ChinaRxiv

AI translation · View original & related papers at
chinarxiv.org/items/chinaxiv-202310.01997

Application of Open-Source Components in Xinhua News Agency Data Service Platform (Postprint)

Authors: Qiao Aijun

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

Abstract

With the convergence of media and the rapid development of mobile Internet, traditional IT architectures are no longer capable of meeting the demands of news reporting under the new paradigm. To address these emerging requirements, Xinhua News Agency has advanced the construction of four major application platforms. Serving as the data core of Xinhua News Agency, the data service platform provides underlying data support for these four application platforms. The data service platform employs a microservices architecture and extensively utilizes open-source, distributed components. This paper introduces the data service platform and its components, and summarizes two major transformations in data management brought about by the data service platform.

Full Text

Application of Open Source Components in Xinhua News Agency's Data Service Platform

Abstract: With the development of media convergence and the rapid growth of mobile internet, traditional IT architectures can no longer meet the demands of news reporting in the new era. To adapt to these new requirements, Xinhua News Agency has advanced the construction of four major application platforms. Serving as the data core of Xinhua, the data service platform provides underlying data support for these four platforms. Built on a microservices architecture, the data service platform employs numerous open-source, distributed components. This paper introduces the data service platform and its components, and summarizes two major transformations in data management brought about by the platform.

Keywords: data service platform; big data; open source components
Chinese Library Classification: TQ018

ChinaRxiv 中X

Document Code: A
Article ID: 1671-0134(2018)07-065-04
DOI: 10.19483/j.cnki.11-4653/n.2018.07.018
Author: Qiao Aijun

1. Overview of the Data Service Platform

Xinhua News Agency's traditional IT architecture has become inadequate for meeting the demands of modern news reporting. In response, Xinhua began constructing a new-generation technical system at the end of 2016. Currently, four major application platforms based on open-source technology have been essentially completed: the all-media editing and distribution platform (New Editing and Distribution), the all-media supply platform (New Supply), the office collaboration platform (New OA), and the all-media business management platform (News Hotspots, News Clues, Landing Statistics, and Reporting Command). The data service platform serves as Xinhua's data aggregation center, providing underlying support for these four platforms. The platform has now been initially completed, offering services for data ingestion, storage, servicing, processing, analysis, statistics, and application. It has achieved aggregation of internal dispatches, imported resources, internet data, user information, behavioral data, and service audit data, providing data services for various application systems.

The data service platform provides four major categories of services externally: service management, data services, big data computing services, and GlusterFS file storage services. Service management enables unified management of service interfaces, providing functions for service registration, discovery, routing, and management. In addition to registering and managing its own services, the service management system also registers services from other systems such as the user authentication management system and office collaboration platform. Data services currently provide 99 services across eight categories: basic services, full-text retrieval, data subscription, semantic analysis, recommendation, tagging, user information, and resource management. For big data computing services, 17 nodes have been deployed at Xinhua's headquarters and 14 nodes at the Dongba data center. These services provide Spark and Storm computing capabilities for the entire agency, along with HDFS, HBase, and Hive storage and analysis capabilities. The GlusterFS distributed file storage service is similarly divided into two clusters: the headquarters primary cluster currently has 30 nodes, while the Dongba backup cluster has 12 nodes, providing 322TB of space for the four major platforms and the data service platform.

[Figure 1: see original paper] Overall Architecture Diagram of the Data Service Platform

The data service platform is divided into four layers from bottom to top: the basic component layer, data processing layer, data service layer, and application layer, all built upon the infrastructure layer.

2. Basic Component Layer

The basic component layer utilizes 18 major open-source components that provide services for the data service platform, all-media editing and distribution, new OA, landing statistics, influence analysis, and project management systems. Based on their functions within the data service platform, we categorize these components into five types: databases, distributed file storage, big data computing, data exchange, and others.

2.1 Database Components

Database components primarily include MongoDB, ES, MySQL (MariaDB), HBase, and Codis/Redis. MongoDB is used to store internal dispatches, while HBase and ES store both internal dispatches and internet data. MySQL is used for relational data required for system operations. Codis/Redis stores frequently used business data in the data service platform, such as authorized classification data, user behavior data, and dispatch recommendation data.

MongoDB is a database designed for scalability, high performance, and high availability, positioning itself between relational and non-relational databases. It can be deployed from a single server to large, complex multi-data center architectures. Leveraging in-memory computing advantages, MongoDB provides high-performance data read and write operations. Its replica replication and automatic failover capabilities provide enterprise-level reliability and operational flexibility for applications, while its sharding mechanism offers convenient horizontal scaling. Its unstructured nature facilitates storage of various types of dispatch data, user data, and log data. In the data service platform, MongoDB has replaced the original Oracle database for storing internal dispatches, serving as the core application database.

ES (Elasticsearch) is a distributed search and analysis system composed of multiple Lucene instances, functioning as a distributed search engine and data analysis engine capable of near-real-time processing of massive data. Combining full-text search, data analysis, and distributed technology, ES provides powerful capabilities. The data service platform uses ES for retrieval processing.

After MySQL came under Oracle's control, its original developers created the MariaDB branch. MariaDB is highly compatible with MySQL and guarantees open-source freedom. The data service platform primarily uses MySQL to store system metadata and configuration data such as service and application information, line and directory monitoring information, and distribution information.

HBase is a distributed, column-oriented open-source database derived from Google's Bigtable paper by Fay Chang. Unlike traditional relational databases, HBase is suitable for unstructured data storage and is primarily used to store internal and internet dispatches for offline data analysis.

Codis/Redis is a key-value in-memory database. Codis is a distributed Redis solution that consolidates multiple Redis servers for horizontal scaling through

Codis Proxy. For upper-layer applications, connecting to Codis Proxy is not significantly different from connecting to a native Redis server—applications can use Codis as if using standalone Redis. Codis handles request forwarding and non-stop data migration transparently at the 底层, making these background operations invisible to clients, which can simply perceive it as a Redis service with unlimited memory.

2.2 Distributed File Storage Components

Distributed file storage components include HDFS and GlusterFS. HDFS (Hadoop Distributed File System) is a highly fault-tolerant system suitable for deployment on inexpensive commodity machines. It provides high-throughput data access, making it ideal for large-scale dataset applications. HDFS relaxes some POSIX constraints to enable streaming reads of file system data. In the data service platform, HDFS primarily provides file system support for components like Hive and HBase, storing logs and user behavior data.

GlusterFS is a distributed file system that stores objects as files. It unifies space across multiple servers to form a storage pool for external use. Compared to other distributed file systems, GlusterFS's greatest advantage lies in its operational simplicity and ease of use. Its architecture consists of three layers: cluster, volume, and Brick. In the data service platform, clusters are divided into three categories based on file size: small-file clusters storing files around 1KB-100KB (primarily CNML files, image thumbnails, and some audio/video keyframes); medium-file clusters storing files around 500KB-10MB (generally images, Word, PDF attachments); and large-file clusters storing files above 10MB (primarily audio/video files). Each cluster establishes different volumes according to business needs, with each volume composed of Bricks from some or all servers in the cluster. A Brick is a created directory used to store data.

2.3 Big Data Computing Components

Big data computing components include Storm, Hive, Spark, and YARN. Storm is used for distributed real-time stream data processing with application scenarios including real-time analysis, continuous computation, online learning, distributed RPC, and ETL. A Storm cluster consists of Nimbus and Supervisor nodes. Nimbus is the master node responsible for task submission, cluster task assignment, and cluster monitoring. Supervisor nodes are compute nodes that accept tasks assigned by Nimbus and manage their own Worker processes. Nimbus and Supervisor coordinate through Zookeeper. In the data service platform, Storm implements data format conversion and data dumping, providing a data foundation for analysis.

Hive is a data warehouse infrastructure built on Hadoop. It provides a series of tools for ETL (extract, transform, load) operations and can store, query, and analyze large-scale data stored in Hadoop. Hive defines a simple SQL-like query language called HQL that allows SQL-proficient users to query data while

also enabling MapReduce developers to create custom mappers and reducers for complex analytical tasks.

Spark is a general-purpose parallel computing framework open-sourced by the AMP Lab at UC Berkeley, similar to Hadoop MapReduce. Spark possesses the advantages of Hadoop MapReduce but differs in that intermediate output results can be saved in memory, eliminating the need to read and write HDFS. Consequently, Spark is better suited for data mining and machine learning scenarios.

YARN's fundamental concept is separating resource management, task scheduling, and monitoring into different modules. Its main approach involves creating a global ResourceManager (RM) and an ApplicationMaster (AM) for each application. Here, an application refers to a single job or a DAG (directed acyclic graph) of jobs. The ResourceManager controls the entire cluster and manages allocation of basic computing resources, carefully arranging resource components (computing, memory, bandwidth, etc.) to NodeManagers. The ResourceManager also allocates resources together with ApplicationMasters and works with NodeManagers to launch and monitor their underlying applications.

2.4 Data Exchange Components

The data service platform employs two message exchange components: RabbitMQ and Kafka. Using message exchange components offers two major advantages: first, system decoupling that reduces inter-system coupling; second, business peak shaving—when producers generate large amounts of data that consumers cannot process quickly, the message component acts as an intermediate layer to store the data, achieving business peak shaving.

RabbitMQ is a message middleware implementing AMQP (Advanced Message Queuing Protocol), originally developed for financial systems to store and forward messages in distributed systems. In the data service platform, RabbitMQ enables message transmission between applications to complete various dispatch processing workflows, serving as the core application. The figure below illustrates the news dispatch processing workflow in the data service platform, showing that the core message queue is implemented by RabbitMQ, which connects internal modules of the data service platform with external systems.

[Figure 2: see original paper] News Dispatch Processing Workflow

Kafka is a distributed streaming platform with three key functions: publishing and subscribing to streaming records (similar to message queues or enterprise messaging systems), storing streaming messages fault-tolerantly, and processing streaming messages in real-time. Kafka is primarily used to build real-time streaming data pipelines and real-time streaming data applications. In the data service platform, Kafka mainly handles exchange and transmission of internet data, user behavior data, service audit data, and log data. For example, data collected from the internet by the landing statistics system, after cleaning and

filtering, is placed into Kafka topics for ingestion by the data service platform or processing and analysis by other systems. User behavior data collected by the probe system, service audit data from the service management system, and log data from application systems and basic components are transmitted via Kafka to Storm for processing or stored in HDFS for analysis by other applications.

2.5 Other Components

Other open-source components include Zookeeper, Nginx, Sqoop, Ambari, and ELK.

Zookeeper is a high-performance distributed system that provides coordination services for distributed applications, offering configuration maintenance, naming services, distributed synchronization, and group services. Most big data components based on distributed architectures require Zookeeper support.

Nginx is a lightweight web server, reverse proxy server, and IMAP/POP3/SMTP proxy server. Written in an event-driven manner, Nginx delivers excellent performance and highly efficient reverse proxy and load balancing capabilities. It offers high stability, supports hot deployment, starts easily, and can run almost continuously 24/7 for months without requiring restart. It also enables software version upgrades without service interruption. Numerous applications in the data service platform use Nginx as a proxy to implement load balancing or directly provide web application services.

Sqoop is a tool for transferring data between Hadoop and relational databases, capable of importing data from relational databases (such as MySQL, Oracle, Postgres, etc.) into Hadoop HDFS and exporting HDFS data to relational databases. It also provides connectors for some NoSQL databases. The data service platform uses this tool to import user data from MySQL into Hive.

Ambari is a web-based management tool that supports deployment, management, and monitoring of Apache Hadoop clusters. Ambari already supports most Hadoop components, including HDFS, MapReduce, Hive, Pig, HBase, Zookeeper, Sqoop, and Hcatalog. Using Ambari facilitates deployment and installation of Hadoop clusters, starting and stopping relevant components, and monitoring the operational status of Hadoop clusters and their components.

ELK is a log collection, processing, and visualization tool composed of three open-source components (ES, Logstash, Kibana). The data service platform uses ELK for application log analysis.

3. Data Service Layer and Data Processing Layer

Some functions of the data service layer and data processing layer are implemented using the Spring Cloud Netflix microservices architecture. The microservices architecture is built through Eureka + Zuul + Ribbon + Feign + Hystrix, encapsulating common modules in the data processing layer into service interfaces registered on the Eureka server. Eureka maintains the lifecycle of each

service and determines service health through heartbeat checks. Zuul is deployed in front of Eureka as an intelligent router providing a unified entry point for external requests. Service clients use Feign for invocation, with Ribbon implementing server-side load balancing. The Hystrix circuit breaker is introduced to avoid cascading failure effects, providing stronger fault tolerance for service delays and failures.

Leveraging the distributed scalability of open-source components combined with the microservices architecture used in the data service layer, Xinhua's data management architecture has undergone two major transformations. The first transformation is the shift from databases to data. Data storage has transitioned from centralized relational databases to distributed databases; data processing has evolved from simple CRUD operations to data processing, analysis, and mining; data types have expanded from single news dispatches to diversified data including internet data, user behavior data, and log data; and data volume has changed dramatically—from approximately tens of thousands of dispatches generated daily to over three million cleaned internet data items collected daily, plus substantial user behavior and log data. The second transformation is the shift from systems to platforms. Originally, IT systems provided functions as individual units. Over time, Xinhua's IT systems became numerous with overlapping functional dependencies, and after new systems went online, old systems could not be decommissioned in time, occupying substantial resources and increasing operational difficulty. The platform establishment has transformed the approach from providing functions to providing service capabilities, decomposing functions into services with unified registration and management to avoid redundant construction and reduce operational difficulty. The platform's service targets have expanded from serving single applications to simultaneously serving multiple applications. Platform construction has shifted from using commercial products to using open-source components, saving system construction costs, accelerating system deployment speed, and meeting the requirements of rapid product launch and continuous iteration in the internet era. The new platform employs a horizontally layered, horizontally scalable distributed architecture to replace the original vertically integrated architecture, thereby enabling rapid, low-cost, dynamic, and smooth expansion of service capabilities.

References: [1] The Apache Software Foundation. Apache Hadoop. http://hadoop.apache.org, 2018(06):13. [2] Wang Fangxu. Design and Implementation of Business System Microservitization Based on Spring Cloud. Electronic Technology & Software Engineering, 2018(08). [3] Wang Zhanhong, Wang Zhanying, Gu Guoqiang, Ma Guochun, Lü Zhenhua. Research and Practice on Distributed Elastic Search. Microcomputer Applications, 2014(30):7.

(Author's Affiliation: Xinhua News Agency Technology Bureau)

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

Source: ChinaXiv — Machine translation. Verify with original.