Introduction to Distributed File Systems

Explore what we'll be studying in the upcoming chapters regarding distributed file systems.

We'll cover the following

Motivation

In modern distributed systems, the importance of durable data storage and retrieval with high availability cannot be overstated. Distributed file systems extend the abstractions of local file systems and are one of the primary building blocks of any distributed service.

What we will learn

We've selected the following three papers to discuss in the next few chapters:

- [GFS] Sanjay Ghemawat, Howard Gobioff, and Shun-Tak Leung. 2003. The Google file system. In Proceedings of the nineteenth ACM symposium on Operating systems principles (SOSP '03). Association for Computing Machinery, New York, NY, USA, 29–43.
- [Colossus] Google Colossus system
- [Tectonic] Pan, Satadru, Theano Stavrinos, Yunqiao Zhang, Atul Sikaria, Pavel Zakharov, Abhinav Sharma, Mike Shuey, et al. Facebook's tectonic filesystem:

Efficiency from exascale. In 19th USENIX Conference on File and Storage Technologies (FAST 21), pp. 217-231. 2021.

Why did we choose these systems?

The field of file systems is ripe with many systems from academia and industry. We picked three systems—one of them is old but has stood the test of time, while the two recent ones collectively show how designs evolve over time.

The Google File System (GFS)

GFS is a seminal work in a distributed file system that Google presented in 2003. GFS provides a custom data consistency model and aims for data scalability, availability, and good performance (primarily high throughput). GFS is an example of co-designing where GFS was purpose-built on commodity hardware and, for specific use cases, write and read-heavy workloads with a lot of concurrent activity. GFS demonstrates how we can build a distributed file system with thousands of machines inside a data center. GFS is a beautiful design due to its simplicity, while still being able to meet its goals.

GFS 2: The Colossus system

The original GFS system used a centralized control plane (in the form of a manager) to keep the design simple. GFS designers used interesting approaches so that a single manager does not become a single point of failure. However, over the years, while a GFS cluster has been able to grow to a multi-petabytes level, the single manager design was approaching its scalability limits. In 2010, Google revealed the descendent of the GFS—Colossus. Google hasn't revealed the full design details of Colossus, but they have shared bits and pieces of information. The primary reasons for building Colossus were:

- To scale for data beyond petabytes with good horizontal scalability.
- To enable more kinds of applications (for example, files with small data, apps that need low latency at the tail, for example, p99, etc.).

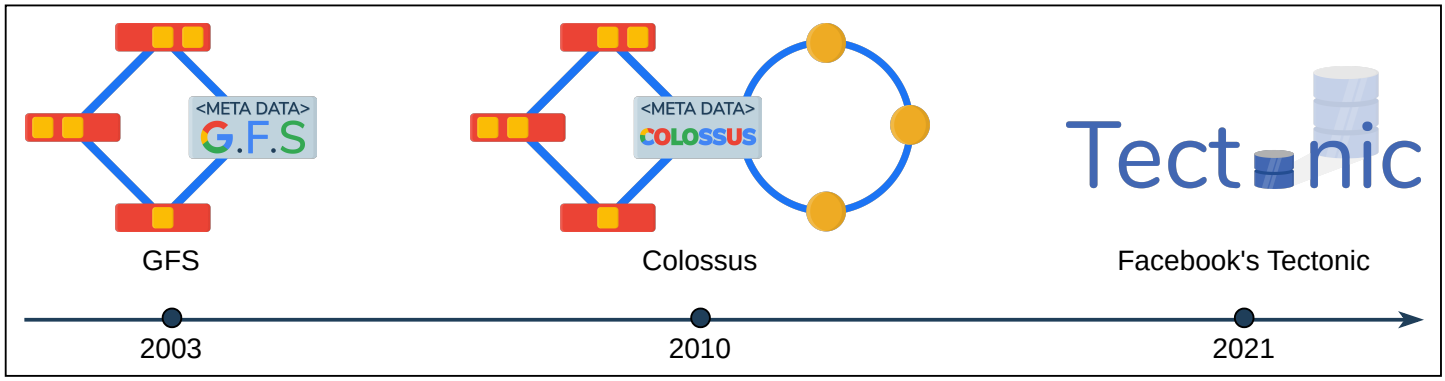
- To enable clients to pick from full replication, Reed-Solomon-based encoding for space-saving, or the use of RAID.

This progression from GFS to Colossus is an example of the design principle that we make the system only as complex as needed for the present (and near-future). Operational knowledge provides a good feedback loop for the next iteration of the system to meet evolving needs.

Hyperscale: Facebook's Tectonic System

Tectonic is Facebook's file system that was revealed in 2021. In many ways, it resembles Colossus. Tectonic's primary focus is on achieving high data scalability with good resource utilization. Tectonic uses many independent services as building blocks, where each building block can scale independently.

We hope our selection of file systems teaches us many important lessons in system design. Let's dive in!



Evolution of distributed file systems

Back
Case Studies: Standin...
Next
Introduction to GFS
Mark as Completed