Key Technologies and Implementation of Broadcasting Supervision Data Asset Management System (Postprint)

Authors: Wang Yang

Date: 2023-10-08T00:00:00+00:00

Abstract

Data related to broadcast television supervision has accumulated to a considerable extent. The construction and maintenance of traditional storage and data processing technology architectures are increasingly inadequate to accommodate the growth rate of data volume and business requirements. Under the premise of ensuring stable operation of the existing system, there is a critical need to employ a new technology architecture for data migration, expansion, and backup. Open-source big data processing and storage software frameworks, exemplified by Hadoop, provide a viable solution. Drawing upon practical experience in constructing data asset management systems, this paper analyzes and introduces the key technologies and processes involved in their development.

Full Text

Preamble

Title: Key Technologies and Implementation of a Broadcasting Supervision Data Asset Management System

Abstract: The data generated from broadcasting supervision has accumulated to substantial volumes, rendering traditional storage and data processing infrastructure increasingly inadequate for the pace of data growth and evolving business requirements. While maintaining stable operation of existing systems, there is an urgent need to adopt new technology frameworks for data migration, capacity expansion, and backup. Open-source big data processing and storage software frameworks, exemplified by Hadoop, provide a viable solution. Based on practical experience in constructing a data asset management system, this paper analyzes and introduces its key technologies and implementation process.

Keywords: big data; Hadoop; HDFS; MapReduce; MPP; Kettle; data warehouse

Classification Code: TN948.6

Document Code: A

Article ID: 1671-0134(2019)07-111-03

DOI: 10.19483/j.cnki.11-4653/n.2019.07.036

Introduction

Broadcasting supervision has undergone major technological transformations, from analog to digital, terrestrial to satellite, and standalone to networked systems. As technology continues to evolve, the number of broadcast frequencies and channels, as well as programming hours, have increased rapidly. Multiple transmission methods—including over-the-air television, cable television, satellite TV, and IPTV—have also emerged quickly, causing the data volume faced by supervision work to expand dramatically. Early work primarily involved structured business data, mainly technical indicators related to broadcast and transmission processes. Broadcast content was monitored in real-time, with recordings kept for very limited durations. Over the past decade, large-scale data storage and processing solutions have emerged from commercial companies and open-source communities, driven by internet development and the urgent need for new technology frameworks suitable for big data environments—aiming for low cost, easy scalability, high concurrency, and compatibility with diverse data types.

These practical application requirements have accelerated the maturation of certain technical solutions, providing feasible answers for broadcasting supervision work and making large-scale storage of audio/video programs possible. Simultaneously, massive data processing technologies and analytical developments can now meet the increasingly sophisticated demands of supervision operations.

[Figure 1: see original paper] Data Asset Management System Technical Architecture Diagram

Currently, broadcasting supervision work primarily utilizes two categories of data: audio/video files corresponding to broadcast programs, and related supervision business data such as broadcast quality reports and television program schedules. The former represents unstructured data, while the latter constitutes structured data. Audio/video file storage has reached petabyte scale, with daily increments in terabytes, encompassing most satellite and cable broadcast programs. Due to storage capacity limits, old audio/video files must be periodically deleted. Although business data volumes are only in the gigabyte range, certain datasets grow rapidly, already reaching hundreds of millions of entries. As these two data categories accumulate, requirements for storage scalability and security have gradually increased, necessitating a balance between read performance

and reliability. Cost-effective and straightforward operations must achieve high concurrent access and distributed storage. Furthermore, massive data volumes create demands for data analysis, requiring new technical solutions to address the insufficient data warehousing capabilities of existing systems. Finally, while traditional commercial databases like Oracle adequately meet relational data management and analysis needs, they involve high costs, complex distributed architecture operations, closed-source code, and pose data asset security concerns. In summary, we need to leverage mature open-source big data solutions from the community to address the data asset management requirements for audio/video files and related business data, implementing a more cost-effective and secure data asset management system that compensates for existing technical system limitations.

1. Technical Architecture Introduction

Individual supervision business systems remain isolated from each other, forming closed loops within each system and failing to achieve effective data sharing. Therefore, the first step requires effective data integration to eliminate data silos, establish unified data standards, and create comprehensive data statistics. Based on this foundation, we can develop visual data statistical analysis functions to achieve accurate grasp and effective management of data assets, supporting the establishment of cross-system, cross-departmental, and cross-regional communication and coordination mechanisms. The following diagram illustrates the technical architecture developed based on these requirements, which has been fully implemented and is currently operating normally.

The bottom layer comprises existing data storage systems, including audio/video data and business data. The middle layer represents the new system in the form of a big data appliance, consisting of a Hadoop platform for storing audio/video files and an MPP database for file metadata and business data. Built upon this foundation is the data asset management system, encompassing functions such as data asset catalogs, storage statistics, incremental statistics, data flow monitoring, and job monitoring. These functions are presented to user terminals on the upper layer through a firewall.

The centralized storage system employs Inspur' s Lustre system, housing historical broadcast program audio/video files. The broadcast database contains technical business data from broadcasts, the program database stores broadcast program catalog data, and the intermediate database synchronizes technical business data from other systems. All three databases use Oracle appliances with RAC (Real Application Cluster) storage. These two data categories undergo transfer and extraction through Hadoop software and ETL tools in the big data appliance, creating data directly usable by the data asset management system.

2. Key Technologies and Implementation

Based on technical architecture requirements, we must determine which key technologies to adopt for implementing data synchronization, ETL, and other functions. The following diagram illustrates data flow between software modules and the technologies or products employed.

[Figure 2: see original paper] Data Asset Management System Software Structure Diagram

The Lustre system in the centralized storage is a Linux-based open-source distributed file system. Its primary characteristic is strong read/write capability, particularly for large files, utilizing Stripe striping technology for parallel file access. Metadata servers store file metadata information, while object storage servers hold the actual files. Lustre' s main issues include relatively weak data protection, as its underlying object storage structure does not support data backup and recovery. Additionally, metadata suffers from single points of failure, addressed through a shared storage layer supporting master-slave switching. If the shared storage layer fails, causing metadata to become inaccessible, the entire system is affected.

We selected Hadoop 2.0 for reading and storing audio/video files. HDFS supports multiple replicas stored across different nodes, enabling data protection with data verification capabilities. Although file operations do not support POSIX interfaces like Lustre and can only be implemented through APIs—with lower read performance—HDFS is suitable for long-term file preservation and data warehouse construction. Moreover, the HDFS ecosystem is exceptionally rich, offering many powerful open-source frameworks, which is significantly advantageous compared to Lustre. For this project, we utilized two management nodes and twenty data nodes. Management nodes use Zookeeper for platform cluster load balancing, while data nodes store hundreds of terabytes of audio/video files via HDFS.

The interface for reading centralized storage system file information is implemented using MapReduce. MapReduce programs parse the centralized storage system contents, extracting metadata from audio/video files and performing two tasks: first, synchronizing audio/video files to the big data Hadoop platform; second, obtaining audio/video file attribute information and synchronizing it to the MPP database. Data synchronization management strategies include file synchronization and metadata synchronization. Metadata synchronization occurs every 15 minutes, primarily using the `find` command to query files added within the last hour under the current date directory, saving metadata to the database. Synchronization scheduled tasks are implemented through Linux' s `crontab` command, with metadata extraction log files located in a logs folder at the same level as the jar package. File synchronization occurs nightly at midnight, copying all files from the previous day to HDFS based on metadata information in the database. After completing primary files, file synchronization is executed periodically based on file additions and synchronization requirements.

For structured data storage, we adopted a MySQL-based MPP (Massively Parallel Processor) database. MPP signifies massively parallel processing, with systems composed of many loosely coupled processing units, each CPU possessing its own private resources and running its own operating system and database instance replica. MPP architecture databases feature parallel task execution, distributed data storage, distributed computing, private resources, horizontal scaling, and Share Nothing architecture.

Horizontal scaling represents the primary design goal of MPP databases, which support strict relational models such as SQL92 with extensions and stored procedures, support transactions, and ensure strong data consistency. Problems addressed include improving data processing performance and volume. Unlike the Shared Nothing architecture where each node uses private resources, the original database' s Oracle RAC employs a Shared Disk architecture, where each node uses its own CPU and memory while sharing disk storage—meaning data sharing. When storage performance reaches bottlenecks, adding nodes cannot achieve parallel capability expansion. Additionally, Oracle RAC' s closed-source code resides in expensive Oracle appliances.

The MPP database cluster comprises six nodes divided into three types: management nodes, data nodes, and SQL nodes. There are two management nodes responsible for managing other cluster nodes, providing configuration data, starting and stopping nodes, and running backups. Since these nodes manage configurations for other nodes, they should be started before other nodes using the `ndb_{mgmd}` command. Four data nodes store cluster data; the number of data nodes relates to replica count and are started with the `ndbd` command. Four SQL nodes access cluster data, typically started using `mysqld --ndbcluster` or by adding `ndbcluster` to my.cnf and using `mysqld`. Management nodes handle cluster configuration files and logs; each node retrieves configuration data from management nodes and requests location determination. When new events occur in data nodes, information about these events is transmitted to management nodes and written to cluster logs.

The ETL process from Oracle RAC to the MPP database employs the open-source ETL tool Kettle. ETL represents the Extract, Transform, Load process —a critical component in building data warehouses. Traditional ETL tools suffer from centralized execution and high server performance requirements. To address these shortcomings, this project adopted a distributed ETL technology based on distributed principles, implementing a cluster-distributed ETL process on top of a distributed file system. This distributed ETL system offers high scalability and throughput efficiency while automatically achieving load balancing with high execution efficiency. Distributed ETL technology includes four components: Spoon, Pan, Chef, and Kitchen. Spoon provides a graphical interface for designing ETL transformation processes, defining extraction sources, targets, and rules—the main body of ETL—while jobs control transformation execution. Pan batch-runs ETL transformations designed by Spoon as a background program without a graphical interface. Chef can create jobs

that facilitate automating complex data warehouse updates by allowing each transformation, task, script, etc., to be checked for correct execution. Kitchen allows batch execution of jobs designed by Chef, such as using a time scheduler, and also runs as a background program. The distributed ETL cluster comprises eight nodes: two master nodes and six worker nodes.

Based on operational needs, we have introduced the technical architecture and key technologies of the data asset management system, along with specific implementation methods. Through this project, audio/video and business data resources have been integrated, providing larger, more secure, and easily scalable storage that lays a solid foundation for further data mining. As new mature projects continuously emerge from the big data open-source software community, the data asset management system can fully integrate new tools to achieve continuous functional expansion.

(Author's Affiliation: National Radio and Television Administration Monitoring Data Processing Center)

Note: Figure translations are in progress. See original paper for figures.