Do Not Start an Agentic AI Project Before Reading This

> This is a monologue from Salil Shekharan, Partner, outlining a critical distinction that will determine whether we build true Agentic AI or just automation with AI labels.

As we continue building our Agentic AI capabilities, I want to clarify an important architectural distinction that will fundamentally determine whether we build transformational systems or simply traditional automation wrapped with AI.

There is a growing industry pattern where teams take an existing workflow, insert LLM calls into a few steps, orchestrate prompts and APIs, and label the outcome as “Agentic AI.”

That is not Agentic AI.

It is important that we align on this early because the engineering mindset, architecture, operating model, testing strategy, and production risks are fundamentally different.

Traditional software and automation systems are built around predefined execution paths:

Input ? Rules ? Output

The developer defines:

- every workflow
- every branch
- every condition
- every exception path

Even when LLMs are added for extraction, summarization, or classification, the system is still fundamentally deterministic. The intelligence is assisting the workflow — not driving it.

True Agentic AI is fundamentally different.

Agentic systems are goal-driven, not workflow-driven.

The system should be capable of:

- dynamically planning execution paths

- reasoning about ambiguity
- selecting tools contextually
- adapting to failures
- maintaining memory and state
- evaluating outcomes
- self-correcting when needed
- escalating intelligently when confidence is low

The most important shift is this:

Traditional systems execute predefined logic.
Agentic systems create and adapt logic at runtime to achieve goals.

If the entire execution path can be fully drawn upfront as a fixed workflow, then we are most likely building automation - “not an agent.”

A useful analogy:

Traditional software is like a train running on fixed tracks.
Automation + LLM is a smarter train with better announcements.
Agentic AI is autonomous navigation where the destination is defined, but the path is dynamically determined based on context, obstacles, priorities, and outcomes.

This means our engineering approach must evolve beyond:

- prompt chaining
- static orchestration
- workflow-centric thinking
- deterministic branching models

Instead, we must design around:

- goals and intent
- reasoning loops
- planning and replanning
- memory architectures
- tool ecosystems
- reflection and validation
- bounded autonomy
- human-in-the-loop governance
- observability and evaluation

This is not just a technology shift. It is a **systems engineering** shift.

The success metric is no longer:
“Did the software execute correctly?”

The success metric becomes:
“Did the system achieve the business outcome safely, reliably, and adaptively under uncertainty?”

Design Clarity is non-negotiable:

- We will not build systems where the design is unclear.
- If you are not fully clear on the architecture, execution model, and agent behavior, do not start building.

Pause, engage with the architecture leads, and resolve the ambiguity first.

- The fundamental design failures that will break at scale.

Every system must have:

- clearly defined agent roles and boundaries
- explicit control flows (including failure and exit conditions)
- well-structured tool invocation patterns
- auditable reasoning and execution paths

Use AI-assisted code review as a baseline - not as an afterthought.

- Validate your logic, interaction patterns, and edge cases proactively before integration.
- Do not build first and figure it out later.
- Get the design right, validate it, and then execute.

As we move forward, I encourage teams to challenge designs that resemble:

- hardcoded workflows with prompts
- deterministic orchestration disguised as agents
- excessive branching logic around LLM calls
- agents that cannot operate outside predefined paths

The goal is not to build smarter workflows.

The goal is to build systems capable of reasoning, adapting, and operating toward outcomes within controlled boundaries.

This distinction will define the maturity, scalability, and long-term differentiation of our platform and solutions.

“This distinction will define whether we lead - or fail to differentiate.”

Revision #1
Created 2026-06-18 04:11:10 UTC by Soubhik Mazumdar
Updated 2026-06-18 04:15:39 UTC by Soubhik Mazumdar