Open in app ↗ Sign up Sign In

Published in Towards Data Science

You have **1** free member-only story left this month. Sign up for Medium and get an extra one

Eric Broda Follow

Jan 26, 2022 · 8 min read · ✦ · Listen

Save

Data Mesh Patterns: Change Data Capture

Exploring the ins and outs of the CDC pattern

Data Mesh uses the Change Data Capture pattern to move data safely and reliably around the enterprise. Let's do a deep dive on this pattern to see exactly how it works in an Enterprise Data Mesh.

91 | 3

Photo by Dariusz Sankowski on Unsplash

Change Data Capture: A Foundational Data Mesh Pattern

Zhamak Dehghani's recent Data Mesh article has introduced a brand-new way of managing enterprise data. The combination of domain ownership, treating data as a product, self-serve for data, and a federated data governance approach promises to revolutionize enterprise data management.

But implementing an Enterprise Data Mesh also requires new way of approaching the technical foundation for enterprise data management. So, it is important to understand the key patterns used by an Enterprise Data Mesh that enable this capability:

- **Change Data Capture**, used by an enterprise Data Mesh to track when data changes in a database; These changes are captured as "events".
- **Event Streaming Backbone**, used to communicate CDC events and other notable events (for example, an API call to the Data Mesh) to interested consumers (within and between Data Products) in an Enterprise Data Mesh. (a deep dive is available here).

- **Enterprise Data Product Catalog**, a repository allowing developers and users to view meta-data about Data Products in the Enterprise Data Mesh.
- **Immutable Change / Audit Log**, which retains data changes within the Enterprise Data Mesh for future audit and governance purposes.

In this article I will do a **deep dive** into the first Enterprise Data Mesh pattern, **Change Data Capture**. Several topics will be addressed:

- Pattern Summary,
- Pattern problem domain and context,
- Pattern execution within the Enterprise Data Mesh, and,
- Candidate vendors that implement CDC to enable the Enterprise Data Mesh pattern.

This article is the first in a series. Subsequent articles will discuss the other Enterprise Data Mesh patterns.

But before we start our deep dive, you may want to level-set with a recap of the Enterprise Data Mesh architecture, available here, and a summary of foundational Data Mesh patterns which is available here.

Pattern Summary

Change Data Capture (CDC) captures entries in a database's transaction log and publishes them (with an Event Streaming Backbone) to any interested party (application, person etc) across the enterprise. It does this outside of the original transaction which means that CDC captures changes in operational (or analytic) data without any impact to the originating application or transaction flow.

Context and Business Problem

Moving data safely, reliably, and consistently around an enterprise is a challenge at the best of times. Enterprises use several approaches to address this challenge, but two that are most common.

First, "two-phase commit" (2PC) can be used where data is updated synchronously across multiple databases. Unfortunately, approach is complex and costly and is

usually reserved for situations where it is crucial to keep multiple data sources in sync.

A second approach is to use ETL (Extract-Transform-Load) techniques to update a primary database first, and then after a delay, a second database is updated. But this approach, typically implemented using batch processes leaves data stale, out-of-sync, and inconsistent.

Solution

Modern CDC products address this problem by providing a repeatable pattern for capturing data changes, and with supporting patterns such as the Event Streaming Backbone, provide a way to move data safely, reliably, quickly, and consistently around the enterprise.

CDC solutions (especially when combined with an Event Streaming Backbone) have many benefits:

- **Simpler Architecture**: CDC captures data from the database transaction log after transactions have been committed to the local data source, thereby eliminating 2PC challenges.
- **Non-intrusive**: By capturing the data change event after the data has been committed, CDC does not require code changes in the source data system; this is a critical factor when migrating data from older legacy systems where it is prohibitive in many cases to change code.
- **Simpler Consumption**: CDC provides a common format of each data change record (usually "before" and "after" columnar data plus relevant meta-data) in an easy-to-use format such as JSON.
- **Production Grade Integrations**: They have production-grade built-in connectors to common event platforms such as Kafka to communicate data changes safely and reliably.
- **Near Real-Time Data Transfers**: The data transferred using CDC is available in near real-time — it is common to have the time from legacy data commit to receipt of the CDC event in a half second (500 milliseconds) or less.

How It Works

Figure 1 (below) illustrates how the Change Data Capture works.

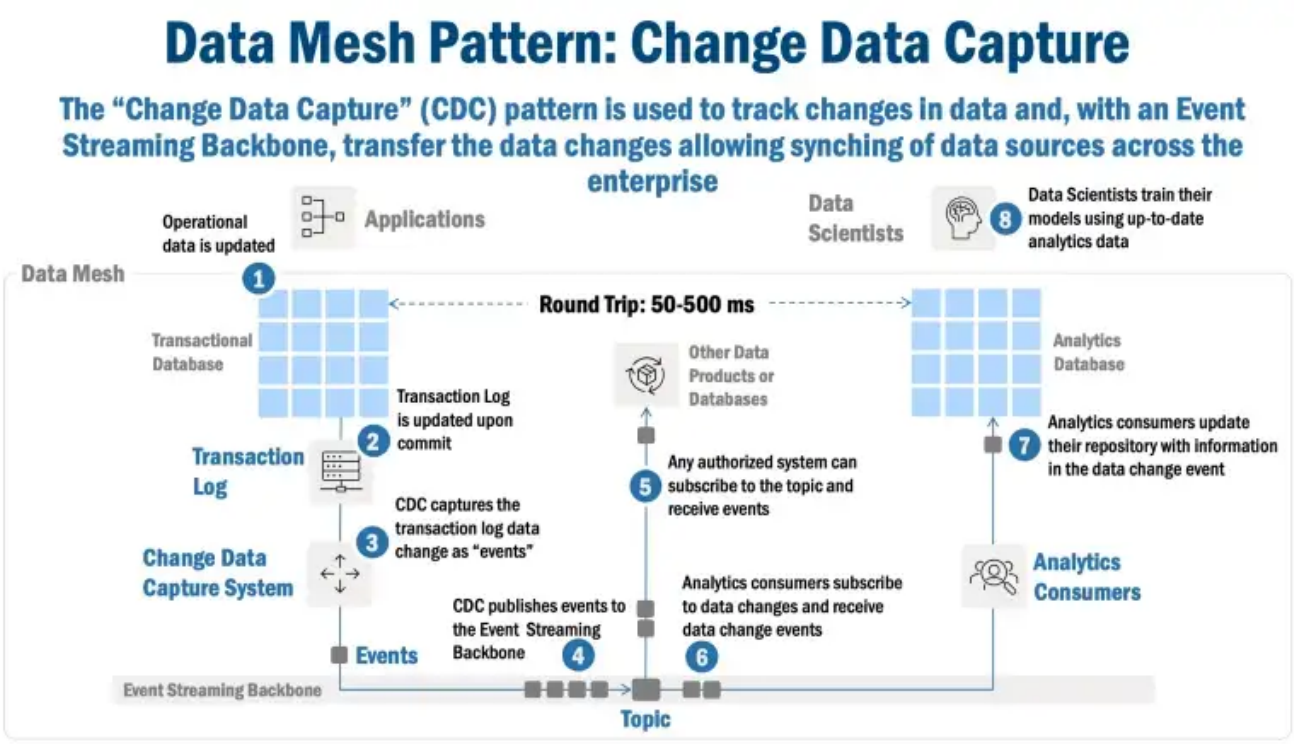

Figure 1: Data Mesh Pattern: Change Data Capture

1. Data is updated (insert, update, delete) within a database. In many cases, these updates occur in transactional or legacy databases, although I have seen scenarios where analytic data act as a source for the CDC.

2. Each operation is committed in the database's transaction log. This is common to almost all database — transactional or analytical — and forms the foundational before/after image of the data that is to be captured.

3. The Change Data Capture (CDC) system captures each entry in the database's transaction log, formats the transaction log entry into a "event" that has well known format (typically JSON, defined by a JSON Schema). These events contain all information required by downstream systems including not just the "after" copy of the data (ie. after the commit occurred) but also the "before" copy of the data which can be used for audit/governance purposes.

4. CDC publishes the event to a topic managed by an Event Streaming Backbone (typically Kafka). The Event Streaming Backbone pattern provides a variety of abstractions that make it easy to send data across the enterprise: **Events**, defined by JSON Schemas, are distributed within the Enterprise Data Mesh; **Topics** are used to queue and distribute events across the enterprise — the Enterprise Data Mesh uses well-known topics that act like queues by allowing

many entities to publish and consume events; **Producers** publish events to topics — producers in an Enterprise Data Mesh may be APIs, applications, or CDC; **Consumers** consume events from topics — consumers in an Enterprise Data Mesh may be any entity or application that subscribe to a topic and are notified when an event is available for processing; **Event Stream Processors** process events either on a per-event or aggregating on a time-window basis enabling for very sophisticated and powerful analysis techniques in an Enterprise Data Mesh; **Brokers** manage the above components to ensure safe and reliable event communications across the entire Enterprise Data Mesh.

5. Any subscriber to a topic would receive the message and process it as needed; Subscribers would presumably subscribe to topics ahead of time.

6. In a common case, the CDC events from a legacy system are received by an analytics event "consumer".

7. Analytics consumers update their analytics database with information in the data change event.

8. Any analytics users may use the synchronized analytics repository using near real-time data; For example, Data Scientists may use the analytic repository — that is synchronized in near real-time (50–500ms round trip) — to train their models using up-to-date data.

Pattern Usage Scenarios for Data Mesh

An Enterprise Data Mesh uses CDC (with the Event Streaming Backbone) for a variety of purposes:

- **Propagate changes to downstream subscribers and system in the organization**: data Mesh uses this to synchronize data within a Data Product as well as between Data Products; It is quite common for AI/Machine Learning training systems are models are fed by this pattern in the Enterprise Data Mesh.

- **Enable the Real-Time Enterprise**: Data Mesh takes advantage of near real-time data capture and transmission to enable next generation analytics and engagement systems which thrive on up-to-the-minute data. Data Scientists recognize that the near real-time data allows for better results.

- **Tracking data changes for audit purposes:** Data Mesh captures CDC events which can be deposited in an immutable change/audit log to support audit and governance requirements (which is especially crucial in mission critical AI/ML applications).

Consider the following example: A customer analytics repository is used to train critical AI/Machine Learning models. However, the data is populated using overnight batch processes. As a result, data scientists realize that this data is inconsistent requiring extensive data engineering efforts to cleanup. They also realize that the data is 24–48 hours old and that this stale data leads to poor AI/ML model results and predictions.

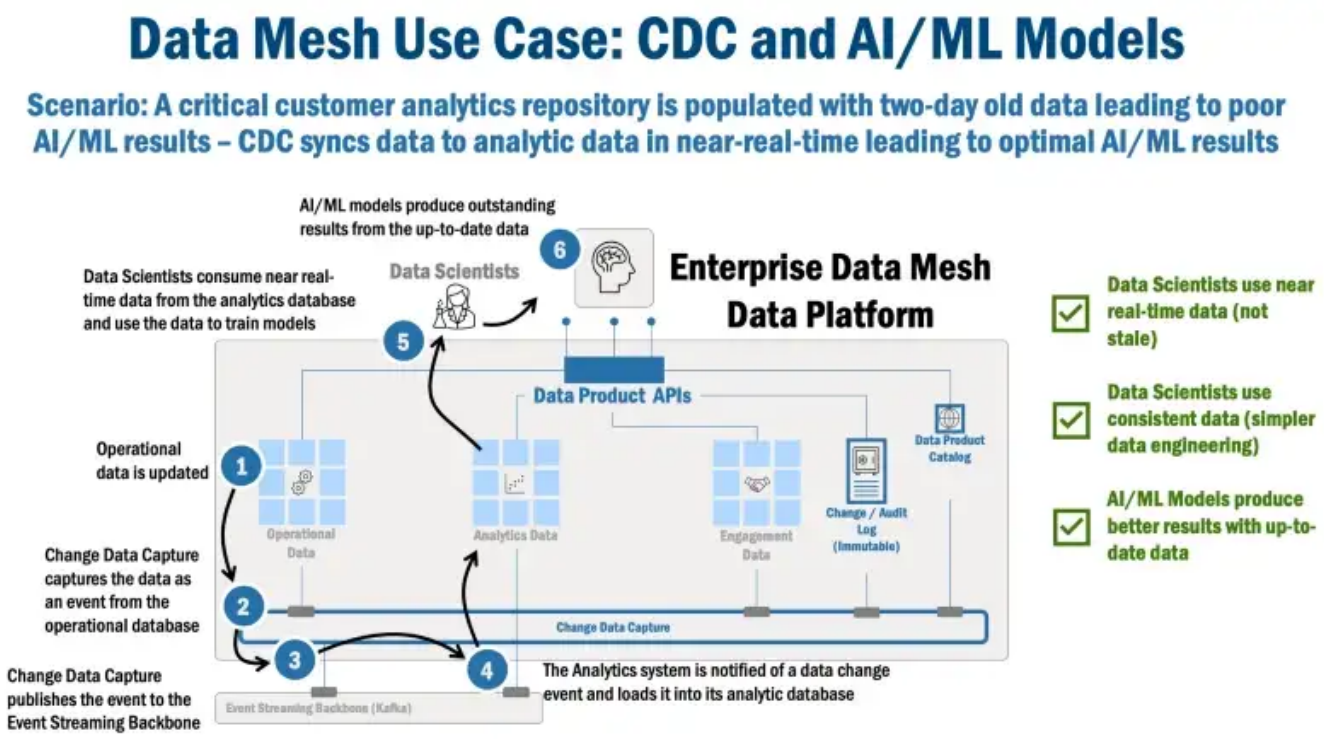

Figure 2: Data Mesh Use Case: CDC and AI/ML Models

Using CDC (and the Event Streaming Backbone) in the Enterprise Data Mesh, the situation is addressed as follows (figure 5):

1. Operational data is updated.
2. Change Data Capture (CDC) captures the data from the operational database and transforms it into an event.
3. CDC publishes the event (containing all information about the recent data change) to the Event Streaming Backbone.

4. An analytics system is notified (in near real-time) of the data change event and loads it into its analytics database.

5. Data scientists consume near real-time data from the analytics database and use it to train their models.

6. AI/ML models produce outstanding results since they are using up-to-date models!

Vendor Landscape

There are two popular and capable CDC products that I have worked with. I am comfortable recommending either of them for use in an Enterprise Data Mesh, although there are differences which I outline that may make one better than the other for your specific situation:

- Debezium: This is probably the most popular open source solution, and for good reason. First it works out-of-the-box with source connectors for Kafka and hence works with minimal configuration and setup with the most popular Event Streaming Backbone product. Second, it integrates with many popular database products including Postgres, MongoDB, Oracle and many others; Third, it has a very active contributor community which bodes well for its long-term future. This is the "go-to" solution if you are integrating modern SQL-based or NoSQL databases into your Enterprise Data Mesh.

- Connect from Precisely: This product offers excellent CDC capabilities (similar to Debezium) but also integrates with mainframe data sources including older IMS and VSAM databases (which Debezium does not yet offer). This makes it a primary consideration for larger Enterprise Data Mesh implementations that aim to accelerate "mainframe modernization" initiatives.

*Full disclosure: I have **no financial interest** in recommending any of the above products — I am highlighting these products because I have some experience with them, and they have worked well for me.*

Concluding Thoughts

The Enterprise Data Mesh is enabling the real-time digital enterprise by making it easy to move data safely, reliably, quickly, and consistently around the enterprise. CDC is one of several foundational patterns used to deliver this capability.

Hopefully this article gives you the necessary insight to build a CDC capability and kickstart your Enterprise Data Mesh!

***

All images in this document except where otherwise noted have been created by Eric Broda (the author of this article). All icons used in the images are stock PowerPoint icons and are free from copyrights.

Data Mesh Data Science Data Management Machine Learning

Artificial Intelligence

Sign up for The Variable

By Towards Data Science

Every Thursday, the Variable delivers the very best of Towards Data Science: from hands-on tutorials and cutting-edge research to original features you don't want to miss. Take a look.

By signing up, you will create a Medium account if you don't already have one. Review our Privacy Policy for more information about our privacy practices.

Get this newsletter

About Help Terms Privacy

Get the Medium app

Download on the App Store GET IT ON Google Play