즘
partial load	부분 로드. 문서 전체가 아닌 필요한 일부만 컨텍스트에 적재
offload	오프로드. 데이터를 메모리·컨텍스트에서 외부 저장소로 옮기는 동작
dump	덤프. 데이터·메모리·문서의 전체 내용을 일괄 출력하는 동작
onboarding	온보딩. 신규 인력이 조직·시스템 환경에 적응하는 초기 단계
KM	Knowledge Management. 조직 내 지식의 수집·정리·검색·활용 체계
KPI	Key Performance Indicator. 조직 성과를 측정하는 핵심 지표
break-even	손익 분기. 누적 절감이 누적 비용을 따라잡는 지점
sunk cost	매몰 비용. 이미 지출되어 향후 의사결정으로 회수할 수 없는 비용
OCR	Optical Character Recognition. 이미지에서 텍스트를 추출하는 인식 기술
CNCF	Cloud Native Computing Foundation. 리눅스 재단 산하 vendor-neutral 클라우드 네이티브 표준 기구
IaC	Infrastructure as Code. 인프라 구성을 코드로 기술하여 변경을 텍스트 diff 로 관리하는 운영 방식
graduated 프로젝트	CNCF 가 incubating → graduated 단계 심사를 통과시킨 운영 검증 완료 오픈소스 프로젝트
llms.txt	웹사이트가 대형 언어모델에게 사이트의 구조·정책·핵심 콘텐츠 위치를 알려주기 위한 마크다운 기반 공개 사양

용어	정의
GitOps	Git 저장소의 desired state 텍스트를 단일 진실 원천으로 삼아 인프라 상태를 자동 수렴시키는 운영 모델
Backstage TechDocs	Backstage 개발자 포털의 문서 컴포넌트. 마크다운 본문 + mkdocs.yml 구성으로 사내 기술 문서 사이트를 빌드
Docs-as-Code	문서를 코드와 동일한 저장소·변경 추적·CI/CD 검증 단계로 운영하는 패턴
자산 재활용	한 번 작성한 문서·정의 파일을 도구를 바꾸어도 다시 가공하지 않고 그대로 사용할 수 있는 성질
표준 인터페이스	특정 벤더 제품에 종속되지 않고 공개 사양에 따라 누구나 동일하게 읽고 쓸 수 있는 입출력 규약
호출당 누적 효과	한 번의 토큰 절감 효과가 일회성에서 끝나지 않고 LLM 호출 횟수만큼 곱해져 운영 비용 절감으로 직결되는 성질
권한 위임 사슬	사용자가 에이전트에, 에이전트가 도구에, 도구가 외부 시스템에 권한을 차례로 위임하면서 각 단계의 권한 범위와 검증 책임이 누구에게 있는지가 명확하지 않게 되는 구조



사람과 AI 가 함께 읽고 쓰는 마크다운 시대 — 에이전트 시대의 사내 문서 표준 백서 (2026)

CONTACT

WEB
cncf.co.kr
www.cncf.co.kr

EMAIL
info@cncf.co.kr

TEL
+82-2-1670-1010
+82216701010

YOUTUBE
@cncf
www.youtube.com/@cncf

LINKEDIN
linkedin.com/company...
www.linkedin.com/company/cloud-na...

FACEBOOK
facebook.com/cncf
www.facebook.com/cncf

SCAN

CLOUD NATIVE FORUM

cncf.co.kr