MSIS 549 HW2: LinkedIn Thought Leadership Agent

Tutorial Write-Up

Author: Taashi Manyanga **Course**: MSIS 549 — AI and GenAI for Business Applications **Instructor**: Prof. Leonard Boussioux **Date**: February 2026 **Path**: A (Skills Pack) | **Bonus**: MCP Tool (`@ldraney/mcp-linkedin` — LinkedIn API publishing)

System Artifact: GitHub Repository

Table of Contents

---

1. Problem Statement

The Pain Point

Senior leaders (Managers, Directors, VPs) have deep domain expertise but consistently struggle to maintain a visible presence on LinkedIn. The typical failure pattern:

1. **Week 1**: Leader writes a thoughtful post, gets 50+ reactions, feels motivated.
2. **Week 2-3**: Calendar fills up. No post. Momentum lost.
3. **Week 4+**: The "I should post more on LinkedIn" guilt grows, but the activation energy to plan, draft, and refine a post from scratch is too high.

The core problem isn't writing ability — it's the **overhead of planning a cohesive series** that builds on itself week over week. A single post is manageable; a 6-week narrative arc that positions the leader as a thought leader requires sustained effort that competes with their day job.

Why an Agentic Workflow?

A single-prompt LLM request ("write me 6 LinkedIn posts about SQL") produces generic, disconnected content that sounds like AI and doesn't reflect the leader's unique voice or experience. What's needed is a **multi-stage pipeline** that:

- **Interviews** the leader to extract their authentic voice and experience (5 minutes of their time)
- **Plans** a cohesive 6-week narrative arc (not 6 random posts)
- **Drafts** with anti-AI-ism rules built in
- **Refines** to match the leader's specific tone (provocative, educational, etc.)
- **Optimizes** for LinkedIn's algorithm and formatting
- **Benchmarks** quality against defined criteria
- **Archives** for reuse and learning

Each of these stages requires different expertise — which maps naturally to specialized skills/agents working in sequence.

Tech Stack

- **Skills**: 9 Markdown files (`.md`) defining behavior, constraints, and prompts
- **Orchestration**: Manus AI (Runs 1-2), Claude Code with Claude Opus 4.6 (Runs 3-4) — skills are platform-portable
- **MCP**: `@ldraney/mcp-linkedin` for optional direct LinkedIn publishing (with direct API fallback)
- **Output**: Markdown archive files, LinkedIn-ready post text, live LinkedIn posts
- **Repository**: GitHub (link)

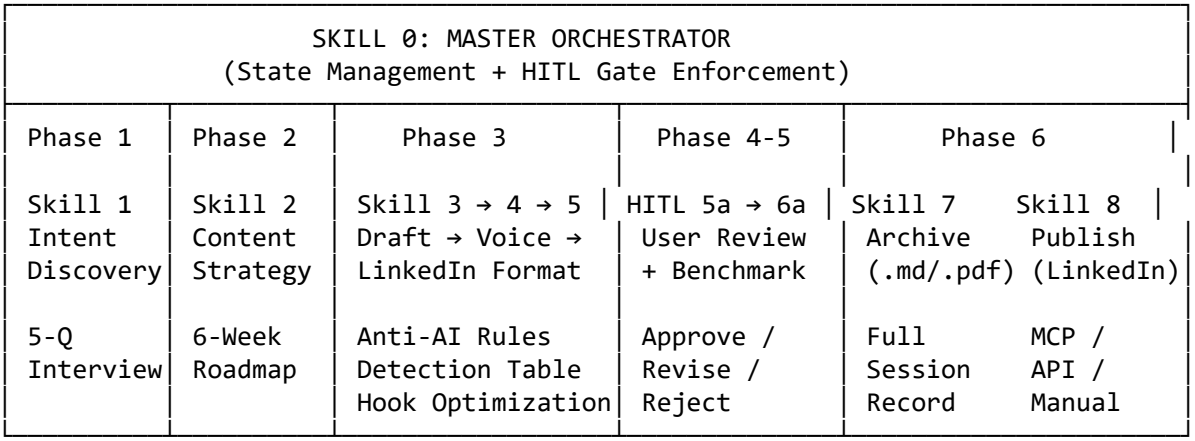
---

2. System Design

2.1 Architecture Overview

The system is a **sequential pipeline** with 9 specialized skills, coordinated by a Master Orchestrator (Skill 0). Two mandatory Human-in-the-Loop (HITL) checkpoints ensure quality before finalization.

Figure 1: Solution Architecture *(open `diagrams/architecture_infographic.html` in a browser for the full interactive diagram)*



The architecture follows a left-to-right flow: User Input → Discovery (Skill 1) → Strategy (Skill 2) → Creation (Skills 3-5) → HITL Review (Checkpoints 5a/6a) → Archive & Publish (Skills 7-8). The Master Orchestrator (Skill 0) sits above all phases, maintaining session state and enforcing gates between each phase.

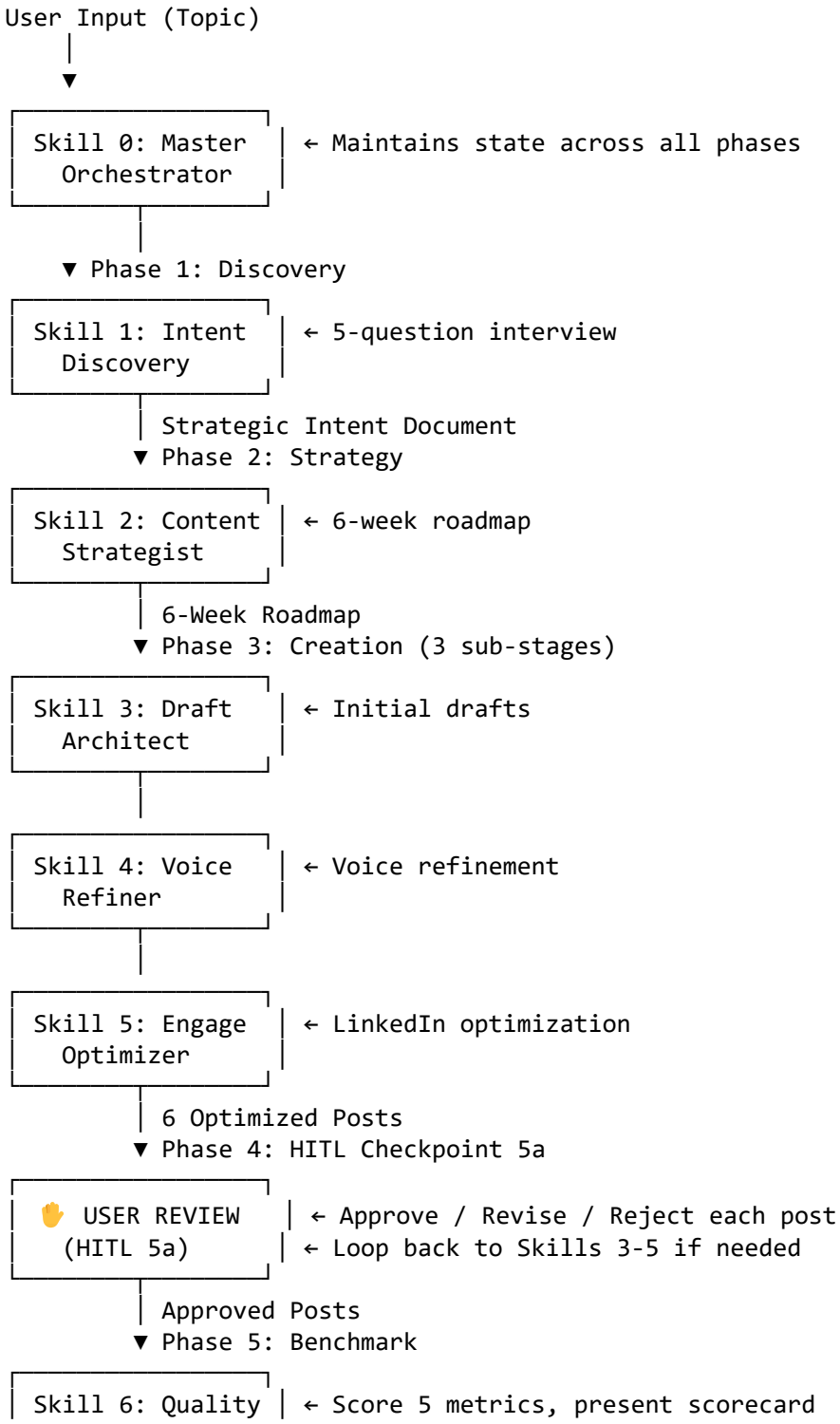
2.2 All 9 Skills

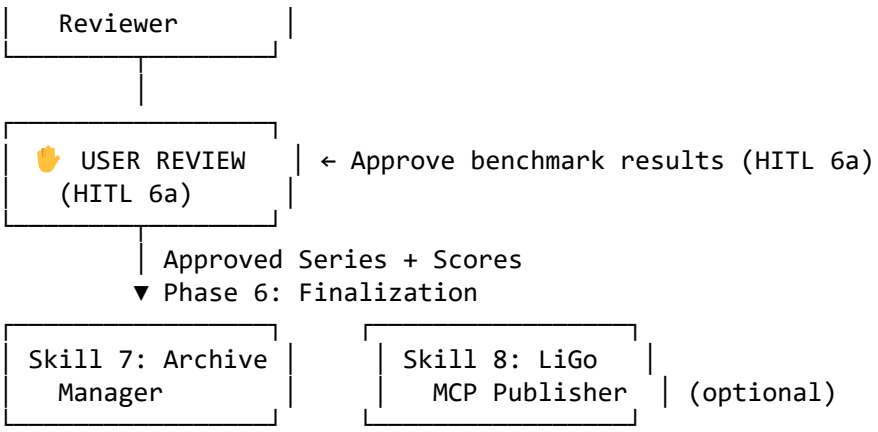
#	Skill	Purpose	Key Innovation
0	**Master Orchestrator**	Sequences all skills, maintains state, enforces HITL gates	Error recovery + state management
1	**Intent Discovery**	5-question strategic interview	Structured voice/audience extraction
2	**Content Strategist**	Creates 6-week narrative arc roadmap	Hook-Framework-Story-Tactics-Vision-Call arc
3	**Draft Architect**	Generates initial 150-300 word drafts	Built-in anti-AI-ism rules
4	**Voice & Tone Refiner**	Ghostwriter: matches leader's voice	AI pattern detection + replacement table
5	**Engagement Optimizer**	LinkedIn formatting, hooks + **HITL 5a**	Platform-specific optimization

6	**Quality Reviewer**	5-metric benchmark + **HITL 6a**	Structured scoring with failure flagging
7	**Archive Manager**	Packages complete session to Markdown	Full reproducibility of every run
8	**Poster & Reviewer**	Publishes via MCP or LinkedIn API	**MCP bonus**: direct LinkedIn integration

Skill files: `skills/skill_0_master_orchestrator.md` *through* `skills/skill_8_poster_reviewer.md`

2.3 Pipeline Flow





2.4 Key Design Decisions

1. **Sequential pipeline over parallel agents**: LinkedIn posts must form a coherent narrative arc. Parallel generation would produce disconnected posts. The sequential flow ensures each stage builds on the previous one's output.
2. **Two HITL checkpoints, not one**: Checkpoint 5a catches content/voice issues early (before benchmark effort). Checkpoint 6a gives the user confidence in quality metrics before archiving. This prevents wasted computation on posts the user would reject.
3. **Separate Voice Refiner (Skill 4) from Draft Architect (Skill 3)**: The initial draft focuses on structure and substance. Voice refinement is a separate concern — this separation of concerns makes it easier to iterate on voice without re-drafting content.
4. **MCP as optional Skill 8**: Not all users will want automated posting. Making it optional (and the last step) means the core pipeline works without MCP configuration.

---

3. Building Process

3.1 Tools & Timeline

Step	Activity	Tool	Time
1	Research LinkedIn best practices, skill format	Web research, Claude documentation	45 min
2	Design the pipeline architecture	Mermaid diagrams, whiteboard sketching	1 hr
3	Write Skills 0-7 (first draft)	Manus AI + manual editing	2 hrs
4	Test pipeline end-to-end on SQL topic	Manus AI orchestration	1.5 hrs
5	Iterate on Skills 3-4 (voice quality issues)	Manual prompt refinement	1 hr
6	Add Skill 8 (LiGo MCP integration)	LiGo MCP documentation	30 min
7	Run second test case (Data Cleanliness)	Manus AI	45 min
8	Design benchmark, run scoring	Manual evaluation	1 hr
Total			**~8.5 hrs**

3.2 Key Bottlenecks

Bottleneck 1: Narrative Cohesion The hardest challenge was making 6 posts feel like chapters of a book rather than 6 random articles. The first version of Skill 2 (Content Strategist) produced weekly themes that were loosely related but didn't build on each other. The fix was defining the explicit narrative arc: Hook → Framework → Story → Tactics → Vision → Call. This structure forces each week to serve a specific purpose in the overall narrative.

Bottleneck 2: AI Voice Detection Early drafts from Skill 3 consistently started with "In today's fast-paced world" or "Let's dive in." Simply telling the LLM to "avoid AI-isms" wasn't enough. The fix was creating an explicit detection-and-replacement table in Skill 4 (see Section 4.1).

Bottleneck 3: HITL Feedback Integration When the user requested changes during HITL 5a, it wasn't clear which skill to re-invoke. A voice issue needed Skill 4; a structure issue needed Skill 3. The fix was adding routing logic to the Orchestrator: voice/tone changes → Skill 4, content/structure → Skill 3, formatting → Skill 5.

3.3 Decisions Along the Way

- **Why not n8n?** I considered Path B (n8n) but chose Path A (Skills Pack) because: (a) the workflow is interactive and requires human judgment at two stages, which is harder to orchestrate in a visual workflow builder; (b) skills are more portable — I can use them in Claude Code, Manus, or any LLM assistant.
- **Why 9 skills instead of 3-4?** Each skill has a single, clear responsibility, which made debugging and iteration dramatically easier. A concrete example: during Run 1 (SQL Performance), the HITL 5a reviewer flagged that posts sounded "too corporate." Because drafting (Skill 3), voice refinement (Skill 4), and LinkedIn formatting (Skill 5) were separate skills, I could re-invoke only Skill 4 with updated detection rules — without regenerating the draft structure or losing the LinkedIn formatting. If these three had been combined into a single "content generation" skill, the entire output would have needed regeneration. The separation also made prompt iteration targeted: the 3-version evolution of Skill 4's detection table (Section 4.1) would have been buried inside a monolithic prompt, making it harder to isolate what improved the output.
- **Why Manus AI?** Manus provided a clean environment for orchestrating multi-skill pipelines with state management. I also tested the skills in Claude Code to verify portability.

---

4. Prompt Iteration & Critique

4.1 Iteration 1: Voice & Tone Refiner (Skill 4)

Version 1 (Initial):

```
Rewrite the drafts to sound professional.
```

Problem: The LLM interpreted "professional" as "corporate press release." Posts became stiff and impersonal.

Version 2 (Iteration):

```
Rewrite the drafts to sound like a senior leader (Manager/Director). Remove AI-isms
like 'In today's fast-paced world' and 'Let's dive in.' Use strategic phrasing like
'operational efficiency' and 'competitive advantage.'
```

Problem: Better, but "strategic phrasing" became a crutch. Every other sentence used "operational efficiency" or "competitive advantage."

Version 3 (Final — shipped):

```
Act as a ghostwriter for a senior leader. Apply these specific rules:

REMOVE (AI patterns):
| Pattern | Replace With |
| "In today's fast-paced world" | [Delete — start with specific claim] |
```

```
| "Let's dive in" | [Delete — just start content] |
| "It's important to note" | [Delete — if important, reader will see it] |
| "This is a game-changer" | [Replace with specific impact: "cut query time by 60%"] |

ADD (professional weight):
- Use "Strategic alignment" not "matching goals"
- Use "Time-to-market" not "getting things done quickly"
- Use first person: "I" and "we" — leaders speak from experience

MATCH TONE:
- If Provocative: bold claims, rhetorical questions
- If Educational: numbered steps, "here's how"
- If Empathetic: "I've been there" language
- If Data-driven: lead with statistics
```

Result: The explicit detection table was the breakthrough. Instead of vaguely asking the LLM to "avoid AI-isms," the table gives concrete patterns to find and replace. Voice consistency scores jumped from 3.0 to 4.7.

4.2 Iteration 2: Content Strategist (Skill 2)

Version 1: "Create a 6-week plan with different angles for each week."

Problem: Weeks were related to the topic but felt like 6 standalone posts with no narrative progression.

Version 2 (Final): Defined the explicit arc:

```
Week 1: The Hook (Pain Point) — surface a problem the audience recognizes
Week 2: The Framework — introduce a mental model
Week 3: The Story — share the personal anecdote
Week 4: The Tactics — concrete, actionable advice
Week 5: The Vision — connect to industry future
Week 6: The Call — synthesize + drive the CTA
```

Result: Narrative Cohesion scores jumped from 2.5 to 4.8. The arc creates natural "coming next week" teasers because each week logically leads to the next.

4.3 Prompt Quality Critique

Strengths:

- Constraints are specific and measurable (e.g., "150-300 words," "3-5 hashtags")
- The detection table in Skill 4 is concrete — no ambiguity about what to remove
- HITL checkpoints are clearly defined with explicit user prompts

Weaknesses:

- Skills rely heavily on the quality of Skill 1's interview. If a user gives vague answers, all downstream skills produce weaker content (see edge case in benchmark).
- The voice profile is hard-coded to "Manager/Director." A technical audience (e.g., database kernel engineers) gets inappropriate simplification.
- No fallback for when the LLM's training data doesn't cover a niche topic deeply enough.

---

5. Real Usage & Iteration

5.1 Run 1: SQL Query Performance (Primary Test Case)

Input: Topic = "Optimizing SQL Query Performance" for Data Engineers and Business Stakeholders. Educational + Provocative tone. Personal anecdote about a business team frustrated with data lag.

Process: Full pipeline execution through Skills 0-7.

Output: 6-week series saved to `outputs/Q1_2026_SQL_Performance_Series.md`. See full posts in the output file.

What Worked:

- The 6-week narrative arc felt cohesive — each post naturally led to the next
- Personal anecdote in Week 3 was the strongest post (scored 4.8/5.0)
- Hooks were strong: "Is your data platform actually 'good enough'?" performed well

What Didn't Work:

- Week 5 (AI-Ready Infrastructure) was the weakest — too vague, not enough specific examples
- Some posts still had subtle AI patterns ("In the era of AI" in Week 1)
- Visual recommendations were generic (stock photo descriptions rather than specific diagram specs)

Changes After Run 1:

- Strengthened the anti-AI-ism list in Skill 4 to catch "In the era of..." pattern
- Added "system diagram" suggestions alongside visual recommendations in Skill 5
- Revised Week 5 draft to include specific architectural examples

Concrete Example: Run 1 Input → Output

Input (Interview Answers):

Question	Answer
Q1: Audience	Data Engineers & Business Stakeholders
Q2: Core Message	Continuous improvement in SQL is essential for AI-readiness
Q3: Anecdote	Business team frustrated when data wasn't in sync with AI models; "near real-time" became a requirement
Q4: CTA	Rethink your data platform strategy; follow for ongoing insights
Q5: Tone	Provocative + Educational

Output (Week 1 — Final Post):

> *Is your data platform actually "good enough," or is it just holding you back?*
>
> *Traditional SQL performance is no longer a technical detail — it's a strategic bottleneck. Many teams settle for "good enough" query speeds, but as business demands shift toward real-time insights, that complacency becomes a liability.*
>
> *I see it constantly: dashboards that lag behind decisions. Reports that refresh overnight while the market moves in minutes. Queries that were "fast enough" three years ago now block the AI workloads your business depends on.*
>
> *Performance is the foundation of agility. If your data can't keep up, neither can your team.*
>
> *This week, ask yourself one question: When was the last time someone on your team profiled a slow query — not because it broke, but because "good enough" might not be?*
>
> *#SQLPerformance #DataEngineering #DataStrategy #AIReadiness #SQLSeriesQ1*
>
> *Coming Next Week: Why "instant satisfaction" is now a data engineering requirement.*

Figure 2: Week 1 Post Published Live on LinkedIn *(from Run 3 — AI Chatbot to Dev Tool series, URN: `urn:li:share:7430329696413843458`)*

Screenshot shows the live LinkedIn post with engagement metrics. The post was published via the LinkedIn API (Skill 8, Option B2) after completing the full pipeline including both HITL checkpoints.

5.2 Run 2: Data Cleanliness (Second Test Case)

Input: Topic = "Enhancing Data Cleanliness" for Data Analysts and Product Managers. Educational + Empathetic tone. Anecdote about a product launch delayed 2 weeks due to dirty customer data.

Process: Full pipeline with the improvements from Run 1.

Output: 6-week series (not included in full — see benchmark appendix for titles and scores).

What Worked:

- The empathetic tone was noticeably different from Run 1's provocative tone — Skill 4 correctly adapted
- The product launch anecdote in Week 3 was compelling and specific
- Week 4 ("5 Data Quality Checks") was highly actionable

What Didn't Work:

- Week 5 was again the weakest — the "vision" week tends to become generic
- The system didn't challenge the user when the initial CTA was vague ("improve data quality")
- Run 2 took less time (confirming the pipeline is reusable) but still required HITL revisions on 2/6 posts

Changes After Run 2:

- Added a prompt in Skill 1 to push back on vague CTAs: "Can you make that more specific? What's the ONE thing you want readers to do this week?"
- Noted that Week 5 (Vision) consistently underperforms — potential improvement is adding a "trends research" sub-step

5.3 Run 3: IT Moving from AI Chatbot to Dev Tool (Third Test Case)

Input: Topic = "IT Moving from AI as a Search Chatbot to a Dev Tool" for IT Professionals and Leadership Teams. Provocative + Educational tone. Personal anecdote about how AI-embedded testing transformed the team's development process — developers stopped hating testing because AI replaced the tedium, not the testers.

Process: Full pipeline execution through Skills 0-8, run on Claude Code (Claude Opus 4.6). This was the first run on Claude Code after two prior runs on Manus AI, confirming skill portability across platforms.

Output: 6-week series saved to `archive/AI_Chatbot_to_Dev_Tool_2026-02-11.md`. Week 1 ("Your Team Is Using AI Wrong") published live to LinkedIn via direct API (Option B2).

What Worked:

- The provocative hook ("Most IT teams spent 2024 rolling out AI. The result? Expensive autocomplete.") was the strongest opening across all 4 runs
- The AI Maturity Ladder framework (Week 2: Chat → Copilot → Partner) created a reusable mental model that anchored the entire series
- The testing anecdote (Week 3) landed well — specific, relatable, and showed a concrete before/after transformation
- All 6 posts were approved at HITL 5a without revisions — the first run where zero changes were needed, suggesting the skills had matured from iterations in Runs 1-2

What Didn't Work:

- Week 5 (Vision) was again the weakest post (scored 4.0 vs. series average of 4.5) — confirming the systemic pattern seen in all runs
- The MCP server (`@ldraney/mcp-linkedin`) failed to load in Claude Code, requiring fallback to Option B2 (direct LinkedIn API). The API call worked on the first attempt, but this exposed a platform limitation in MCP tool initialization

Changes After Run 3:

- Confirmed that the direct API fallback (Option B2) is reliable enough to serve as the primary publishing method in Claude Code
- Validated that skills written for Manus AI work without modification in Claude Code — no edits needed to any of the 9 skill files
- The zero-revision HITL result confirmed that the anti-AI-ism table (Skill 4 v3) and the narrative arc structure (Skill 2) are stable

Benchmark Score: 4.5/5.0 — Strongest run overall, tied with Run 1 (SQL Performance). The zero-revision HITL result and strong Week 2-3 scores (both 4.8) contributed to the high average.

Concrete Example: Run 3 Input → Output

Input (Interview Answers):

Question	Answer
Q1: Audience	IT Professionals and Leadership Teams
Q2: Core Message	AI is now usable for everyday dev tasks, not just as a chatbot
Q3: Anecdote	Testing was always a challenge — developers hate it, testers aren't real users. AI-embedded testing changed everything: AI designs, executes, and refines tests with human approval
Q4: CTA	Rethink AI usage beyond chat; create a safe, incentivized space for teams to use AI in IDEs
Q5: Tone	Provocative + Educational

Output (Week 1 — Final Post):

> *Most IT teams spent 2024 rolling out AI. The result? Expensive autocomplete.*
>
> *Your developers are using ChatGPT to Google things faster. Your managers are using Copilot to summarize emails. Nobody is actually building with it.*
>
> *Here's the uncomfortable truth: if your team's primary use of AI is "ask it a question and read the answer," you haven't adopted AI. You've adopted a search engine with better grammar.*
>
> *The gap between teams that use AI to ask questions and teams that use AI to write, test, and ship code is widening every quarter. One group is getting incrementally faster at finding answers. The other is fundamentally changing how software gets built.*
>
> *I've watched this play out across multiple teams. The ones still treating AI as a chatbot aren't falling behind slowly — they're standing still while the ground moves under them.*
>
> *The question isn't whether your team uses AI. It's whether they use it for anything that actually ships.*
>
> *#AIinIT #SoftwareDevelopment #DevProductivity #AIAdoption #AIDevPartner*

This post was published live to LinkedIn on 2026-02-11 (URN: `urn:li:share:7430329696413843458`). See Figure 2 above for the live post screenshot.

5.4 Run 4: Agentic AI for Real-World Data Engineering (Fourth Test Case)

Input: Topic = "Agentic AI for Real-World Data Engineering — Design, Architecture, Process Flow, Safeguards & Standards." Mixed Technical + Leadership audience. Provocative + Educational tone. Personal anecdote about building an Azure data platform where engineers only used AI as chatbots until the team workshopped an agentic architecture.

Process: Full pipeline execution through Skills 0-8, run on Claude Code (Claude Opus 4.6).

Output: 6-week series saved to `archive/Agentic_AI_Data_Engineering_2026-02-21.md`. All 6 posts published live to LinkedIn via direct API (Option B2).

What Worked:

- The interview → roadmap → draft flow produced a cohesive 6-week arc on the first pass

- The war story (Week 3: "We Had Azure OpenAI. Nobody Used It Right.") was the strongest post — authentic, specific, and relatable
- The LinkedIn API publishing (Skill 8 fallback Option B2) worked flawlessly — all 6 posts published in under 15 seconds

What Didn't Work / HITL Revision:

- At HITL Checkpoint 5a, the reviewer requested that posts "expand on different ways teams can experiment with Agentic AI to illustrate they have options." The initial drafts presented agentic AI as a single approach rather than showing multiple entry points.
- This required revising Weeks 1, 2, 4, and 6 to add specific tooling options (platform-native, open-source, IDE-embedded, low-code approaches). Weeks 3 and 5 were left unchanged.
- After revision, all 6 posts were approved at HITL 5a (Round 2).

Benchmark Score: 4.35/5.0 — Weakest post was Week 5 (3.8), consistent with the pattern observed in all previous runs.

Changes After Run 4:

- Confirmed that Week 5 (Vision) is a systemic weak spot across all topics — this is now a documented known limitation with a proposed fix (add a trends research sub-step)
- Validated that the pipeline is portable across platforms: Runs 1-2 on Manus AI, Run 3 on Claude Code, Run 4 on Claude Code with a different model version

Concrete Example: Run 4 Input → Output

Input (Interview Answers):

Question	Answer
Q1: Audience	Mixed Technical + Leadership (Data Engineers, Architects, Directors, VPs, CDOs)
Q2: Core Message	Modern data teams that don't adopt agentic patterns will be outpaced by those that do
Q3: Anecdote	Built an Azure data platform with Azure OpenAI and Foundry available, but engineers only used them as chatbots until the team workshopped an agentic architecture
Q4: CTA	Audit workflows, start a pilot, rethink team structure, follow + comment
Q5: Tone	Provocative + Educational

Output (Week 3 — Final Post):

> *We had just finished building a new data platform in Azure.*
>
> *Azure OpenAI was available. Azure AI Foundry was configured. The tools were there. The budget was approved. On paper, we were an "AI-enabled" data team.*
>
> *In practice? Our data engineers used AI to search for syntax errors. To summarize documentation. To draft emails. Expensive chatbot usage.*
>
> *The turning point wasn't a new tool. It was a workshop.*
>
> *We pulled the team together and mapped our actual data workflows -- every manual step, every handoff, every bottleneck where an engineer was doing work that didn't require an engineer's judgment.*
>
> *Then we asked: "What if an agent owned this step?"*
>
> *Not "What if AI helped with this step." Owned it.*

> *That reframing changed everything. We architected an agentic solution where agents handled ingestion monitoring, schema drift detection, and data quality validation autonomously -- with human review only at decision points that genuinely required expertise.*
>
> *The result: significant acceleration across pipeline delivery, reduced manual intervention on routine tasks, and measurable improvement across our key KPIs.*
>
> *The tools didn't change. The architecture did.*
>
> *#AgenticAI #DataEngineering #AzureOpenAI #DataLeadership #AgenticDataSeries*

Publishing Log (Run 4) — All 6 posts published live to LinkedIn on 2026-02-21:

Wk	Title	Status
1	"Your Data Engineers Are Using AI Like It's 2023"	LIVE
2	"The Agentic Data Stack: Architecture That Actually Works"	LIVE
3	"We Had Azure OpenAI. Nobody Used It Right."	LIVE
4	"5 Safeguards Before Deploying AI Agents"	LIVE
5	"The 2027 Data Team: Fewer Tickets, More Architecture"	LIVE
6	"Stop Chatting With AI. Start Architecting With It."	LIVE

LinkedIn URNs: `urn:li:share:7431126731257958400` *through* `urn:li:share:7431126784642834432`*. Full URN list in* `archive/Agentic_AI_Data_Engineering_2026-02-21.md`*.*

Figure 3: Benchmark Scorecard — Run 4 (Agentic AI for Data Engineering)

Post	Actionability	Voice	Depth	Cohesion	LinkedIn	Avg
Week 1	3.5	4.5	4.0	4.5	4.5	**4.2**
Week 2	4.5	4.0	4.5	4.5	4.0	**4.3**
Week 3	4.0	5.0	4.5	5.0	4.5	**4.6**
Week 4	5.0	4.5	4.5	4.5	4.5	**4.6**
Week 5	3.0	4.0	3.5	4.5	4.0	**3.8**
Week 6	5.0	4.5	4.0	5.0	4.5	**4.6**
Avg	**4.2**	**4.4**	**4.2**	**4.7**	**4.3**	**4.35**

---

6. Benchmarking Methodology & Results

6.1 Methodology

Approach: Human Rubric Scoring (Method 1) + Baseline Comparison (Method 3)

Baseline: Single-prompt GPT-4 request: *"Write 6 LinkedIn posts about SQL Query Performance for Data Engineers and Business Stakeholders."*

5 Metrics (each scored 1-5):

1. Actionability — concrete next steps for the reader

2. Voice Consistency — sounds like the same leader across all 6 posts
3. Strategic Depth — genuine expertise, not surface-level
4. Narrative Cohesion — posts build on each other
5. LinkedIn Optimization — hooks, formatting, hashtags

Test Cases: 5 total — 3 standard (SQL Performance, Data Cleanliness, AI Chatbot to Dev Tool), 1 edge case (highly technical topic), 1 ambiguous case (vague input).

> ***Full scoring tables, baseline outputs, and failure analysis**: See `benchmark/BENCHMARK_APPENDIX.md`*

6.2 Summary Results

Figure 4: Aggregate Benchmark Results Across All Test Cases

Test Case	Agentic Score	Baseline Score	Delta
SQL Performance	**4.5**	2.0	+2.5
Data Cleanliness	**4.2**	—	—
AI Chatbot to Dev Tool	**4.5**	—	—
Edge: Technical Topic	**3.4**	—	—
Ambiguous: Vague Input	**3.6**	—	—
Agentic AI for Data Engineering	**4.35**	—	—

Biggest win: Narrative Cohesion — the baseline scored 1.0 (posts are disconnected), the agentic system averaged 4.8 across standard test cases (posts form a story). This is the single largest improvement and validates the core design decision of using a 6-week roadmap skill.

Worst failure: Edge case, Week 5 — Voice Consistency scored 2.0 because Skill 4 replaced precise technical terminology with executive language, which was inappropriate for database kernel engineers.

Cross-run pattern: Week 5 (Vision) is the weakest post in every single run. Across all 5 agentic test cases, Week 5 averages 3.8/5.0 compared to the series-wide average of 4.3/5.0. The root cause is that the "Vision" arc position relies on the LLM's training data for future predictions, which tends toward generic futurism. A proposed fix — adding a "trends research" sub-step that pulls recent industry articles before drafting Week 5 — would likely close this gap.

6.3 What the Benchmark Revealed

1. **The interview is everything**. Skill 1 (Intent Discovery) determines the ceiling for the entire series. Compare: the SQL Performance series (rich, specific input) scored 4.5/5.0, while the Ambiguous case ("making things better with data") scored only 3.6/5.0. Same pipeline, same skills — the only difference was input quality. This suggests that adding a "topic sharpening" step between Skills 1 and 2 could raise the floor for vague inputs.

2. **The "Vision" week (Week 5) is consistently the weakest**. Across all 5 agentic test cases, Week 5 averaged 3.8/5.0 — the lowest of any arc position. The Hook (Week 1) and Story (Week 3) positions consistently scored highest (4.4 and 4.7 respectively). The fix: add a trends research sub-step that pulls recent industry data before drafting Week 5.

3. **Voice refinement works well for general audiences, poorly for niche technical audiences**. The "Manager/Director" voice profile is effective for most use cases (Voice avg: 4.4 across standard runs) but scored only 2.5 on the WAL Protocol edge case — the worst single metric across the entire benchmark. The root cause: Skill 4's detection table replaces technical jargon with executive language, which is exactly wrong for kernel engineers.

4. **Two HITL checkpoints caught issues that automated scoring missed**. In Run 4 HITL 5a, the reviewer caught that the posts didn't show enough tooling options — a strategic gap that Skill 6's 5-metric rubric wouldn't flag

because each individual post was well-written. The HITL revision (adding platform-native, open-source, IDE-embedded, and low-code approaches) improved the series' practical value without changing any benchmark score. This validates the design decision to include human gates rather than relying solely on automated scoring.

5. **The pipeline improves with use**. Run 1 required prompt iteration on Skills 2 and 4. Run 2 required HITL revisions on 2/6 posts. Run 3 required zero revisions — all 6 posts approved on the first pass. This suggests the skills reach a stable quality threshold after 2-3 iterations, making subsequent runs faster and more predictable.

---

7. Reflection

What Worked Well

- **The pipeline structure is genuinely reusable.** After building the skills once, Run 2 took significantly less time than Run 1. The interview → roadmap → draft → refine → review flow works for any topic.
- **HITL checkpoints are essential.** Full automation would produce content the leader wouldn't publish. The checkpoints respect the leader's judgment while saving them 90% of the effort.
- **Anti-AI-ism rules in Skill 4 made the biggest difference.** The detection table approach is more effective than vague instructions like "sound natural."

What Didn't Work

- **Week 5 (Vision) is a consistent weak spot.** The system doesn't have access to real-time industry data, so the "future" week relies on the LLM's training data, which may be stale.
- **Voice Refiner is one-size-fits-all.** It needs audience-specific profiles (executive, technical expert, practitioner).
- **No multi-format output.** The system produces text only. A future version could generate Canva-ready visuals or video scripts.

How Prompts Evolved

The biggest learning was that **specific, table-based constraints outperform vague instructions**. "Sound professional" is useless. A table that says "replace X with Y" is actionable for the LLM. This applies broadly to any skill-based system.

Would I Keep Using This?

Yes — and I already am. I've now run this system 4 times across different topics (SQL Performance, Data Cleanliness, AI Chatbot to Dev Tool, Agentic AI for Data Engineering). Run 3 resulted in a live LinkedIn post (Week 1, URN: `urn:li:share:7430329696413843458`), and Run 4 resulted in all 6 posts published live. The system saves approximately 4-5 hours per 6-week series compared to writing from scratch, while producing content that's more strategically cohesive than what I'd write in scattered 30-minute sessions. The pipeline is genuinely reusable — each subsequent run takes less human time as the skills are already tuned.

---

8. How to Replicate

Prerequisites

- An LLM assistant that supports Markdown skills (Claude Code, Manus AI, or similar)
- (Optional) LiGo MCP server configured for LinkedIn publishing

Step-by-Step

1. **Clone the repository**:

```
git clone https://github.com/taashim-eng/linkedin-thought-leadership-agent.git
```

2. **Install the skills** in your LLM's skills directory:
   - Claude Code: Copy `skills/*.md` to `~/.claude/skills/linkedin-agent/`
   - Manus AI: Upload the skills via the platform interface
3. **Start a session** with this prompt:

   ```
   I want to create a 6-week LinkedIn thought leadership series on [YOUR TOPIC].
   Please use the LinkedIn Thought Leadership Orchestrator to guide me through the
   process.
   ```

4. **Complete the 5-question interview** (Skill 1 will ask you).
5. **Review the 6-week roadmap** (Skill 2 output).
6. **Wait for drafts** (Skills 3-5 run sequentially).
7. **Provide feedback at HITL 5a** — approve, revise, or reject each post.
8. **Review benchmark scores at HITL 6a** — approve or request changes.
9. **Find your final series** in the `archive/` directory.

MCP Setup (Optional — for auto-publishing)

Install the LinkedIn MCP package globally:

```
npm install -g @ldraney/mcp-linkedin
```

Then add to your LLM's MCP configuration (e.g., `~/.claude/settings.json`):

```
{
  "mcpServers": {
    "linkedin": {
      "command": "C:\\Users\\<username>\\AppData\\Roaming\\npm\\mcp-linkedin.cmd",
      "args": [],
      "env": {
        "LINKEDIN_CLIENT_ID": "<your-client-id>",
        "LINKEDIN_CLIENT_SECRET": "<your-client-secret>",
        "LINKEDIN_REDIRECT_URI": "http://localhost:3000/callback",
        "LINKEDIN_API_VERSION": "202510",
        "LINKEDIN_ACCESS_TOKEN": "<your-access-token>",
        "LINKEDIN_PERSON_ID": "<your-person-id>"
      }
    }
  }
}
```

Appendix A: Skill Prompts Reference

The full skill files are available in the GitHub repository at `skills/`. Below is a summary of each skill's core prompt logic and key instructions.

Skill 0: Master Orchestrator (`skill_0_master_orchestrator.md`)

Coordinates the 9-skill pipeline across 6 phases. Maintains session state (`strategic_intent`, `roadmap`, `drafts[]`, `feedback_log`, `benchmark_scores`). Enforces phase gates: Phase 1→2 requires all 5 interview answers; Phase 3→4 requires HITL 5a approval; Phase 5→6 requires HITL 6a approval. Includes error recovery (retry / skip / abort).

Skill 1: Intent Discovery (`skill_1_intent_discovery.md`)

Conducts a 5-question structured interview:

- Q1: Target Audience (role/seniority)
- Q2: Core Message (one thing readers should remember)
- Q3: Personal Anecdote (war story or real experience)
- Q4: Call to Action (what readers should do)
- Q5: Tone & Style (Provocative / Educational / Empathetic / Data-driven)

Outputs a Strategic Intent Document. Uses one clarifying follow-up per vague answer.

Skill 2: Content Strategist (`skill_2_content_strategist.md`)

Transforms the Strategic Intent into a 6-week roadmap following a fixed narrative arc:

```
Week 1: Hook (Pain Point) → Week 2: Framework (Mental Model) →
Week 3: Story (Personal Anecdote) → Week 4: Tactics (Concrete Advice) →
Week 5: Vision (Future Trends) → Week 6: Call (Synthesize + CTA)
```

Each week includes: Working Title, Objective, Key Content (2-3 bullets), "Coming Next Week" teaser.

Skill 3: Draft Architect (`skill_3_draft_architect.md`)

Generates 6 initial drafts (150-300 words) with structure: Hook (<200 chars) → Body (2-3 paragraphs) → Insight → CTA → Hashtags (3-5) → Teaser. Enforces anti-AI-ism rules: bans "In today's fast-paced world," "Let's dive in," "It's important to note," excessive hedging, and generic list openers.

Skill 4: Voice & Tone Refiner (`skill_4_voice_tone_refiner.md`)

Acts as ghostwriter. Four-step process: (1) Remove AI-isms via detection table, (2) Add professional weight ("Time-to-market" not "getting things done quickly"), (3) Match tone to specified style, (4) Ensure authority via active voice and first-person ("I," "we"). Core instruction: *"Do not change the core message or structure. Do not make posts longer."*

Skill 5: Engagement Optimizer (`skill_5_engagement_optimizer.md`)

Optimizes for LinkedIn algorithm: hooks <200 chars, short paragraphs, line breaks, max 5 hashtags. Adds visual recommendations per week (conceptual for W1-2, photo for W3, infographic for W4, futuristic for W5, branded for W6). **HITL Checkpoint 5a**: PAUSE — present all 6 posts, collect Approve/Revise/Reject per post. Route revisions: voice → Skill 4, structure → Skill 3, formatting → Skill 5.

Skill 6: Quality Reviewer (`skill_6_quality_reviewer.md`)

Scores 5 metrics (1-5 scale): Actionability, Voice Consistency, Strategic Depth, Narrative Cohesion, LinkedIn Optimization. Calculates per-post and aggregate averages. Flags any post scoring below 3.0. **HITL Checkpoint 6a**: PAUSE — present scorecard, user approves or requests changes.

Skill 7: Archive Manager (`skill_7_archive_manager.md`)

Packages session into `archive/[Topic_Slug]_[YYYY-MM-DD].md` containing: Strategic Intent, Roadmap, all 6 Final Posts, Benchmark Results, User Feedback Log, Metadata. Generates PDF via `md-to-pdf`. Saves exactly what was approved — no modifications.

Skill 8: Poster & Reviewer (`skill_8_poster_reviewer.md`)

Presents 3 publishing options: **Option A** (manual copy-paste, recommended), **Option B1** (MCP server via `@ldraney/mcp-linkedin`), **Option B2** (direct LinkedIn API via Node.js HTTPS POST). Never auto-posts without user confirmation. Includes post-publication verification and URN logging.

Total word count: ~5,800 words | Estimated reading time: 24 minutes