# Comparing Workflow Orchestration Approaches

## 1. Overview

This document compares two approaches for handling dynamic workflows in an AI-driven system:

1. LLM + Stategraph Approach

2. LLM + Autogen + Stategraph Approach

The document also provides a detailed workflow for the LLM + Autogen + Stategraph approach, including examples of shared context and centralized context and their evolution as the workflow progresses.

## 2. Comparison Table: LLM + Stategraph vs. LLM + Autogen + Stategraph

|  |  |  |
| --- | --- | --- |
| Aspect | LLM + Stategraph | LLM + Autogen + Stategraph |
| Dynamic Workflow Generation | LLM predicts the next step, and Stategraph executes it. No centralized orchestration for dependencies. | LLM predicts or sequences tasks, and Autogen handles orchestration, including dependencies and dynamic task routing. |
| Orchestration | No orchestration beyond task sequencing by LLM. | Autogen manages dependencies, parallel tasks, retries, and fallbacks dynamically. |
| Error Handling | Limited to Stategraph's predefined transitions and retries. | Autogen dynamically adjusts workflows based on errors (e.g., retry a task, switch paths). |
| Dependencies | Dependencies must be predefined in the Stategraph logic. | Autogen dynamically resolves dependencies between tasks based on context. |
| Transparency | Moderate: Stategraph provides clear execution paths, but LLM reasoning can be opaque. | High: Autogen maintains a centralized context for visibility into workflow state and decisions. |
| Flexibility | High for task prediction; limited for dependencies and fallback paths. | Very high: Combines LLM's reasoning with Autogen's dynamic routing and Stategraph's structured execution. |
| Scalability | Scales well for simple workflows but struggles with complex dependencies. | Scales well for both simple and complex workflows with multiple agents and parallel tasks. |
| Best Use Case | Simple or linear workflows where task dependencies are minimal. | Complex workflows with conditional tasks, multiple dependencies, and advanced error handling. |

## 3. When to Use Each Approach

### LLM + Stategraph

Use this approach when:

* • Workflows are simple, linear, or semi-linear.
* • Task dependencies are minimal or can be hardcoded into the Stategraph.
* • Error handling and retries are not critical.

Example Use Case: A document validation and processing pipeline with predefined steps like Validation → Triage → Decision.

### LLM + Autogen + Stategraph

Use this approach when:

* • Workflows involve complex dependencies or parallel tasks.
* • Dynamic routing, fallback paths, and real-time error handling are required.
* • Scalability and flexibility are priorities.

Example Use Case: Healthcare claim processing with conditional tasks like Eligibility, Pricing, and Adjustments, where different paths depend on intermediate results.

## 4. Workflow: LLM + Autogen + Stategraph

This section provides a detailed workflow for the LLM + Autogen + Stategraph approach, showing step-by-step progression with shared context and centralized context updates.

### Step-by-Step Example

Initial Setup:

Centralized Context:  
{  
 "document\_type": "claim",  
 "completed\_steps": [],  
 "results": {},  
 "pending\_tasks": ["Validation", "Triage", "Eligibility", "Pricing", "Decision"],  
 "errors": []  
}

#### Step 1: LLM Prediction - First Task

LLM predicts "Validation".  
Autogen delegates the Validation task to the Stategraph.  
Stategraph executes Validation, and the result is updated in the centralized context.  
  
Updated Centralized Context:  
{  
 "document\_type": "claim",  
 "completed\_steps": ["Validation"],  
 "results": {  
 "Validation": {"status": "success", "details": "Document is valid"}  
 },  
 "pending\_tasks": ["Triage", "Eligibility", "Pricing", "Decision"],  
 "errors": []  
}

#### Step 2: LLM Prediction - Next Task

LLM predicts "Triage".  
Autogen delegates the Triage task to the Stategraph.  
Stategraph executes Triage, and the result is updated in the centralized context.  
  
Updated Centralized Context:  
{  
 "document\_type": "claim",  
 "completed\_steps": ["Validation", "Triage"],  
 "results": {  
 "Validation": {"status": "success", "details": "Document is valid"},  
 "Triage": {"type": "claim", "intent": "process\_claim", "details": "Claim details"}  
 },  
 "pending\_tasks": ["Eligibility", "Pricing", "Decision"],  
 "errors": []  
}

#### Step 3: LLM Prediction - Eligibility

LLM predicts "Eligibility".  
Autogen delegates the Eligibility task to the Stategraph.  
Stategraph executes Eligibility, and the result is updated in the centralized context.  
  
Updated Centralized Context:  
{  
 "document\_type": "claim",  
 "completed\_steps": ["Validation", "Triage", "Eligibility"],  
 "results": {  
 "Validation": {"status": "success", "details": "Document is valid"},  
 "Triage": {"type": "claim", "intent": "process\_claim", "details": "Claim details"},  
 "Eligibility": {"status": "success", "eligibility": True}  
 },  
 "pending\_tasks": ["Pricing", "Decision"],  
 "errors": []  
}

#### Step 4: Eligibility Fails (Error Scenario)

If the Eligibility task fails:  
Result: {"status": "failed", "reason": "Missing data"}  
Autogen triggers fallback logic to retry or skip to Pricing.  
  
Updated Centralized Context:  
{  
 "document\_type": "claim",  
 "completed\_steps": ["Validation", "Triage"],  
 "results": {  
 "Validation": {"status": "success", "details": "Document is valid"},  
 "Triage": {"type": "claim", "intent": "process\_claim", "details": "Claim details"}  
 },  
 "pending\_tasks": ["Eligibility", "Pricing", "Decision"],  
 "errors": ["Eligibility check failed: Missing data"]  
}

#### Step 5: Final Steps

LLM predicts the remaining steps: "Pricing" → "Decision" → "Stop".  
Stategraph executes these steps, and the final context is updated.  
  
Final Centralized Context:  
{  
 "document\_type": "claim",  
 "completed\_steps": ["Validation", "Triage", "Eligibility", "Pricing", "Decision"],  
 "results": {  
 "Validation": {"status": "success", "details": "Document is valid"},  
 "Triage": {"type": "claim", "intent": "process\_claim", "details": "Claim details"},  
 "Eligibility": {"status": "success", "eligibility": True},  
 "Pricing": {"status": "success", "pricing": "Calculated"},  
 "Decision": {"status": "complete", "summary": "All tasks processed successfully"}  
 },  
 "pending\_tasks": [],  
 "errors": []  
}