Munder Difflin Multi-Agent System Report

Agent Workflow Diagram

The system utilizes a central Orchestrator to coordinate between specialized agents. The flow begins with request parsing, moves to inventory validation and historical pricing analysis, executes the transaction, and concludes with financial reporting.

Roles of the Agents

The system is divided into four distinct agents to handle specific aspects of the business logic:

OrchestratorAgent

Serves as the central controller. Its primary role is to manage the lifecycle of the user request. It acts as the bridge between the user and the specialized agents, ensuring that stock is checked before quotes are generated and that transactions are only recorded after a quote is finalized. It also triggers the AccountAgent to update financial status after sales.

QuoteAgent

Handles all Natural Language Processing (NLP) tasks. Its responsibilities include parsing unstructured text into structured JSON (`structure_request`), retrieving relevant historical data (`get_quote_history`), and generating the final pricing and explanation (`calculate_quote`). It focuses on the "creative" and "analytical" side of the process.

InventoryAgent

Manages the "backend" operations. Its responsibilities are strictly defined as checking stock levels (`check_item_stock`, `get_inventory_availability`) and recording financial transactions (`order_item_stock`). It is the only agent authorized to write to the transactions table.

AccountAgent

Handles all financial auditing and reporting. It provides standardized reports on the company's fiscal health (`get_financial_report`), including cash balances and total inventory valuation. This separates financial oversight from the operational logic of the other agents.

Workflow Description

The workflow follows a sequence starting with the **Orchestrator** sending raw text to the **QuoteAgent**, which parses it into a structured format. The **Orchestrator** then requests the

InventoryAgent to check stock levels. Following the stock check, the **Orchestrator** asks the **QuoteAgent** to retrieve historical data to infer a final price and explanation. Once the quote is established and the sale is recorded by the **InventoryAgent**, the **Orchestrator** calls the **AccountAgent** to generate a financial summary, ensuring the cash and inventory balances are accurately reflected in the system's state. If the **InventoryAgent** reports that stock has fallen below the minimum threshold, the **Orchestrator** triggers a replenishment order, provided the **AccountAgent** confirms sufficient cash is available.

Evaluation Results

The system processed **20 unique quote requests**, resulting in a net increase in cash reserves. The inventory management logic maintained stock levels, automatically reordering items when they fell below the minimum threshold.

Metric	Start Value	End Value	Net Change
Cash Balance	$42,389.22	**$43,925.52**	+$1,536.30
Inventory Value	$7,610.78	**$6,074.48**	-$1,536.30
Total Assets	$50,000.00	**$50,000.00**	$0.00

Analysis

Revenue Generation: The agents generated $1,536.30 in net cash flow, a significant increase over previous iterations.

Inventory Efficiency: Inventory value decreased by an amount exactly matching the cash increase, indicating high transaction integrity. The system successfully managed several large-scale orders, such as Request 20, which involved thousands of flyers and invitation cards.

Asset Stability: The total assets remained perfectly stable at $50,000.00, confirming that the **AccountAgent** and **InventoryAgent** are correctly synchronized during the asset conversion process (inventory to cash).

Specific Strengths:

- The **QuoteAgent** maintains its "friendly" pricing tone, successfully applying discounts for massive bulk orders (e.g., Request 20 total of $6,200).
- The **AccountAgent** provides a robust layer of verification, allowing the Orchestrator to monitor financial health in real-time rather than relying on unverified local variables.
- The system correctly handled stock-outs for 11 out of 20 requests, protecting the company from committing to unfulfillable orders.

Specific Strengths:

The QuoteAgent successfully interpreted the requirement for "friendly" pricing, applying discounts for bulk orders and rounding final totals down to integers or multiples of 5 (e.g., rounding $93.50 to $90.00) in strict adherence to the prompt's tone guidelines.

The system correctly identified scenarios where demand exceeded supply. For requests where the stock was insufficient to fulfill the order, the Orchestrator correctly halted the process and returned a message indicating the quote could not be processed, thereby preventing the company from selling phantom inventory.

Suggestions for Further Improvements

Implement backorder capability

Currently, if an item is out of stock, the Orchestrator rejects the entire order. A better approach would be to have the QuoteAgent check the supplier delivery date if stock is insufficient. If the delivery arrives before the client's deadline, the system should allow the sale with a specific warning in the final quote explanation, allowing the company to capture revenue that is currently being lost.

Parallel execution for multi-item orders.

The current Orchestrator iterates through items in a request one by one to check stock, which causes latency for large orders. Modifying the InventoryAgent to accept a list of items for batch processing would allow the Orchestrator to validate the entire order's inventory in a single tool call, significantly reducing execution time and token usage.