Data Pipelines: The Hammer for Every Nail

Abhishek Tiwari

Citation: *A. Tiwari*, ”Data Pipelines: The Hammer for Every Nail”, Abhishek Tiwari, 2023. doi:10.59350/g4jev-hbh56

Published on: July 08, 2023

For big data and complex data processing, data pipelines have emerged as a popular solution for managing and manipulating data. They provide a systematic approach to extract, transform, and load (ETL) data from various sources, enabling organizations to derive valuable insights. However, as with any technology trend, data pipelines have not been immune to misuse and overuse. In this article, we explore the idea that data pipelines have become the new anti-patterns, as they are often misapplied and misunderstood. I have written my thoughts about this topic a few years back but not a lot changed since then ([1]).

The DAG Model and the Misconception

Data pipelines are commonly implemented as Directed Acyclic Graphs (DAGs), where data flows through a series of processing steps, with each step represented as a node and the dependencies between steps represented as edges. The DAG model offers advantages such as parallelism, fault tolerance, and scalability, making it suitable for many data processing scenarios. However, it is crucial to recognize that a DAG is not synonymous with a workflow, despite sharing some similarities.

Data pipelines, when visualized as DAGs, can create an illusion of workflow-like behavior due to the sequential flow of data. This illusion often leads to confusion, as users may mistakenly assume that a DAG can capture the complexities and nuances of a complete workflow. In reality, a DAG lacks the necessary workflow context, and relying solely on it can result in incomplete solutions and missed opportunities.

For instance, Apache Airflow uses DAGs to describe finite-running batch processing workloads. Airflow provides rich scheduling and execution semantics enabling data engineers to easily define complex pipelines, running at regular intervals. Firstly, Airflow does not support infinitely-running (days, weeks, or even longer) asynchornus processes. Secondly, it processes data in batch context and not at individual context. So if you are processing customer orders using Airflow, context is batch of customer orders and not individual customer order. Thirdly, Airflow follows a task-based approach and primarily concerned with executing individual tasks. Lastly, Airflow is not suited for complex state management and coordination requirements.

Task-based vs. State-based

While task-based DAGs can emulate certain aspects of a workflow, they lack the comprehensive state management offered by state-based workflow orchestration systems. Primary emphasis a DAG is executing tasks in a specific order and handling dependencies. A state-based workflow encompasses the entire process, including the logic, decision-making, and human interaction. , the focus is on the states that the workflow goes through and the transitions between those states. State-based work-

flows are well-suited for modeling long-running processes with complex state management and coordination requirements.

The Danger of Overusing Data Pipelines

The danger lies in perceiving every problem as a nail and using data pipelines as the hammer. While data pipelines excel at handling data transformations and aggregations, they may not be the most suitable solution for all scenarios. Overusing data pipelines without considering the broader workflow context can lead to several issues.

Data pipelines are designed to operate within a fixed structure defined by the DAG. This rigidity can limit adaptability and make it challenging to incorporate dynamic or ad-hoc decision-making processes into the workflow.

When data pipelines are misused as complete workflow solutions, they tend to become overly complex. Trying to represent the entire workflow as a single DAG can result in convoluted pipelines that are difficult to understand, debug, and maintain.

As data pipelines grow in size and complexity, maintaining them becomes increasingly cumbersome. Changes or additions to the workflow often require modifications to the underlying DAG, leading to a tedious and error-prone process.

While data pipelines are known for their scalability, overreliance on them can lead to scalability issues at a workflow level. The absence of a holistic view of the workflow can result in bottlenecks or inefficient resource allocation.

Orchestrating complex workflows involves more than just coordinating data processing steps. It requires managing dependencies, handling failures, handling user interactions, and handling exceptions, which are not inherent capabilities of a simple DAG-based data pipeline.

Use The Right Tool for the Job

To address the limitations of data pipelines, businesses should adopt dedicated workflow orchestration tools or frameworks that provide features beyond the scope of data pipelines. State-based workflow orchestration systems, like Cadence, Amazon Simple Workflow Service (Amazon SWF), Temporal, or AWS Step Functions, offer robust workflow modeling, execution, and monitoring capabilities. Such tools enable the modeling and execution of complex workflows, offering capabilities like conditional branching, event-driven triggers, user interactions, and exception handling.

Every Company has a Workflow Problem

In today's digital age, companies across various industries are recognizing the importance of effectively managing their business processes. From transportation and logistics to e-commerce and food delivery, the core operations of many successful companies can be viewed as workflow problems. Workflow platforms have emerged as a crucial component for these companies, enabling them to orchestrate complex tasks, automate processes, and ensure efficient collaboration.

Let's take Uber as an example. Uber, the ride-hailing giant, has disrupted the transportation industry by leveraging a robust workflow platform at the heart of its operations. From connecting drivers and riders to managing payments and handling customer support, Uber's entire business model can be conceptualized as a well-designed workflow.

(1) Rider Request and Driver Assignment: When a rider requests a ride through the Uber app, a series of tasks are initiated. These tasks involve finding available drivers in the vicinity, calculating the estimated time of arrival, and selecting the most suitable driver based on various factors such as distance, rating, and availability. This process involves complex algorithms and real-time data processing, all orchestrated through a workflow platform.

(2) Ride Execution: Once a driver is assigned, the workflow transitions into the ride execution phase. This includes tasks such as navigation guidance, tracking the ride's progress, and providing real-time updates to both the rider and the driver. The workflow platform ensures seamless communication and coordination between all parties involved, ensuring a smooth and efficient ride experience.

(3) Payment and Rating After the ride is completed, the workflow transitions to the payment and rating phase. This involves tasks such as calculating the fare based on distance and time, processing payment transactions securely, and prompting the rider and driver to rate each other. The workflow platform handles these tasks, ensuring accurate billing, secure payments, and valuable feedback for continuous improvement.

(4) Customer Support: In case of any issues or queries, Uber relies on its workflow platform to manage customer support. This involves tasks such as receiving support tickets, routing them to the appropriate teams, and tracking the progress of issue resolution. The workflow platform ensures timely and efficient customer support, maintaining high service standards.

By modeling its core business as a workflow problem, Uber has been able to optimize its operations, enhance user experience, and scale its services globally. The workflow platform acts as the backbone of Uber's business, enabling seamless coordination, automation, and monitoring of tasks throughout the entire lifecycle of a ride.

While Uber is a prime example of a workflow-driven business model, the concept can be extended to

various other industries. For instance:

(1) E-commerce companies like HelloFresh heavily rely on workflow platforms to manage order processing, inventory management, fulfillment, and delivery logistics. Workflows ensure smooth coordination between various stakeholders, from order placement to final delivery, while maintaining high efficiency and customer satisfaction.

(2) Food Delivery: Services like UberEats, DoorDash, and Grubhub utilize workflow platforms to manage the entire food delivery process. From receiving orders, assigning delivery drivers, tracking deliveries, and handling payments, workflows ensure the seamless execution of food delivery operations.

(3) Logistics and Supply Chain: Companies involved in logistics and supply chain management, such as DHL and FedEx, employ workflow platforms to handle tasks such as order tracking, warehouse management, shipment routing, and delivery scheduling. Workflows streamline these complex operations, ensuring accurate and timely deliveries.

Conclusion

While data pipelines have undoubtedly revolutionized data processing and analysis, it is important to recognize their limitations and avoid treating them as one-size-fits-all solutions. Mistaking data pipelines for complete workflows can lead to the emergence of anti-patterns, resulting in reduced flexibility, increased complexity, and maintenance challenges. To build effective workflows, organizations should embrace state-based workflow orchestration tools that go beyond the capabilities of data pipelines.

References

[1] A. Tiwari, “Friends don’t let friends build data pipelines,” Jul. 2018, *Abhishek Tiwari*. doi: 10.59350/hcf5n-nfg79.