Hyper-Aether: A Vision for AI-Native Computing Based on VM Fabric

Hirofumi Inomata

Abstract

We propose Hyper-Aether, a vision for AI-native computing in which software systems are constructed as dynamically generated execution structures.

At the core of Hyper-Aether is VM Fabric, an execution substrate that organizes AI-generated virtual machines into a structured graph.

Instead of writing programs, users provide specifications, and the system generates both computation and system topology.

This work outlines the architecture, its components, and expected benefits.

Keywords: AI-native computing, virtual machines, hypervisor, system generation, distributed systems, software architecture, VM Fabric

1 Introduction

Modern software development requires manual programming and system design.

Hyper-Aether replaces this process with specification-driven generation.

In this model, systems are constructed dynamically rather than written manually.

2 Motivation

Current systems suffer from:

- high development cost: significant human effort is required for implementation
- rigid architectures: system structure is fixed and difficult to change
- manual system design: developers must explicitly design system topology

Hyper-Aether aims to automate both computation and structure.

3 Hyper-Aether Architecture

Hyper-Aether consists of several components:

- AI Generator: generates virtual machines from user specifications
- VM Fabric: organizes generated VMs into a connected structure
- Hypervisor: executes and isolates all virtual machines
- Hardware: provides physical computing resources

VM Fabric is the core execution layer within this architecture.

4 VM Fabric

We define VM Fabric as:

$$= (V, E)$$

where

- V is the set of virtual machines generated by AI
- E represents communication relationships between VMs

Each VM is generated as:

$$v_i = G(spec_i)$$

where

- $spec_i$ is a user specification
- G is the AI generator

This model represents computation as a graph of interacting components.

5 Execution Model

Execution is defined as:

$$Exec = \sum_i (v_i)$$

where

- is the hypervisor execution function
- v_i is a virtual machine

This means all computation is uniformly executed by the hypervisor.

6 Comparison with Conventional Systems

Figure 1: Comparison of execution models. Conventional systems rely on manually written programs, while Hyper-Aether generates execution structures dynamically.

Unlike conventional systems, which execute a single program, Hyper-Aether constructs a system dynamically.

7 Example Scenario

Figure 2: Example VM Fabric for a web application. Each component is represented as a VM.

In this example:

- UI VM: handles user interface interactions
- API VM: processes application logic
- DB VM: stores persistent data
- Auth VM: manages authentication

The system structure is generated automatically.

8 Advantages

VM Fabric provides:

- dynamic structure: systems can change over time
- modularity: each VM represents a separate component
- isolation: failures are contained within individual VMs
- reduced human effort: manual programming is minimized

9 Limitations

This work has limitations:

- no implementation yet: the system is conceptual
- AI dependency: performance depends on generation quality
- performance overhead: virtualization introduces additional cost

10 Evaluation Plan

Future evaluation will measure:

- development cost reduction
- execution performance
- scalability of VM Fabric

11 Discussion

Hyper-Aether changes the definition of computing:

$$Program = Specification$$

This shifts development from coding to describing intent.

12 Conclusion

We presented Hyper-Aether, an AI-native architecture based on VM Fabric.

This approach replaces programs with dynamically generated structures, potentially reducing development cost and improving flexibility.