AXIOM Media Pipeline: Technical Architecture Overview

Date: May 28, 2025

Author: Arturo Cruz Suárez

Status: In-Development / Prototype

1. Executive Summary

AXIOM Media Pipeline is an industrial-grade infrastructure solution designed to orchestrate the full lifecycle of audiovisual assets, from initial ingest to technical review distribution. The system addresses the critical challenge of asset fragmentation by implementing a Single Source of Truth (SSOT), ensuring data integrity and version lineage at all times.

Developed with a focus on distributed systems engineering, AXIOM utilizes a decoupled architecture based on asynchronous micro-tasks to automate media transcoding and metadata extraction. By aligning workflows with MovieLabs' 2030 Vision principles, AXIOM reduces "logistical debt" within productions, allowing creative teams to focus on storytelling while the system ensures the right asset is available in the right format, automatically and at scale.

2. The Problem: Media Fragmentation & Logistical Debt

Contemporary audiovisual production operates under constant delivery pressure, which frequently results in significant logistical debt. AXIOM identifies and resolves three critical pillars of inefficiency:

- Asset Fragmentation and "Version Hell": The lack of a centralized registry leads to naming collisions and the use of outdated assets, causing costly errors during post-production and final delivery.
- Manual Bottlenecks: Manual transcoding and Quality Control (QC) processes consume expensive engineering and artist hours that should be dedicated to creative tasks.
- Data Silos: Fragmented storage prevents a unified view of the production’s health, making it difficult to track asset progress across different departments (VFX, Color, Audio).

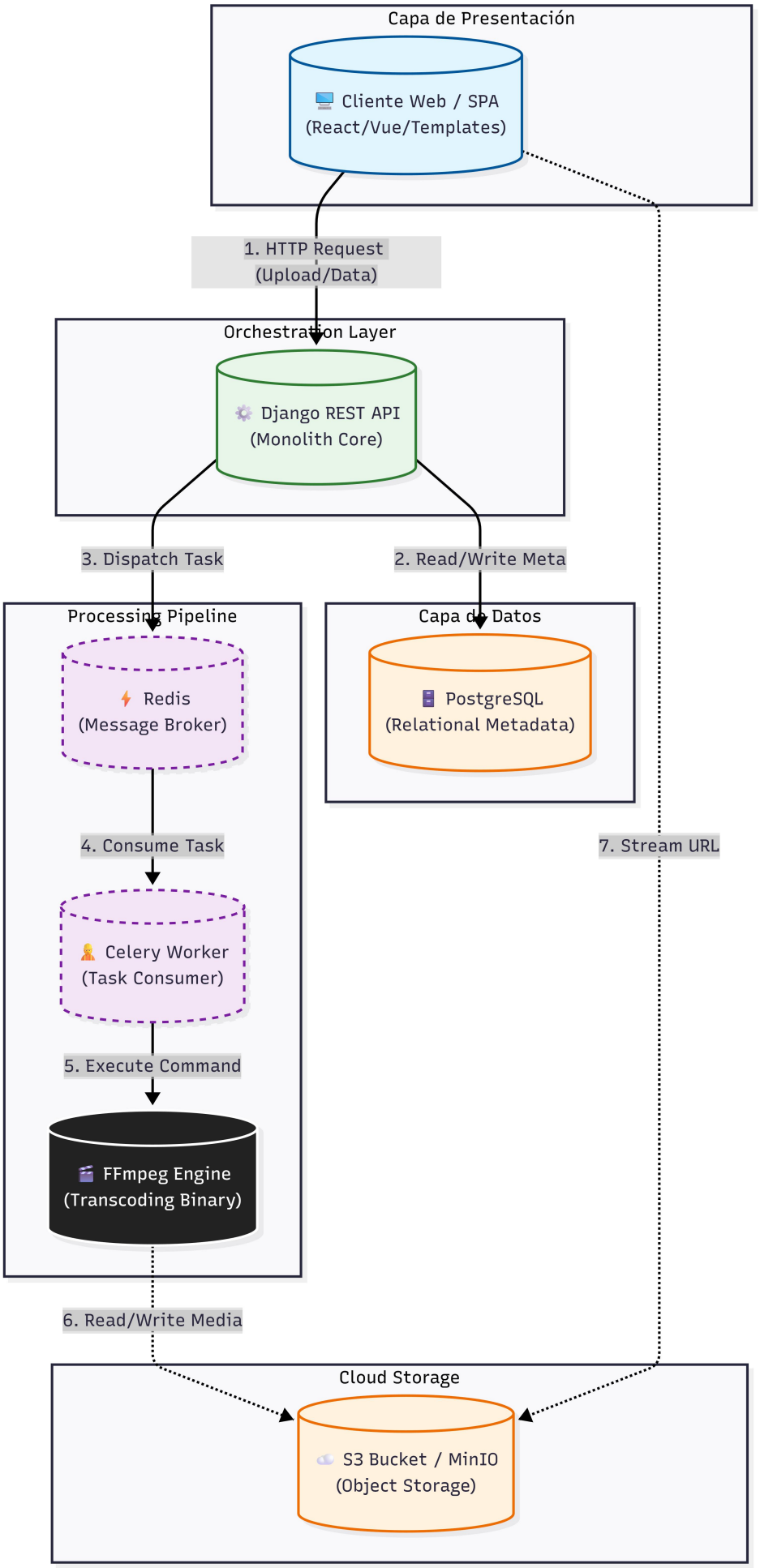
3. System Architecture

AXIOM’s architecture is based on a decoupled, event-driven design engineered to mitigate bottlenecks inherent in large-scale media processing. The system strictly separates metadata management from binary processing, allowing for horizontal scalability of compute nodes.

3.1 High-Level Architectural Flow

- The architecture is designed under a Software-Defined Workflow model, ensuring that heavy media operations do not interfere with platform availability.
- Direct Cloud Ingestion (Principle 1): The system is projected to support direct transfers to Global Object Storage (S3), minimizing latency and eliminating unnecessary resource consumption on the application server.
- Decoupled Processing Engine: Task orchestration is managed through a Messaging Fabric (Redis/Celery), ensuring that the transcoding engine (FFmpeg) operates asynchronously and scalably.
- Metadata Integrity: PostgreSQL acts as the Single Source of Truth (SSOT), storing critical asset metadata and guaranteeing full traceability throughout the entire pipeline.
- Orchestration Layer (Django REST Framework): The flow begins when a client issues an HTTP request (REST/JSON). Django acts as the State Orchestrator, validating request integrity through advanced Serializers. The persistence layer (PostgreSQL) ensures ACID properties for structured metadata, keeping the asset state consistent at all times.
- Binary & Object Storage Abstraction (S3): Following the "Separation of Concerns" principle, high-resolution files never pass through the relational database. AXIOM implements a direct data flow toward Amazon S3. The system only stores pointers (URLs) and cryptographic signatures in PostgreSQL, allowing the storage layer to scale independently of API performance.
- Distributed Task Queue (Redis & Celery): To avoid blocking the main API thread during CPU-intensive tasks, AXIOM implements a Producer-Consumer pattern.
- Producer: Django dispatches transcoding signals to a Message Broker (Redis).
- Broker: Redis manages message orchestration and queue prioritization, decoupling user response time from video processing time.
- Consumer: A pool of Celery Workers monitors the queues, executing media processing tasks via FFmpeg.
- Lifecycle Completion & State Synchronization: The cycle concludes when the Worker generates the derivative asset (Proxy), persists it in S3, and atomically updates the state in the Single Source of Truth (PostgreSQL). This completion notifies the system that the asset is ready for streaming distribution, finalizing the data lineage from ingest to delivery.

3.2 High-Level Diagram



3.3 Data Flow & Lifecycle

- The asset lifecycle in AXIOM is not linear but a process of state transitions designed to avoid orphaned data:
- Ingest Stage: The client sends the binary (to S3) and the metadata (to the API). Django acts as the initial validation filter.
- Persistence Stage: Metadata is recorded in PostgreSQL with a PENDING status. The master file is locked in S3 to prevent accidental modifications (Immutability).
- Processing Stage: An asynchronous task is triggered via Redis. Celery takes control, downloads the necessary segment, and executes FFmpeg.
- Verification Stage (QC): Once the proxy is generated, the system verifies that the resulting file is readable and meets bitrate standards.
- Finalization: The database state changes to READY. The system releases access to the Proxy for the review team (the producer or supervisor).

4. Engineering Deep Dives

This section analyzes the critical components that allow AXIOM to transition from a school project to an industrial-grade infrastructure.

4.1 Asynchronous Ingest & Job Orchestration.

To guarantee a responsive and high-availability platform, AXIOM decouples heavy computational loads from the main API execution thread. This process is managed through a robust asynchronous orchestration engine.

- Producer-Consumer Architecture: Utilizing Celery, the system dispatches video processing tasks to a queue managed by Redis. This allows the user to receive an immediate response (202 Accepted) while the actual work occurs in the background.
- Fault Tolerance & Error Catching: The pipeline is designed to detect execution failures in real-time. If FFmpeg encounters an error (corrupt file, lack of space, container incompatibility), the Celery Worker captures the exception and updates the asset status to ERROR in the database. This prevents "zombie" processes and ensures immediate resource liberation.
- Graceful Failure UX: Instead of infinite loading states, AXIOM provides technical feedback to the user. A processing failure triggers a signal that allows the user to retry the ingest or check the integrity of their original file, maintaining system transparency.

4.2 Data Integrity & Lineage.

AXIOM treats every uploaded asset as an immutable entity. Instead of a traditional filesystem based on overwriting, the pipeline implements a relational versioning model that guarantees the historical traceability of every project.

- Atomic Versioning: Every new ingest generates a unique record in the database linked to a Parent Project ID. This allows for a complete history of iterations without losing previous metadata.
- Audit Trail & State Machine: The system records not only the file but also the state history (Pending -> Rejected -> Approved) and technical comments associated with each version. This turns the database into an infallible audit log for the production.
- Storage-Database Consistency: Through the use of UUIDs and unique pointers to S3, AXIOM ensures that no filename collisions occur. Even if two versions are named the same by the user (e.g., "Final_Cut.mp4"), they remain distinct and protected objects within the infrastructure.

4.3 Media Proxy Generation & Automated Transcoding

Transcoding in AXIOM is not just a format change; it is a data normalization process.

- Engine: We utilize FFmpeg integrated via Python's subprocess to ensure access to low-level libraries.
- Logic: The system automatically detects the source codec and applies a transcoding recipe toward H.264/AAC, optimizing the file for streaming without perceptual quality loss.
- Error Handling: If the video process fails due to a corrupt file, Celery captures the FFmpeg error log and persists it in PostgreSQL, allowing the user to know exactly why the ingest failed (Data Integrity).

5. Scalability & Future Roadmap

AXIOM has been designed under an elastic architecture that allows for the transition from a local implementation to a Cloud Native internet-scale infrastructure.

5.1 Future Milestones (Data Engineering Vision)

To align the pipeline with Netflix's Studio Engineering standards, the roadmap includes:

- Big Data Integration: Implementation of Apache Spark for massive asset telemetry analysis and rendering resource optimization.
- Event-Driven Architecture: Migration toward Apache Kafka to manage real-time event flows across multiple departments (VFX, Audio, Color).
- AI Metadata Tagging: Use of Computer Vision models for automated scene tagging during ingest, reducing asset discovery latency.
- Hybrid Cloud Storage: Implementation of S3 Lifecycle policies (Standard, Glacier) to optimize storage costs based on access frequency.