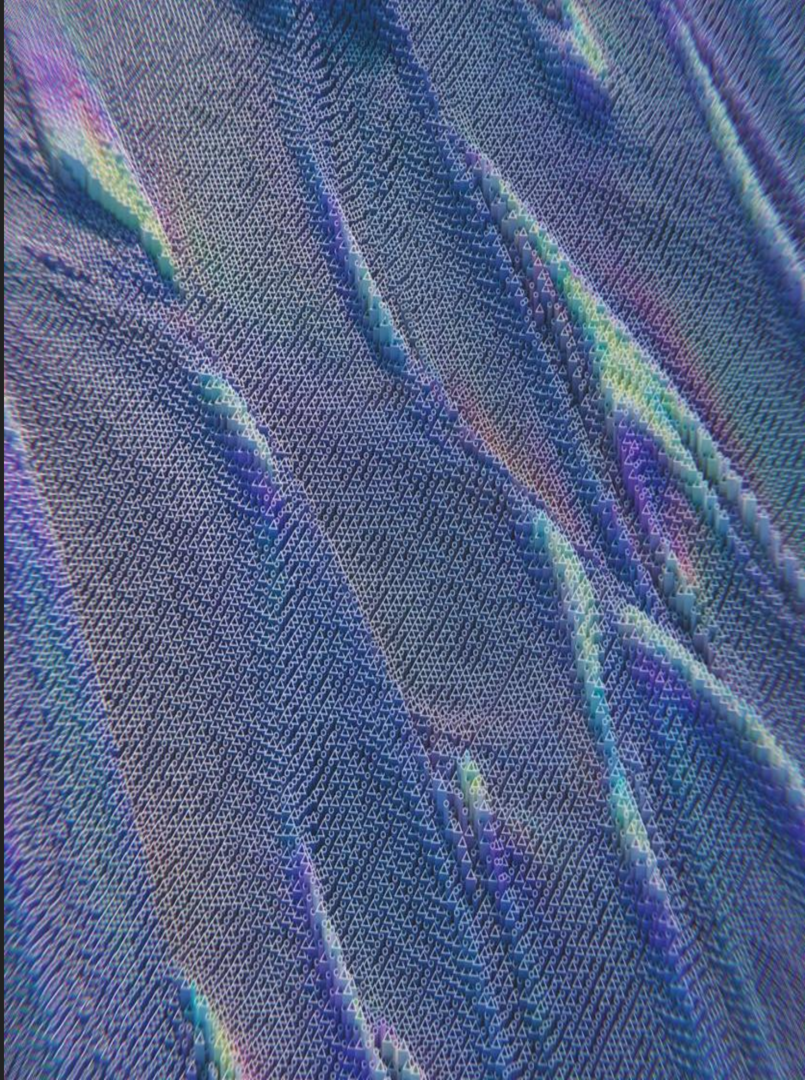
FILUTA

Graph-Based Orchestration of Heterogeneous AI Models: A Control Node Approach to Composite Intelligence

By G. Michael Youngblood, Filip Dvořák, Slavo Švancár, Tomáš Balyo, Michal Ficek, and Martin Dousek

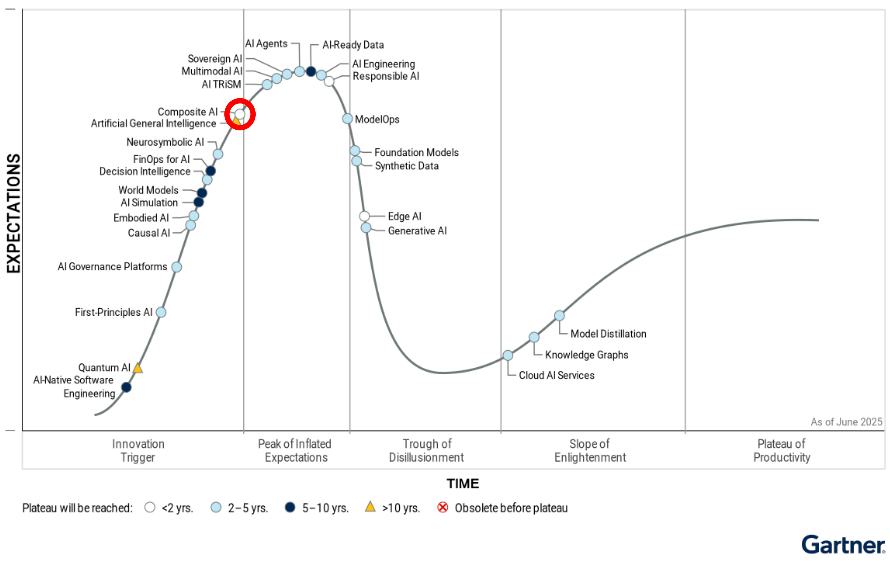
IJCAI 2025 Composite AI Workshop (W23 CompAI)



01/Why Composite AI?

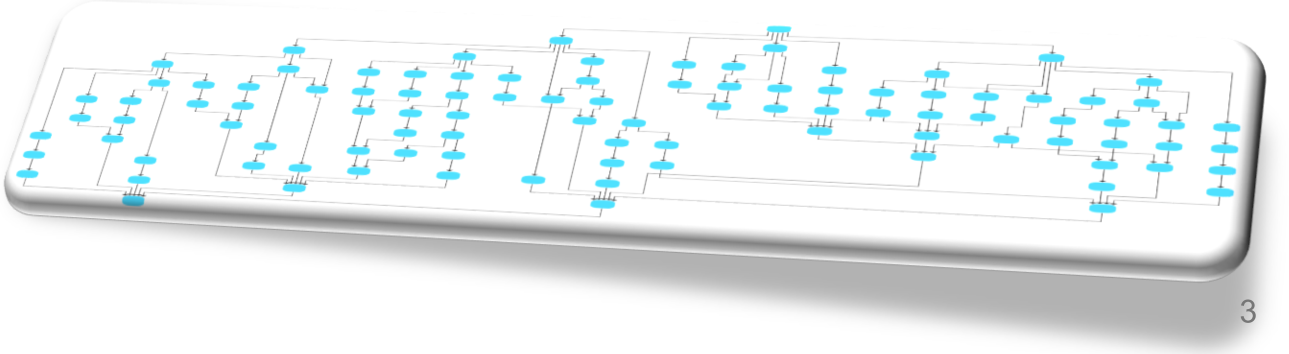
- Real-world problems need multiple AI/ML models—no single approach is universally optimal
- Composite AI: combining models for efficient problem-solving
- Gartner (2025): Composite AI is a key trend after deep learning's practical limitations

Hype Cycle for Artificial Intelligence, 2025



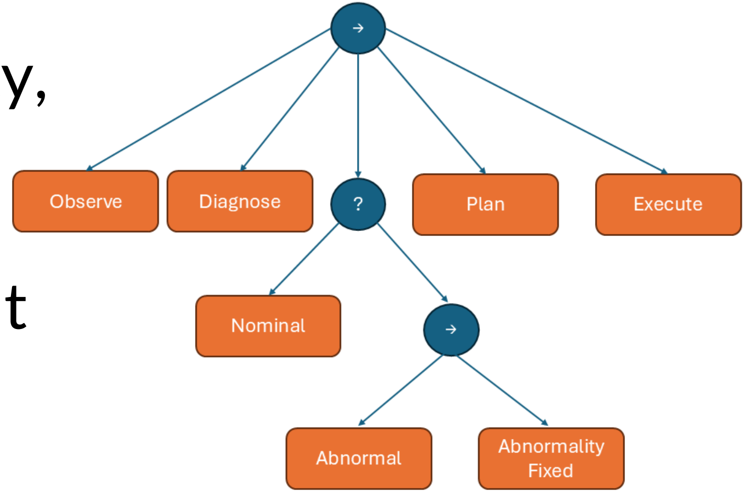
02/Our Approach

- Introduce a framework for integrating diverse AI models
- Use graph structures
  - Directed Acyclic Graphs (DAGs) and
  - Behavior Trees (BTs) for orchestration
- Key invention:
  The logic-driven “control node” for model connection.



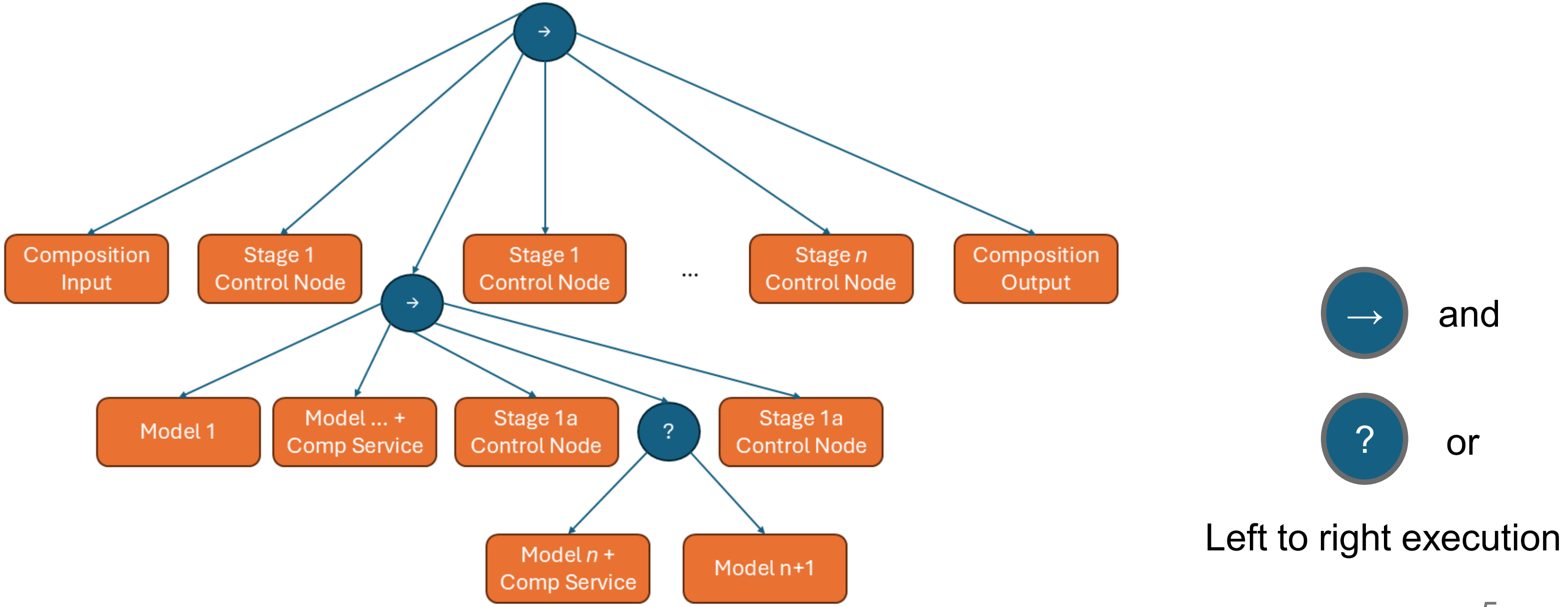
03/The Power of Graphs in AI Orchestration

- Graphs capture relationships between model components (nodes/edges)
- Advantages: Visualization, scalability, modular design, and dynamic adaptability
- Foundation for system management and control



04/Behavior Trees

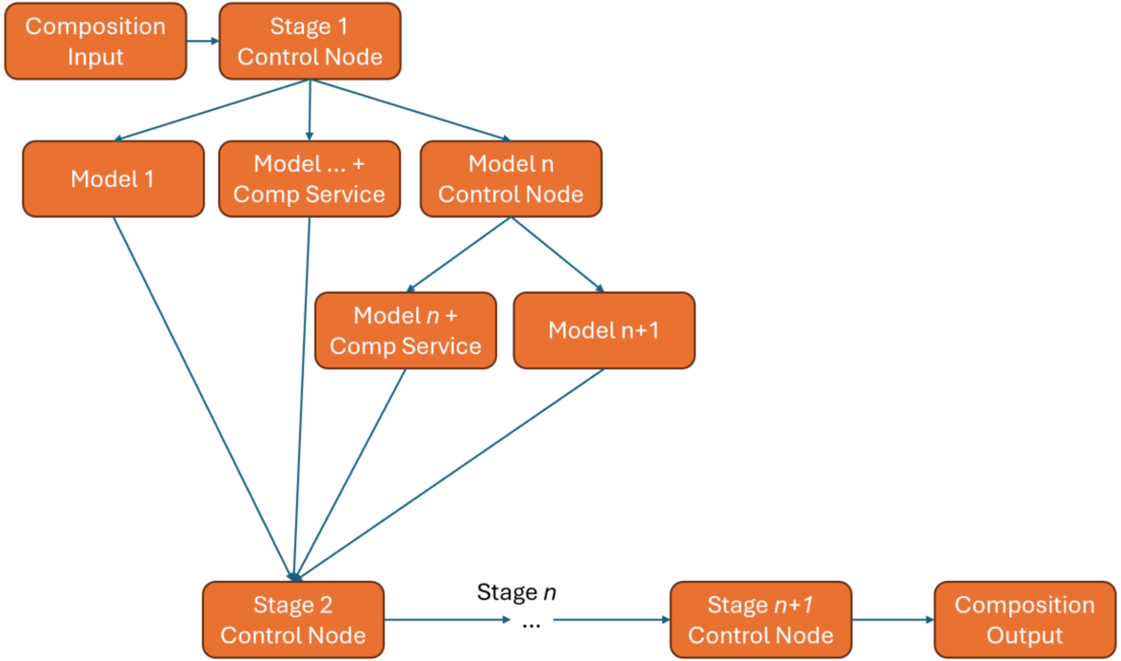
- Hierarchical, tree-like structures for agent behavior
- Node types: Composite (AND/OR), Decorator, Leaf (action/check)
- Easy to visualize, human-readable, verifiable



Left to right execution

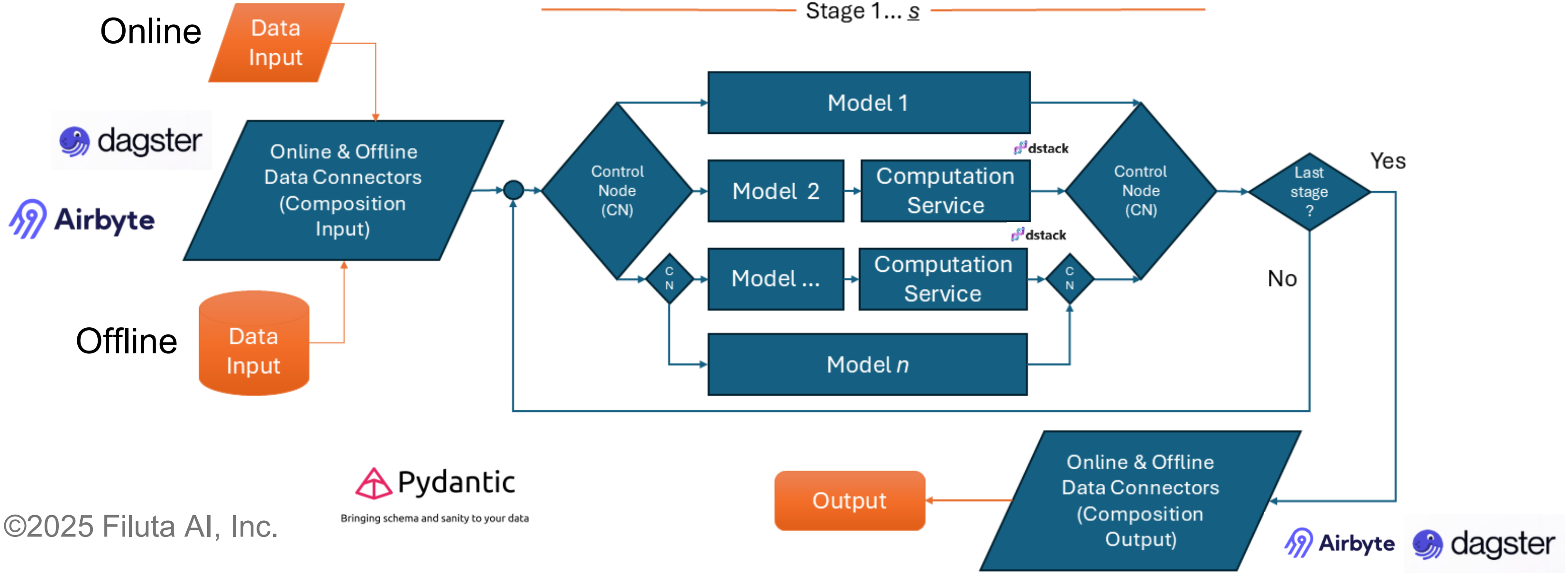
05/Directed Acyclic Graphs (DAGs)

- Used to manage dependencies and execution order
- Prevent cycles and deadlocks; enable efficient computation



06/Control Nodes: Logic-driven Orchestration

- Data and model aware, logic-driven
- Marshal data into/out of models
- Capabilities: combination, selection, transformation, orchestration
- Example: Reduce hallucinations by consensus across models



07 / Control Nodes: Configuration and Capabilities

1. **Permeability**: How data flows through the gate

   (a) **Fully permeable:** complete pass-through to downstream

   (b) **Partially permeable**: partial pass-through to downstream

   may be combined, selected, or transformed before being passed

   (c) **Impermeable**: Data is not passed-through

   will be combined, selected, or transformed before being passed downstream if at all

2. **Combination**: inputs are combined before being passed downstream

3. **Selection**: selected inputs are passed downstream if at all (e.g., filtered)

4. **Transformation**: inputs are altered before being passed downstream

5. **Downstream Model Configuration**: specific model configurations for downstream model

   Example: LLM hyper-parameters

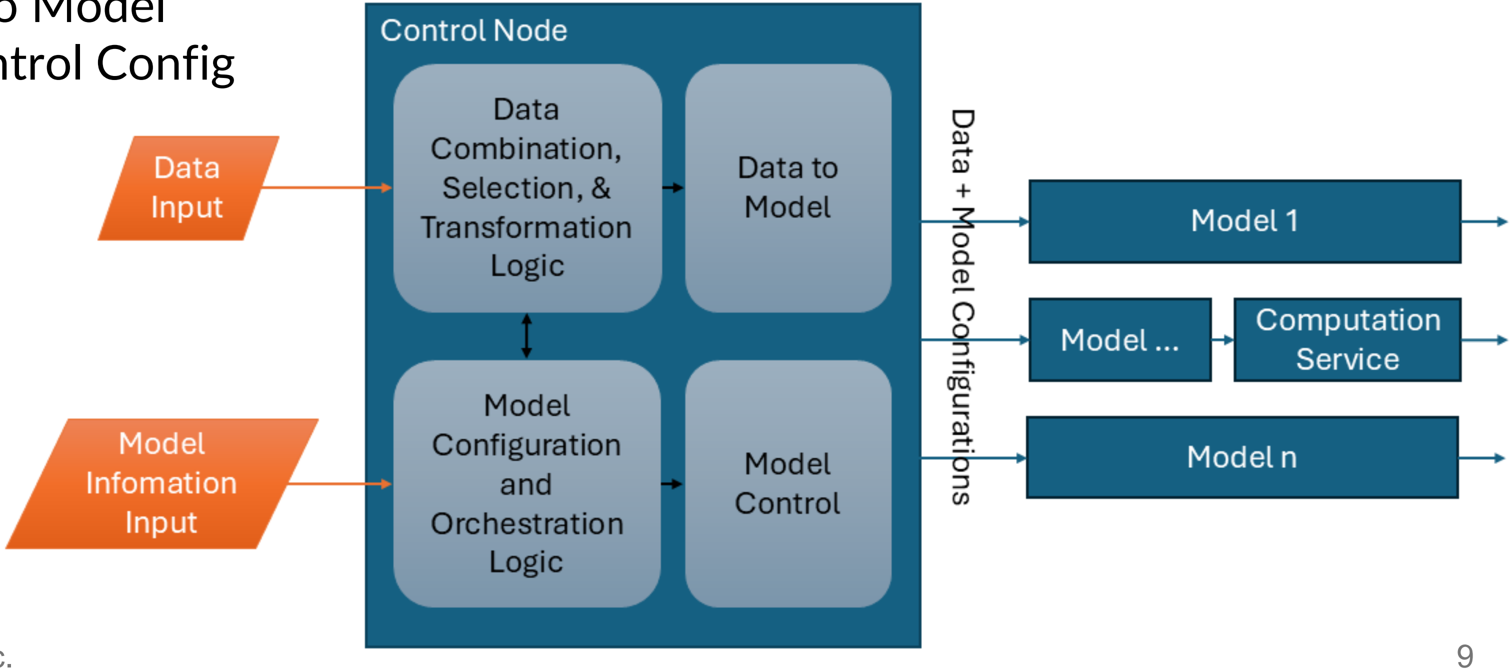
6. **Orchestration**: Defines conditions of received outputs and when/how to proceed in processing

   Examples:

   - Orchestration may only require a few responses instead of all responses before proceeding
   - It may require specific properties to the responses before proceeding
   -However, it should never permanently halt processing and any resolutions needed made locally

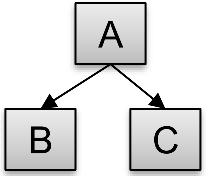
08 / Control Nodes: Internals

- Data Permeability → Combination, Selection, and Transformation
- Model Configuration and Orchestration
- Data out to Model
- Model Control Config

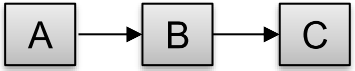

Data Input
Model Infomation Input
Control Node
Data Combination, Selection, & Transformation Logic
Data to Model
Model Configuration and Orchestration Logic
Model Control
Data + Model Configurations
Model 1
Model ...
Computation Service
Model n

09/Ways to Compose Models

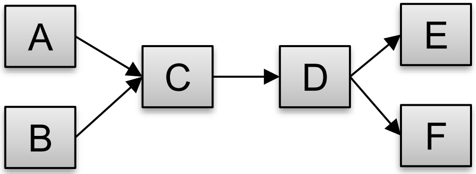
- **Hierarchical**: One inside another (nesting)



- **Sequential**: Chained execution



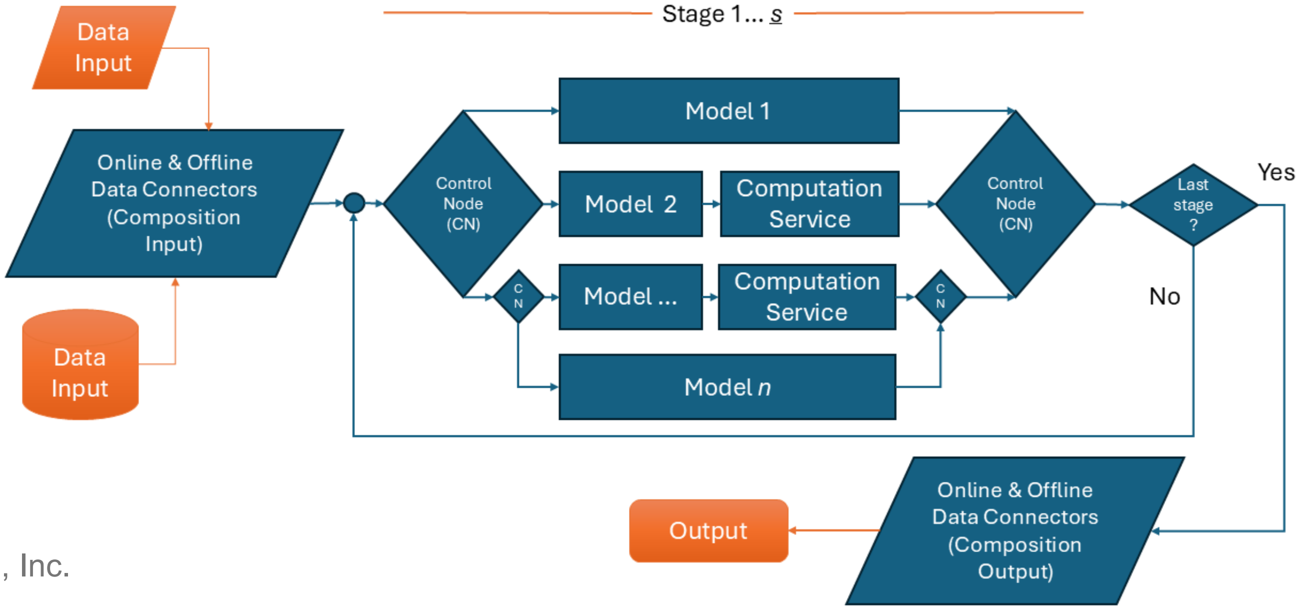
- **Concurrent**: Parallel/interleaved operation



- *Control nodes implement all patterns*

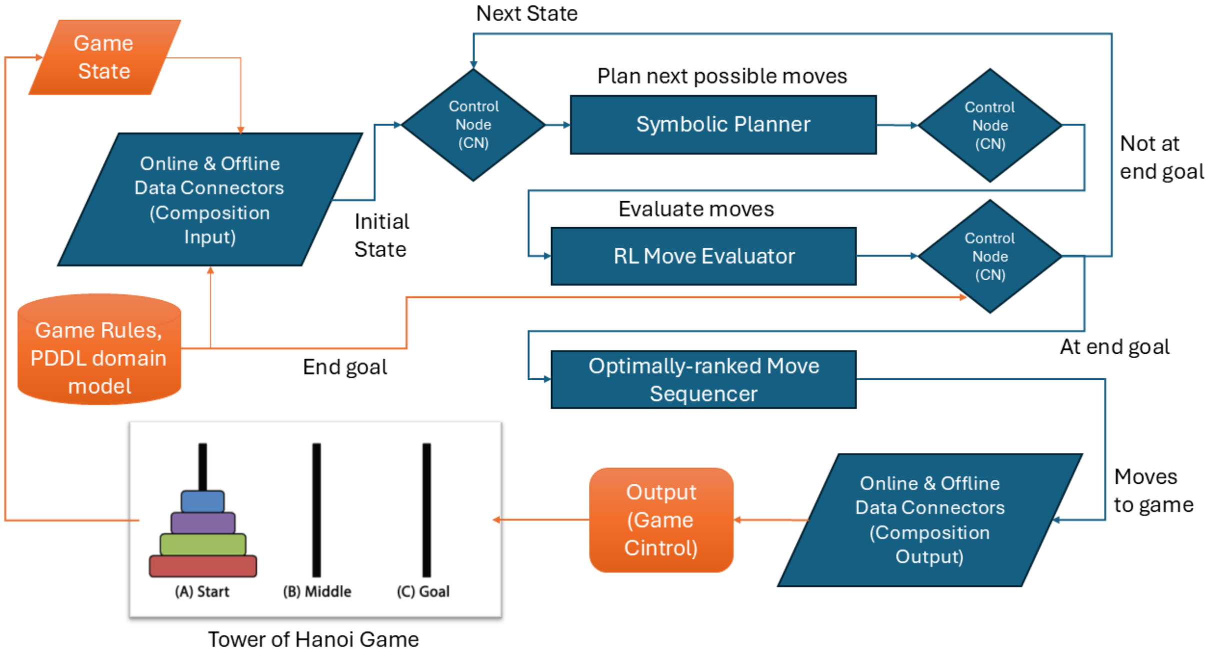
10/Building Composite AI Systems

- Data connectors handle online/offline data for real-time + training
- Models: ML, planning, constraint programming, LLMs, etc
- Orchestrate data and models using control nodes via graph configuration



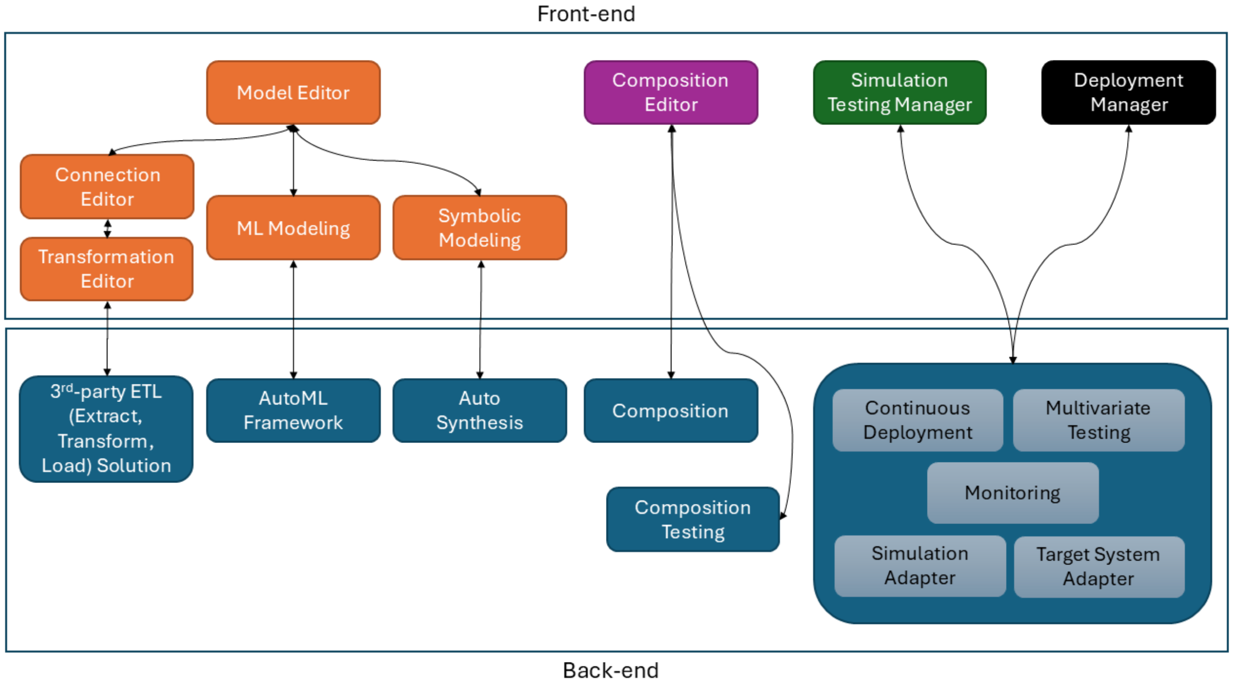
11/Case Study: Tower of Hanoi

- Combines symbolic planner (PDDL) and RL move evaluator
- Data connectors convert game state into predicates/features
- Control node integrates valid moves, scores, and selects optimal action



12/End-to-End Composite AI Systems

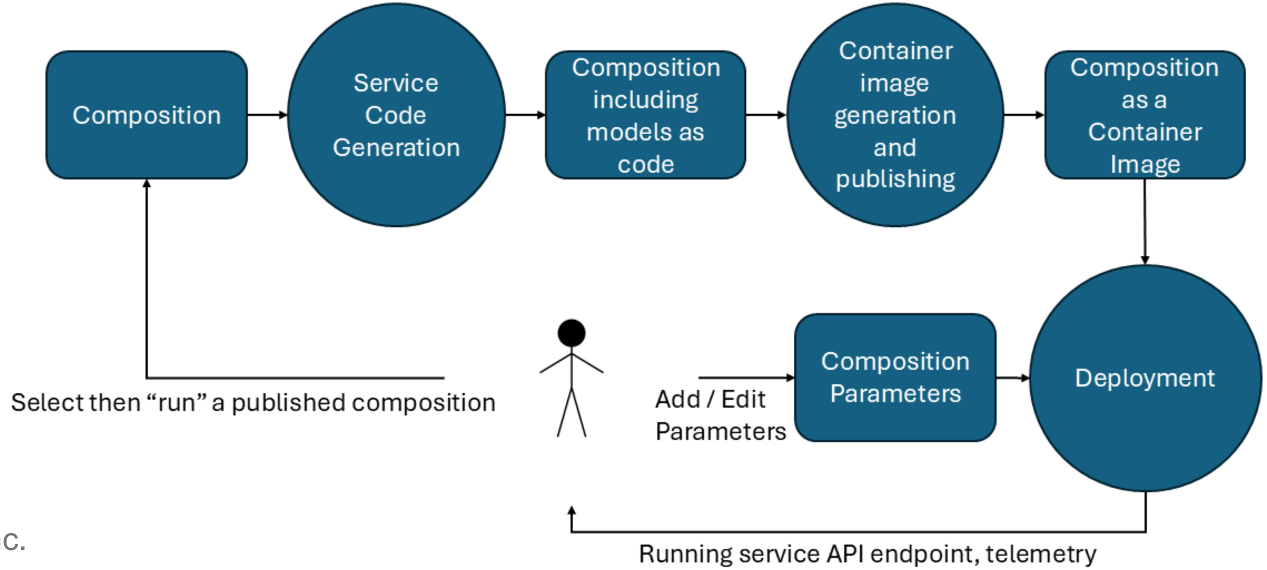
- Components:
  1. Model Editor
  2. Composition Editor
  3. Simulation Testing
  4. Deployment Manager



- Workflow: model development → composition → simulation → deployment

13/Deployment and Scalability

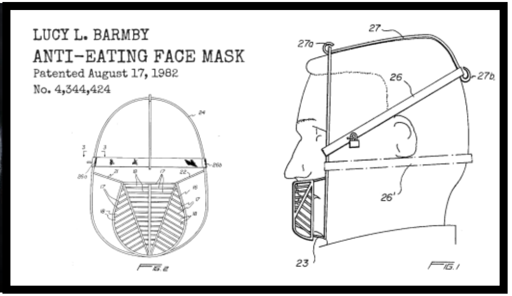
- Push-button deployment of containerized (Docker) microservices
- Support for cloud, edge, or on-premise
- Built-in monitoring, logging, high availability



Composition
Service Code Generation
Composition including models as code
Container image generation and publishing
Composition as a Container Image
Deployment
Composition Parameters
Add / Edit Parameters
Select then "run" a published composition
Running service API endpoint, telemetry

14/Benefits of Composite AI

- **Flexibility**—combine best-suited models for each subproblem
- **Performance**—low-latency, real-time operations, scalable
- **Robustness**—handle complex, dynamic environments
- **Industrial Applicability**—beyond theoretical research



16/Conclusion

- Composite AI with graph-based architecture + control nodes enables robust problem-solving
- Flexible, scalable, and supports rapid development/deployment
- Demonstrated with practical case studies (e.g., Towers of Hanoi, game QA, industry)

Thank you!

Any questions?

michael@filuta.ai

filuta.ai

The work in this presentation is
US Patent Pending
Filing US-20250005328-A1
PCT/US24/36456
(Filings in EU, Japan, Canada, Australia)