Dify Made LLM Application Development Visual. The Applications Have No Agent Schema.

by Nick Clark | Published March 28, 2026 | PDF

Dify provides a visual platform for building LLM applications with workflow orchestration, RAG pipelines, and agent capabilities through a drag-and-drop interface. The platform makes LLM application development accessible to non-developers. But Dify applications and agents have no canonical schema. Each application is a visual workflow configuration, not a structurally defined agent. The gap is between visual agent construction and canonical agent definition.

Dify's visual approach to LLM application development makes agent creation accessible. The gap described here is about structural agent definition behind the visual interface.

Visual configuration without structural validation

Dify allows users to visually configure agent workflows: connect LLM nodes, tool nodes, and conditional logic in a canvas. The visual interface is intuitive. But the resulting agent has no structural schema. There is no validation that the agent carries required governance fields, that memory is properly typed, or that identity is correctly established. The visual configuration produces a functional workflow, not a structurally validated agent.

Workflow state without governed lineage

Dify workflows maintain execution state across nodes. But the state has no governance lineage. There is no record of which governance policy authorized each state transition. The workflow logs what happened. It does not record the governance context under which it happened.

What a canonical agent schema provides

A canonical agent schema would add structural validation to Dify's visual builder. The visual interface would ensure that every agent includes required fields: identity, memory, governance, capabilities, execution state, and lineage. The builder would validate structural completeness. The resulting agents would be interoperable with agents built on any platform that understands the schema.

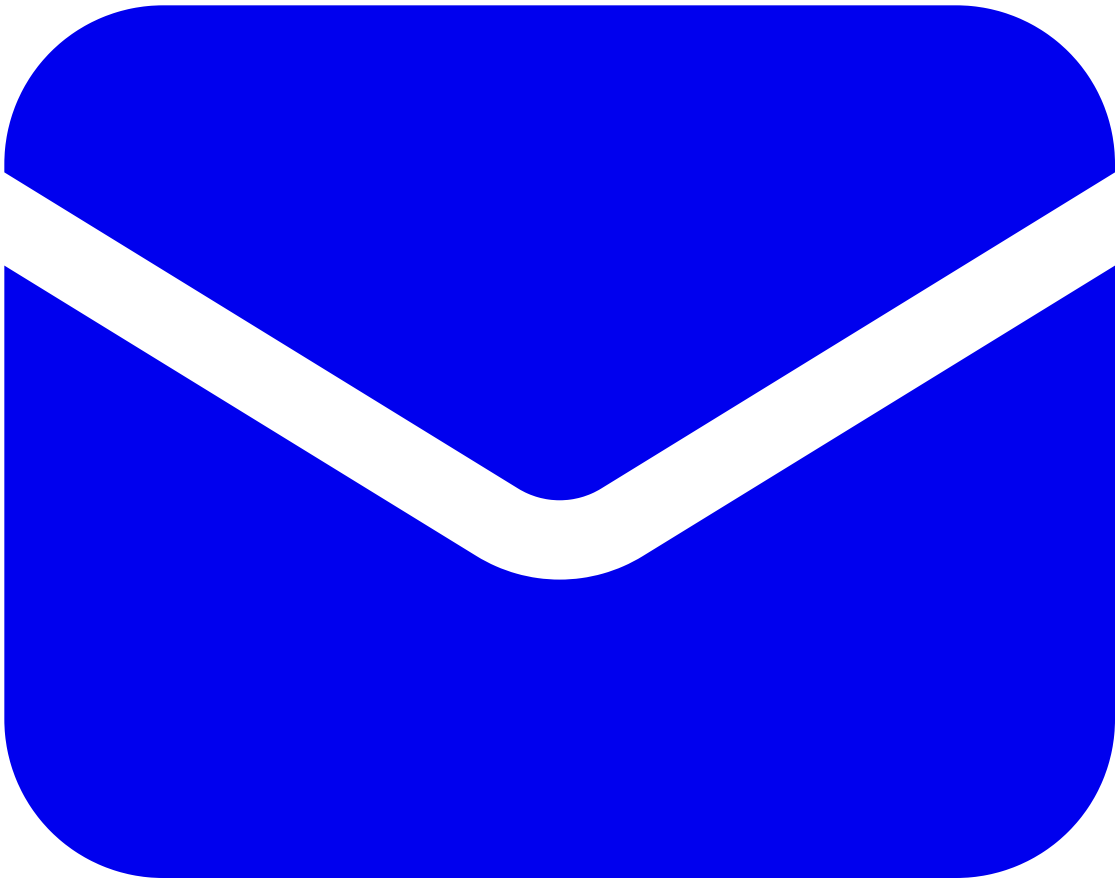
Agent Schema All 21 steps →

Define what an autonomous agent is — structurally.

Patent
US 19/452,651 · filed
Primary Technical Disclosure
○ Cognition-Compatible Semantic Agent Objects and Structural Validation
Secondary Technical
○ Partial Agent Structural Validity: Fewer Fields, Still Deterministic○ Minimum Two-Field Validation Threshold: The Floor of Semantic Structure○ Field Interaction Rules: Deterministic Constraints Between Canonical Fields○ Field-Based Role Typing: Agent Roles Derived From Structural Composition○ Semantic Templates: Predefined Field Arrangements as Agent Class Contracts○ Structural Scaffolding Logic: Resolving Missing Fields Through Inference or Defaulting○ Field-Aware Default Resolution: Deterministic Behavior When Fields Are Absent○ Traceable Semantic Lineage Graph: Mutation History Embedded in Agent Objects○ Serialization With Stateless Compatibility: Reconstruction Without External Session State○ Schema Governance Through Versioned Policies: Cross-Version Structural Interoperability
Applications (General)
○ Enterprise AI Agent Interoperability Through Canonical Schema○ Robotic System Standardization via Structural Field Composition○ Multi-Vendor AI Agent Interoperability○ Digital Twin Standardization Through Canonical Fields○ Healthcare AI Agent Portability○ Defense Coalition Interoperability○ Insurance Claims Processing Through Standard Agents○ Legacy System Integration via Schema Bridging
Applications (Specific)
○ LangChain Built the Agent Framework. It Did Not Define What an Agent Is.○ AutoGen Enabled Multi-Agent Conversations. The Agents Have No Structural Definition.○ CrewAI Organized Agents Into Teams. The Agents Still Have No Schema.○ Semantic Kernel Integrated AI Into Enterprise Code. The Agents It Creates Have No Schema.○ OpenAI Assistants API Provides Agent Tooling. It Does Not Define Agent Structure.○ Google Vertex AI Agents Provide Managed Agent Infrastructure. The Agents Have No Canonical Schema.○ Amazon Bedrock Agents Orchestrate Foundation Models. The Agents Have No Structural Definition.○ Haystack Built Composable NLP Pipelines. The Pipeline Components Have No Agent Schema.○ LlamaIndex Built the Data Framework for LLM Applications. The Data Objects Have No Agent Schema.● Dify Made LLM Application Development Visual. The Applications Have No Agent Schema.
Agent Schema overview →
AQ
deterministic
autonomy

Legal

Last updated: 2026-03-03



-
- nick@qu3ry.net
- 72 28 14 36 01

Invented by Nick Clark | Founding Investors: Devin Wilkie