Context Engineering

31 July 2025

https://arxiv.org/pdf/2507.13334

Paper Goals & Approach

- Formalize ‘context engineering’ as a discipline
- Meta-survey: >1,400 papers (2020–2025)

Why This Paper?

- **LLM performance and reliability are driven by the *context*** they see at inference. Poor context → hallucinations, cost, latency.
- **Proven benefits from better structured context**
  e.g., Chain-of-Thought ↑ math accuracy 17% → 79%
- **Prompt engineering is no longer enough**—modern AI requires dynamic, multifaceted information streams, not static prompts.

Prompt engineering has been great for demos, but Context Engineering is needed for production systems

What is Context Engineering?

Context Engineering is the design, assembly, and optimization of the entire information payload provided to an LLM.

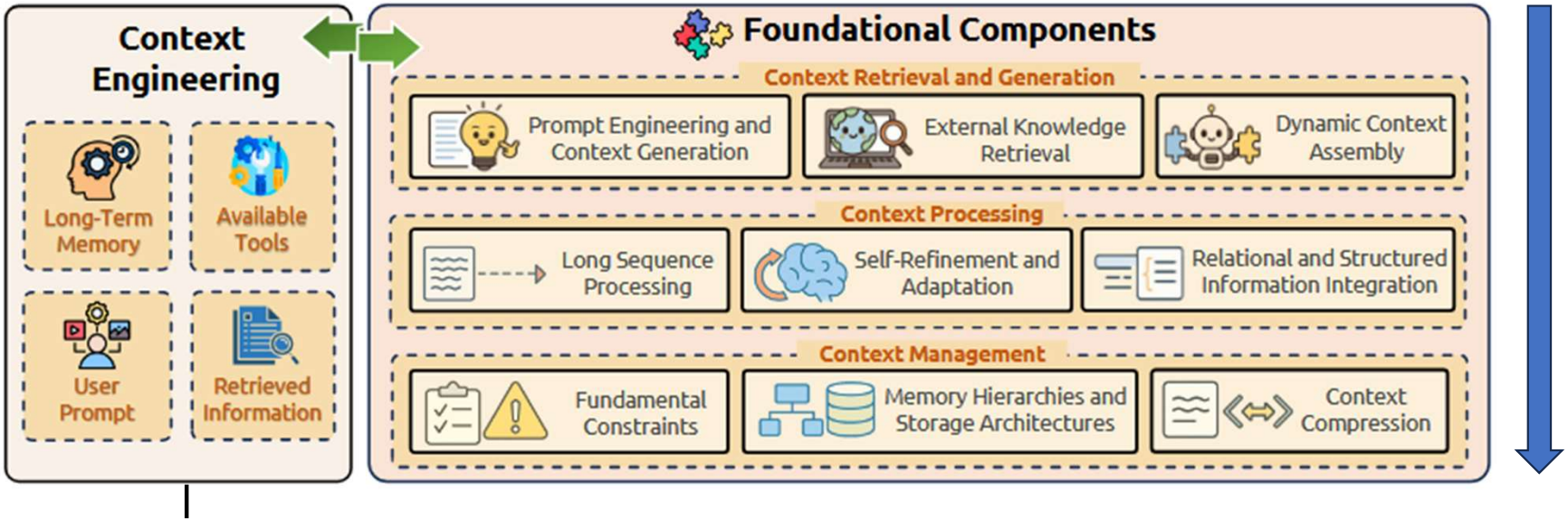
It optimizes a pipeline of context functions, with constraints on context length and processing.

Components of Context: (the payload)

- Instructions
- Retrieved knowledge (often via RAG)
- Tool signatures (for function calling/reasoning)
- Persistent memory
- Dynamic state (e.g. in multi-agent settings)
- User queries

Taxonomy and Systematization

Foundational Components – The Context Pipeline



Context Engineering Implementations:

- *Retrieval-Augmented Generation (RAG):* Modular, agentic, graph-enhanced types.
- *Memory Systems:* Explicit long-term/short-term memory, memory-augmented agents (e.g., MemGPT).
- *Tool-Integrated Reasoning:* Function calling, tool use, orchestration frameworks.
- *Multi-Agent Systems:* Orchestration, protocols, coordination, communication frameworks.

Thoughts and discussion…

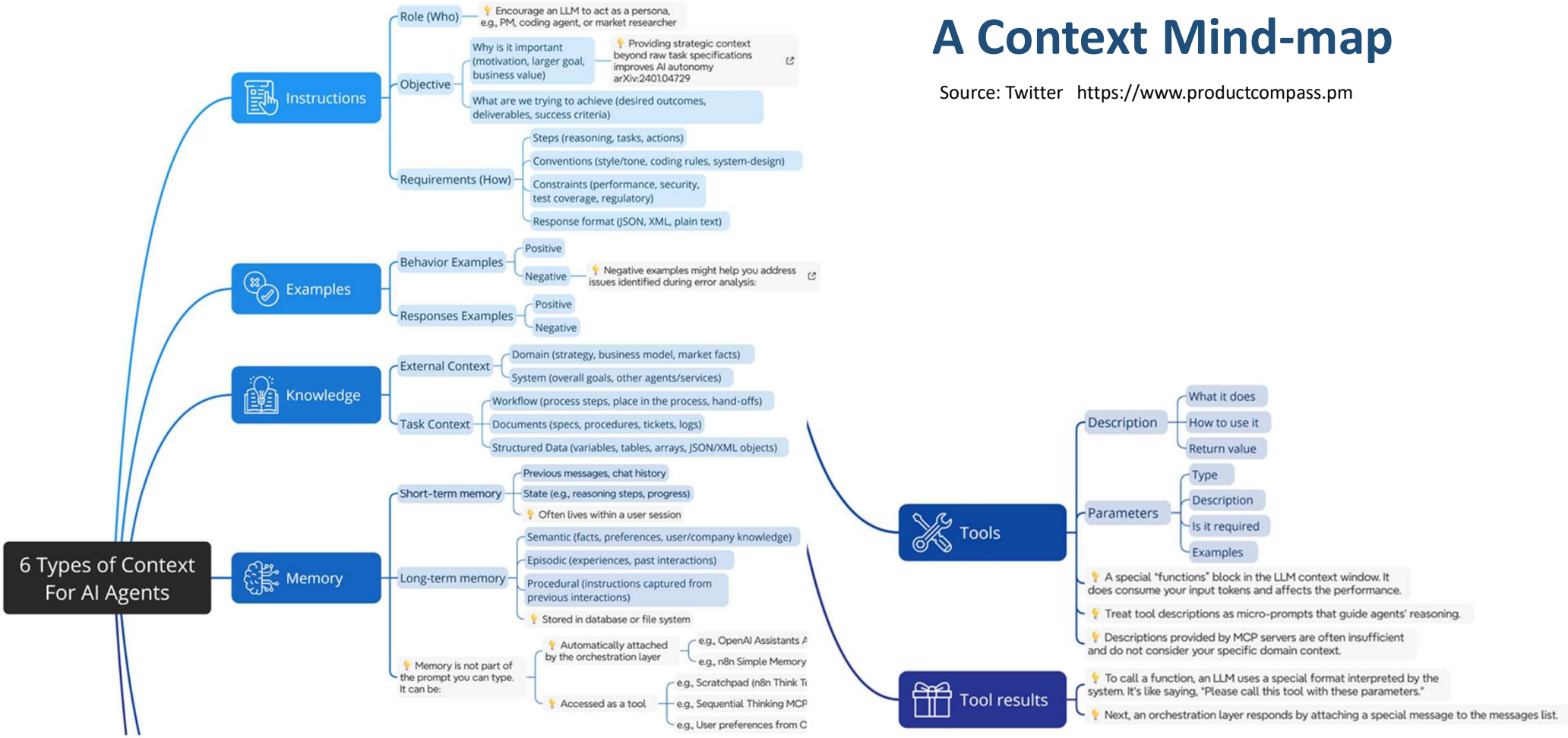
- Context Engineering is to LLM applications what Feature Engineering is to classical Machine Learning
  —a foundational, discipline-defining lever for next-generation model performance.

- How to improve?
  - Audit your largest RAG chain
  - Measure tokens in/out, record latencies
  - Version manage your entire *context functions*, not just your prompts

- Output structuring is the new bottleneck
  Context engineering dramatically enhances LLM performance, but current models are much better at understanding complex context than generating equally good long-form output.

A Context Mind-map

Source: Twitter https://www.productcompass.pm



Diagrams from the Paper

The taxonomy of Context Engineering in Large Language Models

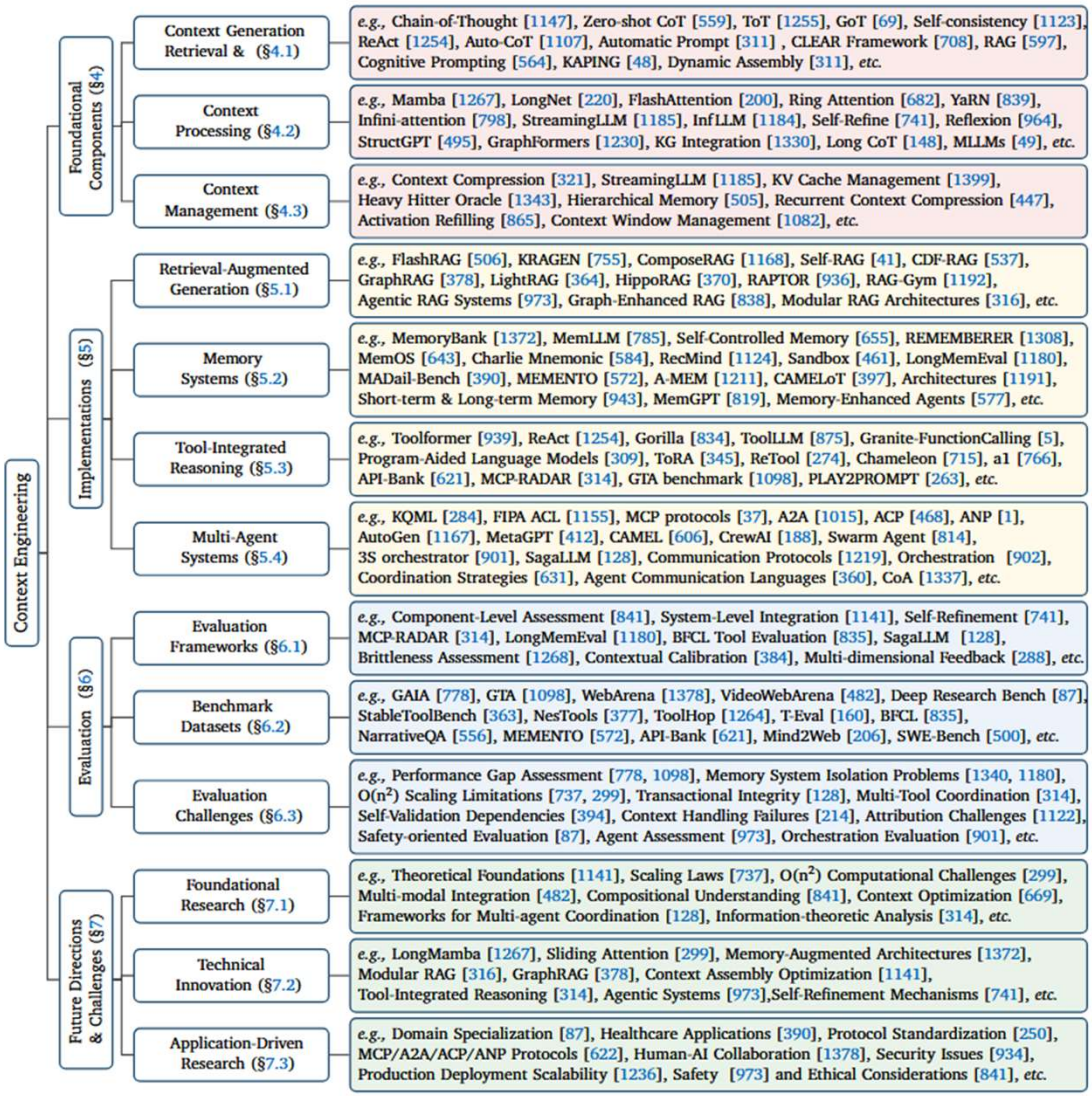
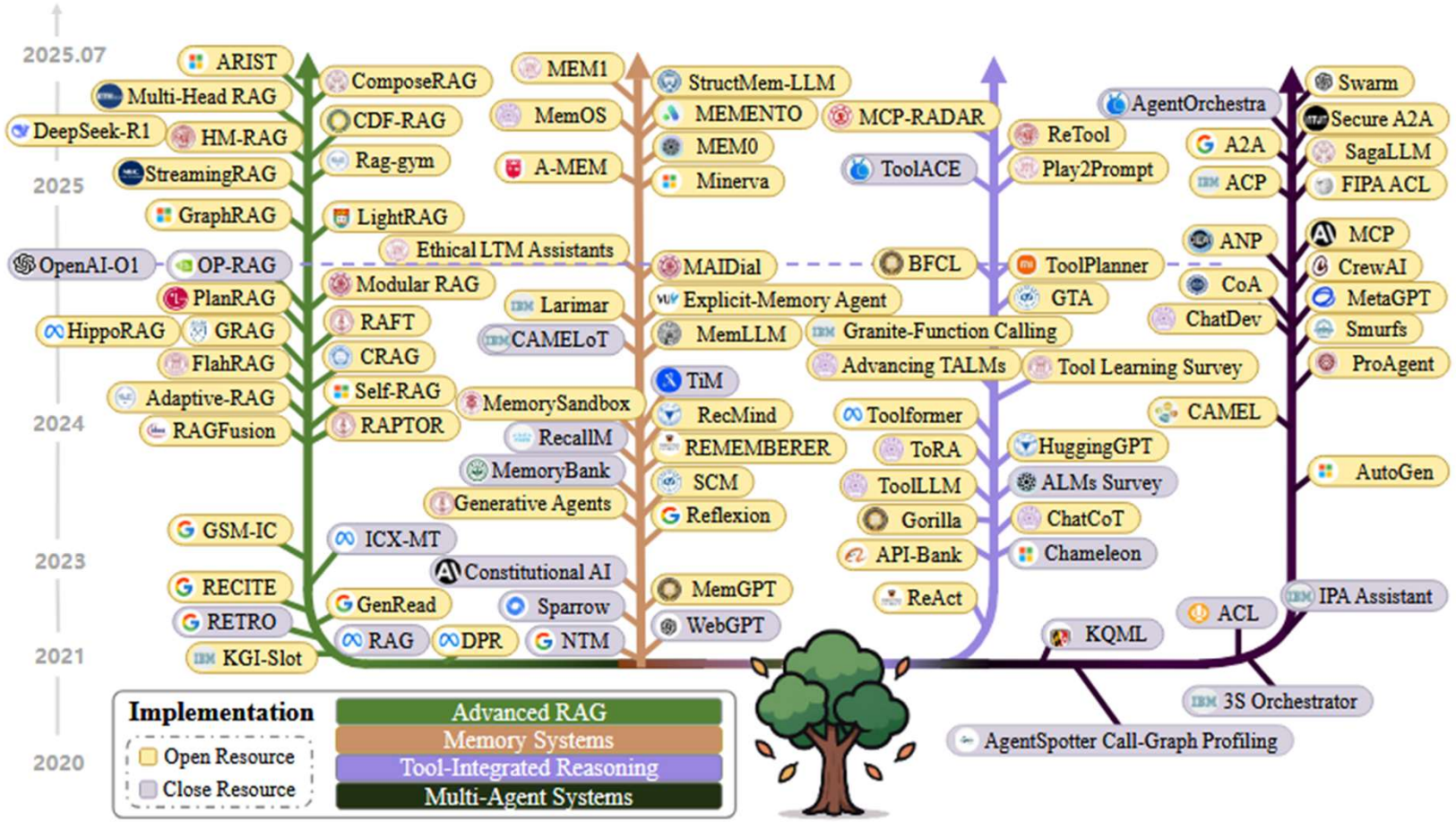
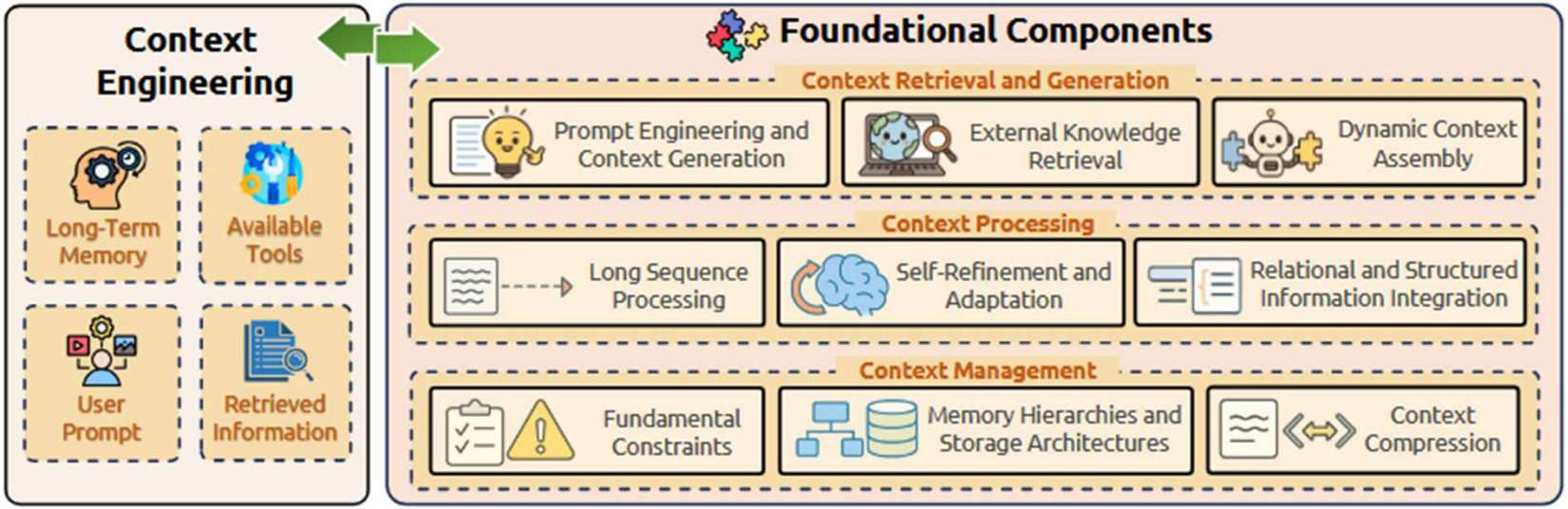
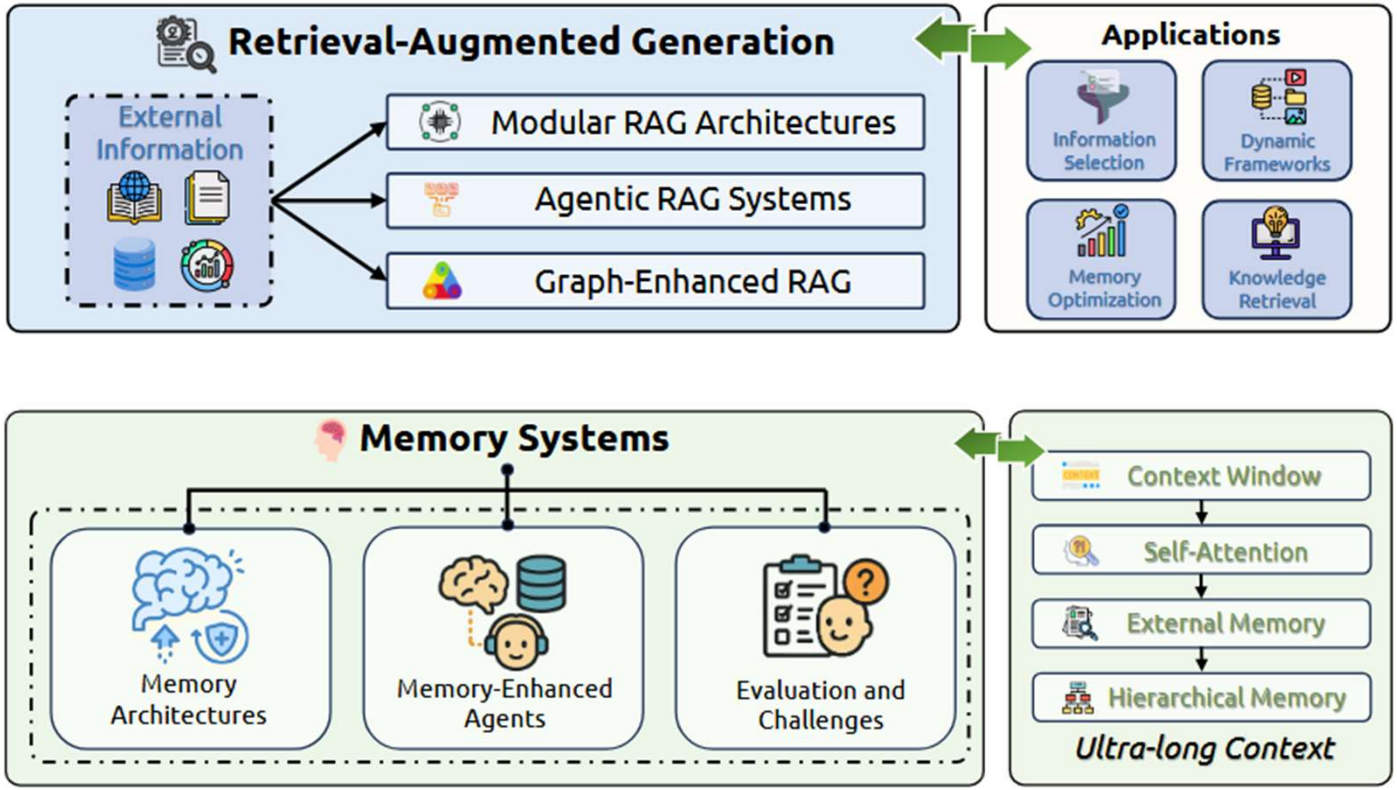
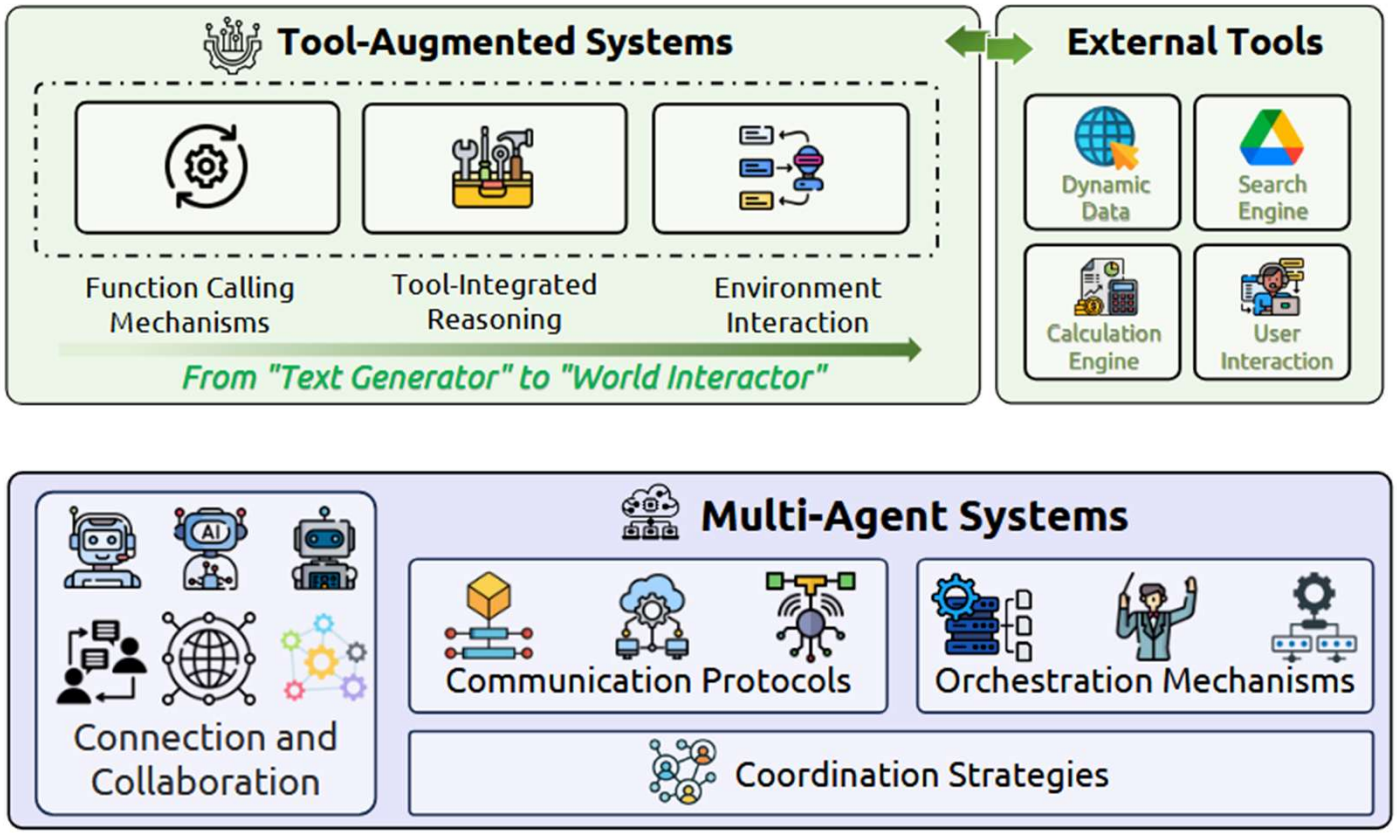


Figure 2: Context Engineering Evolution Timeline:




Context Engineering
Long-Term Memory
Available Tools
User Prompt
Retrieved Information
Foundational Components
Context Retrieval and Generation
Prompt Engineering and Context Generation
External Knowledge Retrieval
Dynamic Context Assembly
Context Processing
Long Sequence Processing
Self-Refinement and Adaptation
Relational and Structured Information Integration
Context Management
Fundamental Constraints
Memory Hierarchies and Storage Architectures
Context Compression


Retrieval-Augmented Generation
External Information
Modular RAG Architectures
Agentic RAG Systems
Graph-Enhanced RAG
Applications
Information Selection
Dynamic Frameworks
Memory Optimization
Knowledge Retrieval
Memory Systems
Memory Architectures
Memory-Enhanced Agents
Evaluation and Challenges
Context Window
Self-Attention
External Memory
Hierarchical Memory
Ultra-long Context


Tool-Augmented Systems
External Tools
Function Calling Mechanisms
Tool-Integrated Reasoning
Environment Interaction
From "Text Generator" to "World Interactor"
Dynamic Data
Search Engine
Calculation Engine
User Interaction
Multi-Agent Systems
Connection and Collaboration
Communication Protocols
Orchestration Mechanisms
Coordination Strategies