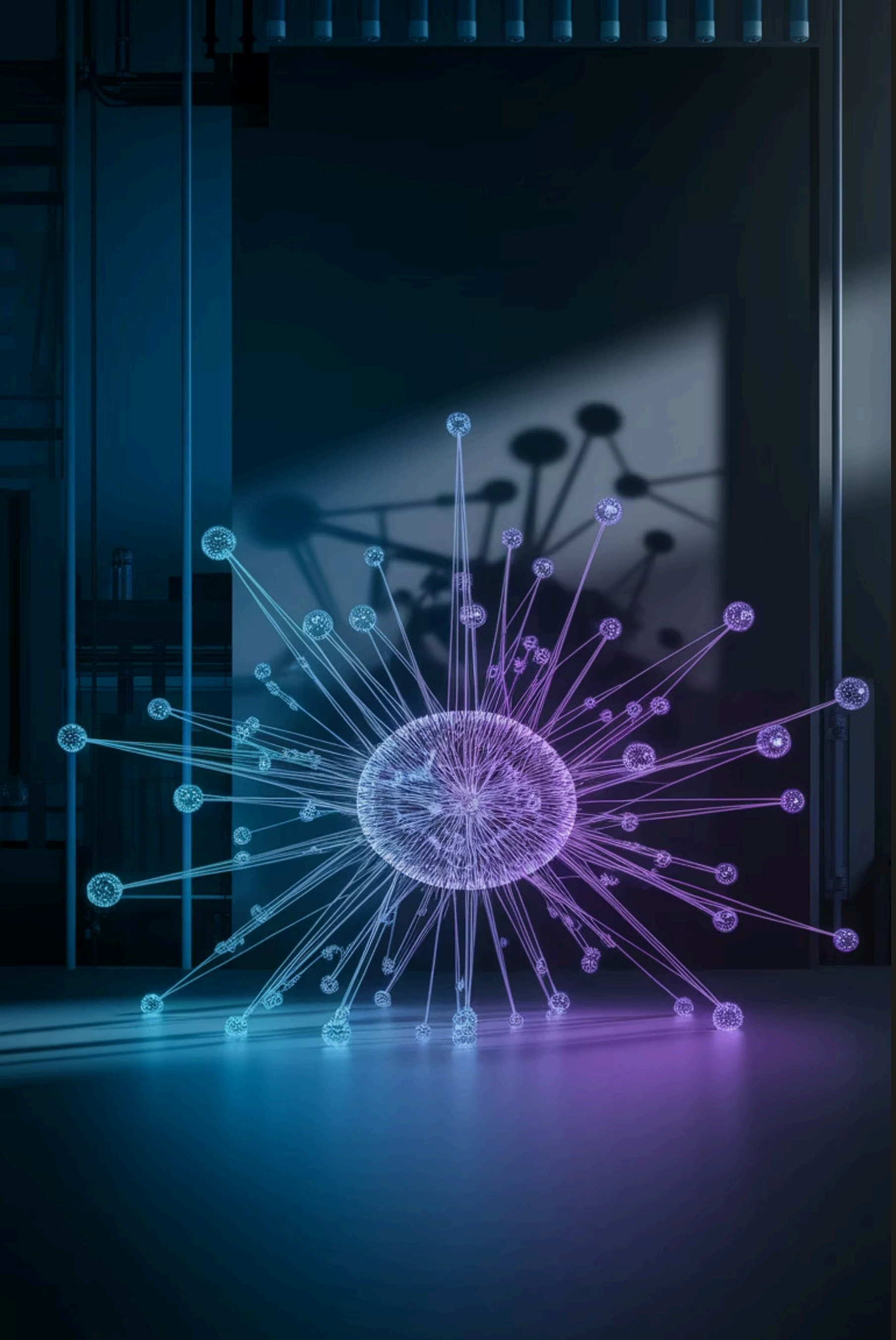


Decoding the GenAI Revolution

Welcome to our exploration of how GenAI is transforming software development. We'll examine practical shifts and application impacts that directly affect our work as engineers.

by Martin Rojas

The GenAI Tsunami

Unprecedented Growth

75% enterprise adoption in 2024, with constant innovation from major players.

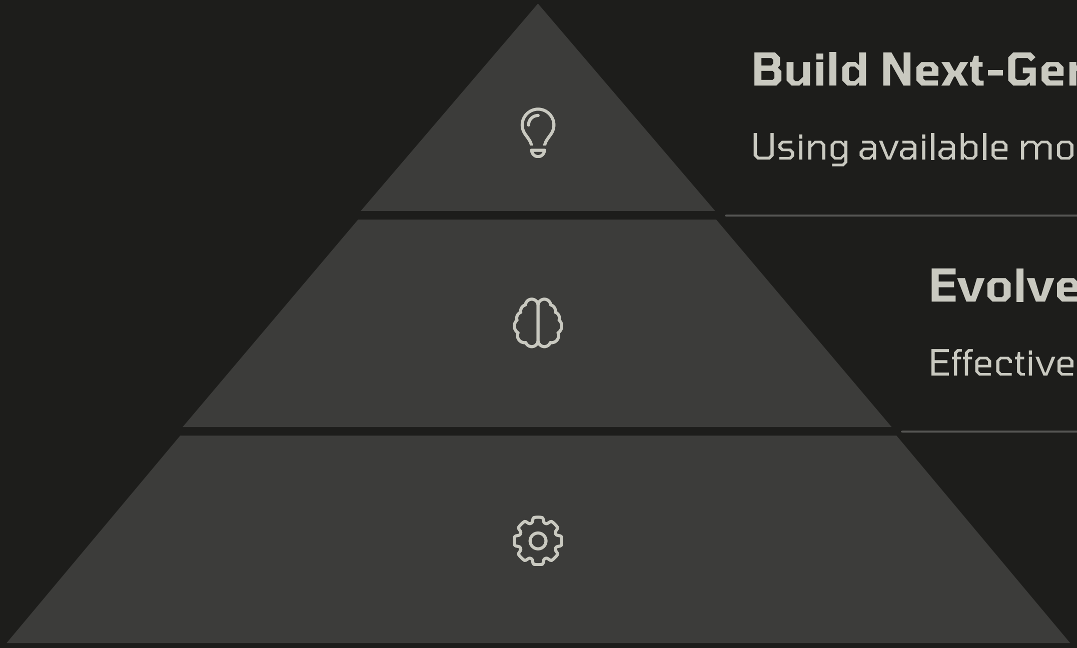
Developer Impact

Shifting from experimentation to practical integration in our applications.

Productivity Potential

20-50% speed increase cited in development workflows.

The Opportunity & Challenge



Build Next-Gen Apps

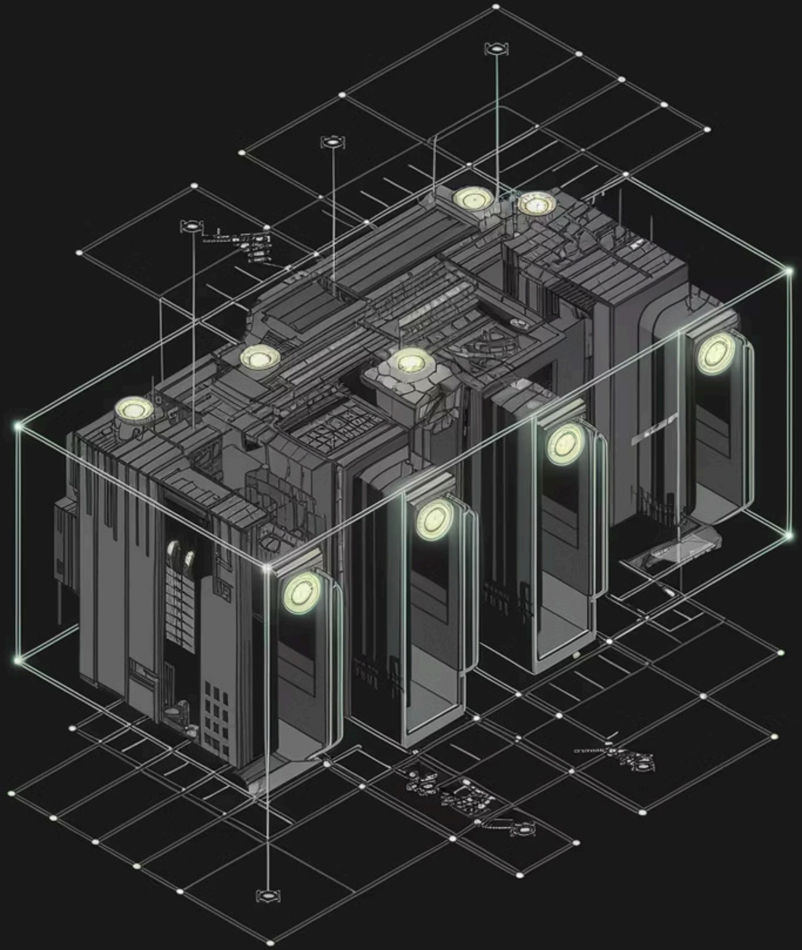
Using available models and frameworks

Evolve Our Skillsets

Effective prompting and integration strategies

Understand Technical Shifts

And their impact on our applications



Architectural & Scaling Insights

Transformer Basics

Multi-Head Attention allows models to grasp complex relationships in data.

Sparse Attention

Enables longer context windows efficiently, making models like Claude 3.5 and Gemini 1.5 Pro feasible.

Scaling Laws

Balance is key. A moderately sized model trained on more data can outperform larger models.

Why Models Behave Differently

Alignment (RLHF)

Makes models helpful and safe

Instruction Tuning

Enables following commands out-of-box

Fine-Tuning

Specializes for your domain knowledge

Unlocking Reasoning Capabilities

Challenge

Models struggle with multi-step logic

Chain-of-Thought

Guide models to outline steps

Tool Use Frameworks

Allow models to call external tools

Better Results

Improved accuracy on complex tasks

Chain-of-Thought Deep Dive

What is CoT?

A prompting strategy that makes LLMs show their work by outputting intermediate reasoning steps.

Benefits

- Better accuracy on complex tasks
- More transparency in reasoning
- Handles multi-step problems

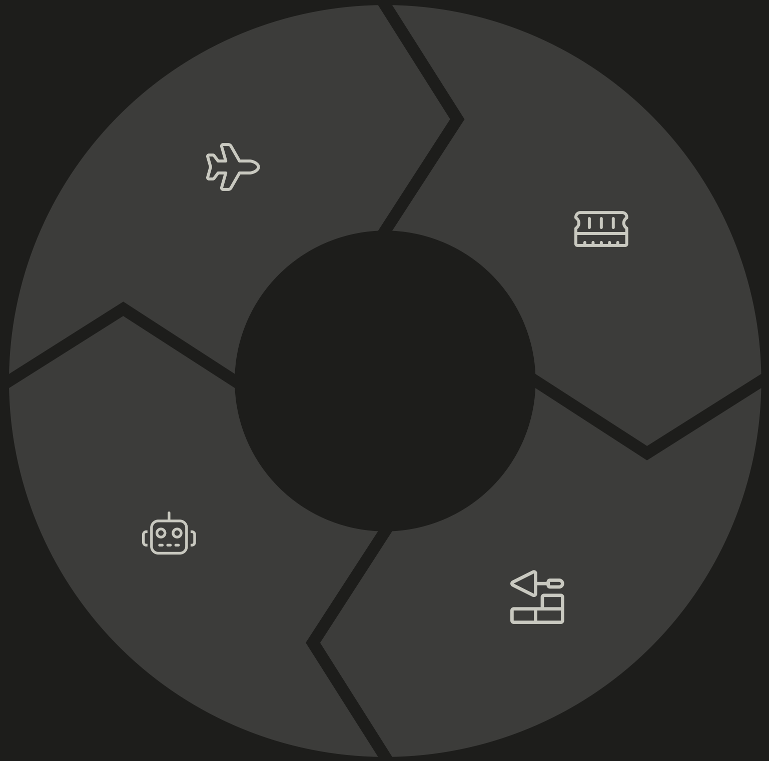
How to Use CoT

- Zero-Shot: Add "Let's think step by step."
- Few-Shot: Provide examples of step-by-step format

Limitations

- Higher cost and latency
- Hallucinations still possible
- Works best with capable models

AI Agents & Workflow Automation



Plan

Break down goals into steps

Remember

Store context via vector DBs

Act

Execute tasks autonomously

Use Tools

Interact with APIs and databases

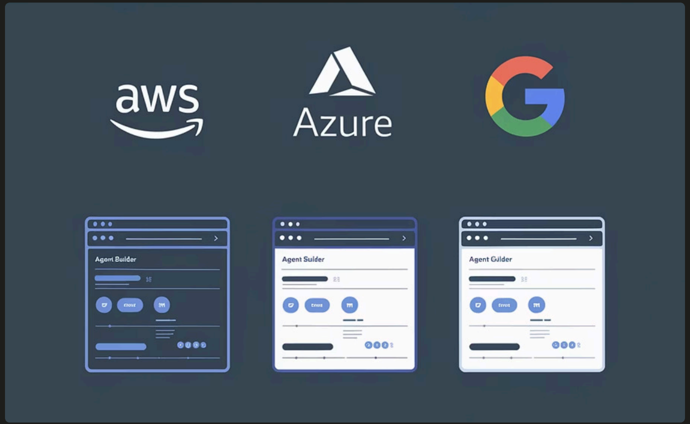
Frameworks for Building Agents

LangChain

Popular, modular framework in Python/TS. Provides building blocks to connect LLMs, memory, and tools.

Semantic Kernel

Microsoft's framework integrating LLMs with conventional code. Focuses on orchestrating plugins.



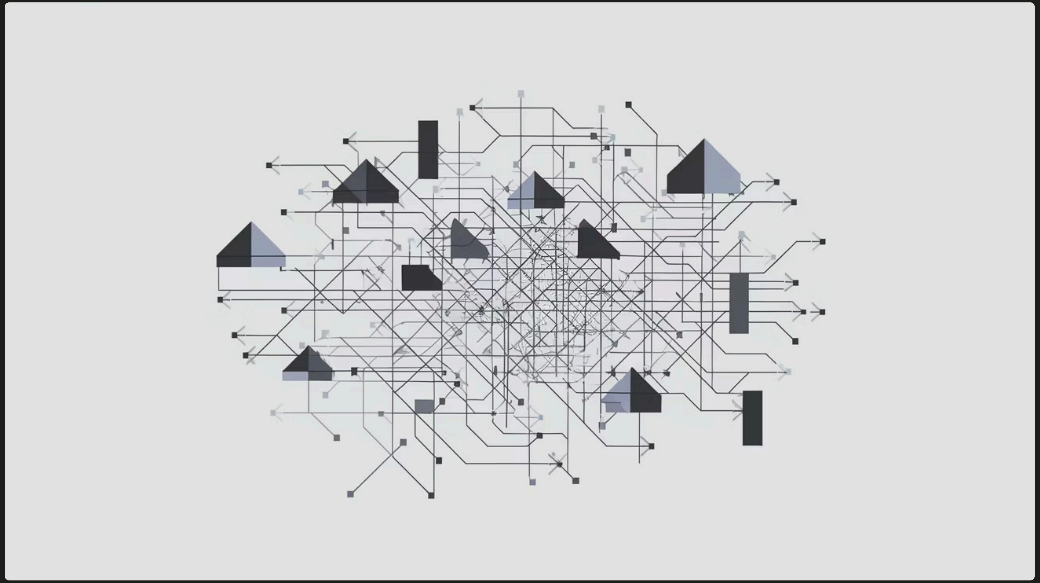
Cloud Platforms

AWS Bedrock, Azure AI, and Google Vertex AI offer managed agent-building services.

Open Standards: MCP & A2A

The Problem

Connecting every AI model/agent to every tool/API leads to integration chaos (M×N problem).



MCP (Model Context Protocol)

"USB-C for AI" - Standard way for applications to talk to any tool or data source.

A2A (Agent-to-Agent Protocol)

Enables different AI agents to discover each other and collaborate on tasks.

Extended Context Windows

Massive Input Capacity

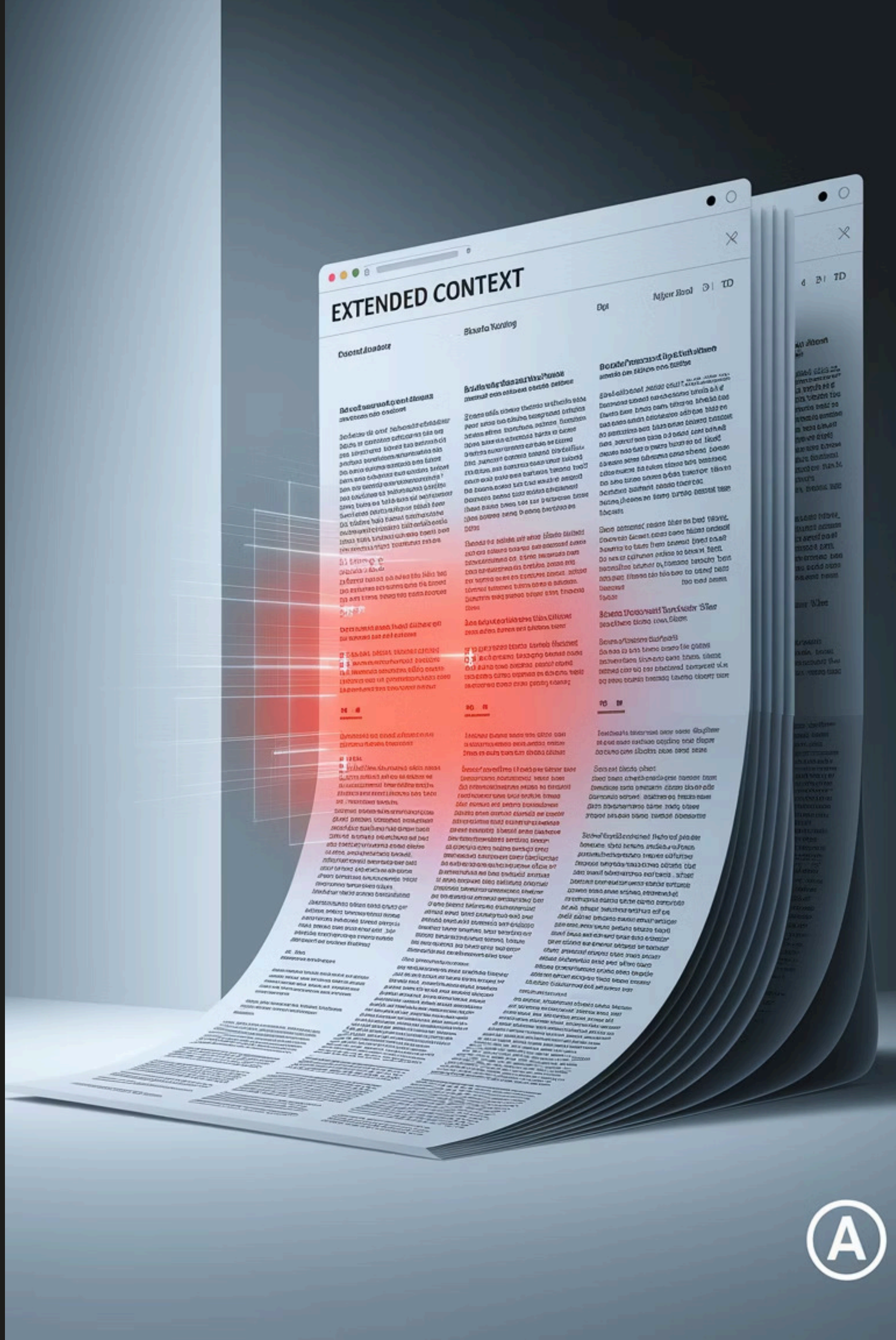
Models like Gemini 1.5 Pro and Claude 3.5 can process 1M+ tokens.

Developer Use Cases

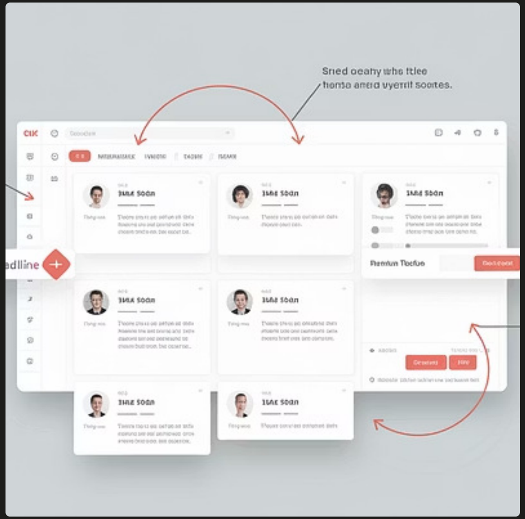
Analyze entire codebases, summarize long reports, maintain chat history.

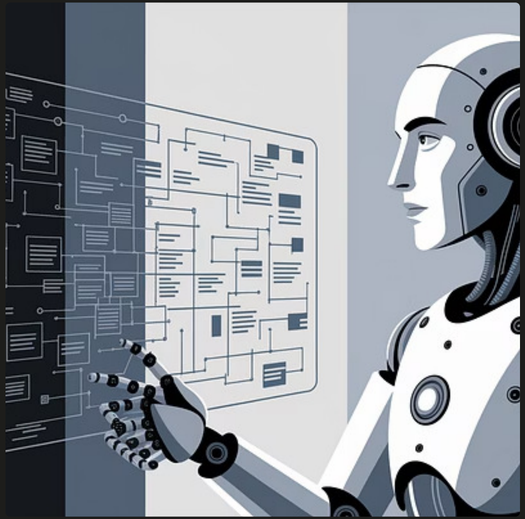
Trade-offs

Higher cost, increased latency, and potential "lost in the middle" issues.



Multimodal Models





Models like GPT-4o, Claude 3.5, and Gemini can now understand images, audio, and sometimes video, enabling richer, more intuitive applications.

Designing Explainable Agent UIs

Show Reasoning

Display the agent's thought process and plan to build user trust.

Indicate Tool Usage

Make it clear when external tools or APIs are being used.

Provide Controls

Allow users to approve or reject actions before execution.

Ex|laing its reasoning process.
The itmerd ειο rene bodecdess uprrove rgeertie txt moceuer peneoel lonse andl lme oneece rodere couise eeoprte aine ens themeomot eh r roe nence. com pне onerdme stand andl eesiign.
PREVIOUS
NEXT
APPROVE

Managing AI Applications

Performance

Implement model routing, response streaming, and caching to manage latency.

Guardrails

Add input/output filtering, security checks, and rate limiting for safety.

Testing

Develop evaluation suites with test prompts and validation logic.

Monitoring

Track costs, quality metrics, and watch for bias or drift in production.

Key Takeaways

1

GenAI as a Platform

Build by composing available models, prompts, agents, and tools.

2

Responsible Integration

Prioritize reliability, security, ethics, and user trust.

3

Experiment & Evaluate

Test different models and techniques for your specific use case.

4

Leverage the Ecosystem

Utilize frameworks, standards, and community knowledge.

Atlanta Cloud Conference Sponsors

PLATINUM SPONSOR

KENNESAW STATE
UNIVERSITY
COLLEGE OF COMPUTING AND
SOFTWARE ENGINEERING

GOLD SPONSORS

<epam>

SILVER SPONSORS

MacStadium

MOTION
RECRUITMENT