Hadoop 分布式文件系统的模型分析 *

王　峰，雷葆华
（中国电信股份有限公司北京研究院　北京 100035）

摘 要　Hadoop 分布式文件系统是遵循 Google 文件系统原理进行开发和实现的，受到了业界极大关注，并已被广泛应用。鉴于当前缺乏从系统设计理论的角度对其开展的相关研究，本文从 Hadoop 分布式文件系统架构的建模入手，通过对模型各组成部分进行分析，并将其与传统的分布式文件系统进行比较，总结出 Hadoop 分布式文件系统具有的海量、高可扩展性、高可靠性、高性能等面向云计算领域应用的重要特征。本文有助于研究者系统、深入地研究 Hadoop 分布式文件系统的设计与实现，并为云计算背景下的分布式文件系统设计提供重要的参考。

关键词　Hadoop 分布式文件系统；系统模型；云计算

1　引言

Hadoop 分布式文件系统(hadoop distributed file system，HDFS)是具有高可靠性和高可扩展性的分布式文件系统，能够提供海量的文件存储能力[1]。它的开发和实现遵循了 Google 文件系统 (Google file system，GFS) 的核心原理，而 GFS 作为 Google 云计算核心技术体系的底层，为相关技术(如 MapReduce 分布式计算模型、Bigtable 分布式数据库

* 国家高技术研究发展专项经费资助项目(No.2008AA01A317-3)

2008. K-INGN 2008. First ITU-T Kaleidoscope Academic Conference, 2008

2　刘伟彦，张顺颐.下一代网络中基于遗传算法的 QoS 组播路由算法.电子与信息学报,2006,28(11)：2157~2161

Research on the Service-Aware Model of NGN Architecture and Key Technologies

Rao Xiang[1], Zhang ShunYi[1], Zhou Yun[2]
(1.Nanjing University of Posts and Telecommunication,Nanjing 210003,China；
2.Nanjing University of Finance & Economics,Nanjing 210046,China)

Abstract　By building a intelligent NGN Service-aware model, proposing a QoS control and management architecture of NGN data flow, this paper presents a approach for the key technologies of multi-services transport, service identification, QoS control and management applied in the design and implementation of NGN.

Key words　next generation network, service-aware, QoS control　　(收稿日期：2010-10-15)

表 1　分布式文件系统的透明性需求

需求	描述
访问透明	用户程序不必考虑文件的分布问题，能够通过完全相同的编程接口对本地文件和远程文件进行访问和操作
位置透明	用户程序能够看到统一的文件名字空间，对文件进行访问的路径名与其在远程服务器中的存放位置无关
移动透明	用户程序无需关心文件的存放位置发生移动所产生的影响，这一需求的满足与位置透明性的实现相关
性能透明	用户程序始终能够获得高效的文件访问能力，即使远程服务器上的文件服务负载在一定范围内发生了变化
扩展透明	用户程序导致的文件服务需求的增大或者网络规模的扩张等问题，能够通过系统的自动扩展得到有效的解决

等)的实现提供了有效的支撑。同样，HDFS 本身以及以它为基础的一系列开源软件技术的研究和开发，已被业界广泛应用到云计算的具体实践中，获得了非常好的效果。

当前，针对 HDFS 的研究普遍关注其具体的技术细节和实施效果，尚缺乏对其进行系统设计理论上的分析和比较。本文从模型分析的角度入手，首先介绍在业界获得普遍认同的分布式文件系统的用户需求和架构模型，然后针对 HDFS 的体系架构进行特征描述和建模分析，最后将 HDFS 与传统的分布式文件系统进行比较，总结了 HDFS 在云计算领域中应用的优势及存在的问题，并对其设计和应用提出建议。

2　分布式文件系统模型

分布式文件系统是分布式系统的关键技术之一，能够以文件的方式实现信息资源的共享。在云计算环境中，分布式文件系统承担着为用户提供文件服务的重任，它要保证用户在访问、保存在云中的文件时能够获得接近甚至在某些方面超出其在使用本地磁盘时的服务质量 (包括性能、可靠性等)。

分布式文件系统通过网络为用户提供远程文件服务，它的设计目标是要使得用户感知不到其访问的是存储在远程服务器中的文件[2]。因此，分布式文件系统的设计特别强调系统对用户的透明性。系统的透明体现在多个方面，具体内容见表 1。

满足用户的透明性需求对于分布式文件系统设计非常关键，直接影响了用户对远程文件的访问体验。除此以外，还有其他一些设计需求，包括分布式文件系统需要具有高可用性，能够支持异构客户端的并发访问，能够提供文件数据的多个拷贝并保证文件数据的一致性和安全性等。

针对这些需求，已经有数量众多的分布式文件系统被提出，它们在设计和实现上各具特点。为能够对这些文件系统进行分析和比较，参考文献[2]提出了如图 1 所示的分布式文件系统的远程文件服务模型。

该文件服务模型得到了学术界和产业界的广泛认同，主要由扁平文件(Flat File)服务、目录服务和客户端模块 3 部分组成。其中，扁平文件服务实现对服务器磁盘上保存

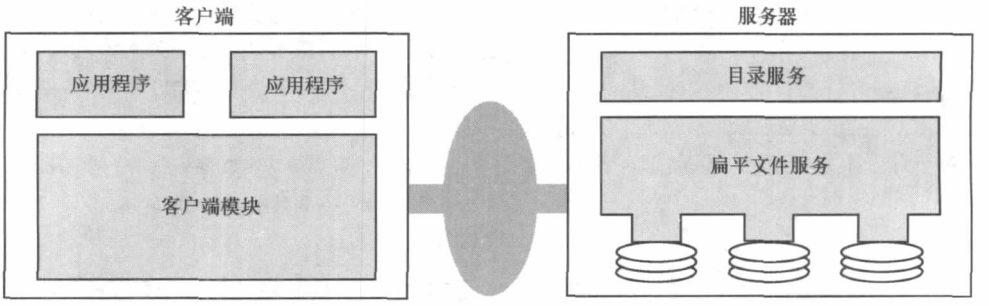

图 1　分布式文件系统的文件服务模型

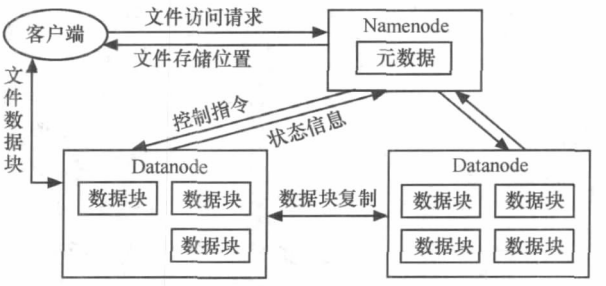

图 2　HDFS 的体系架构

的文件内容的操作，负责创建、删除文件以及读写文件的内容和属性，为每个文件创建惟一的文件标识符供后续操作时引用；目录服务实现的是文件的文本名字与其对应标识符之间的映射，负责目录的创建、删除以及目录中文件的增删和查找，生成的目录也是以文件方式保存并由扁平文件服务负责管理；客户端模块则是运行在客户端上，负责封装对扁平文件服务和目录服务的访问，提供了从客户端本地文件系统的文件操作接口到远程服务器的相关功能调用的映射。

扁平文件服务和目录服务向客户端模块提供高效、基于网络通信的调用接口，这些接口的功能能够组成完备的文件操作集合。客户端模块将这些操作接口进行进一步封装，再以与常规的本地文件操作接口接近的方式提供给应用程序，尽量减少远程文件存放对用户应用程序执行造成的影响，实现透明化。

3 HDFS 分析与建模

与之前的分布式文件系统相比，HDFS 的设计与实现更着重考虑了如何对海量数据进行高可扩展和高可靠的存储和管理，其体系架构如图 2 所示[1]。

HDFS 体系架构采用了主—从方式。其中，作为惟一主节点的 Namenode 负责对文件系统树和文件、目录等元数据信息进行管理和维护，以提供统一的文件名字空间；作为从节点的数量众多的 Datanode 除了具备基本的存储能力外，还具有计算能力以对节点自身携带的存储资源进行管理。在 HDFS 中，一个文件被划分成一个或多个数据块并被分散存储在不同的 Datanode 上，每个数据块都可以通过 Datanode 之间的互相复制而具有多个备份。当客户端访问文件时，首先把相关的包含了文件文本名字的申请发送给 Namenode，然后，Namenode 将相关的元数据信息(主要是文件数据块在 Datanode 上的存储位置) 反馈给客户端，进而客户端直接和相应的 Datanode 建立连接并进行具体的文件操作。Datanode 定期将自身的状态 (如当前保存的文件数据块信息)提交给 Namenode，并接受 Namenode 的管控，例如实施热点文件数据块的复制和迁移。

根据 HDFS 的体系架构描述并基于图 1 所示文件服务模型，可以建立 HDFS 的模型，如图 3 所示。

在图 3 所示的模型中，服务器侧的目录服务主要由 Namenode 提供，负责对文件系统的名字空间进行操作，如打开、关闭、重命名文件和目录等，还需要管控数据块与 Datanode 之间的映射关系。服务器侧的扁平文件服务主要由 Datanode 提供，负责处理客户端发来的文件读写请求，执行数据块的创建、删除等操作以及根据 Namenode 的指令进行节点间的数据块复制等。客户端模块则主要负责向应用程序提供基于 Java 语言的文件访问功能接口，并处理应用程序的文件操作请求和从服务器侧返回的响应。模型中各个部分之间的通信包括两类：一类是远程过程调用，主要用于客户端与 Namenode、Datanode 与 Namenode 之间的通信，如客户端申请的发送与反馈、Datanode 状态信息的提交等；另一类是基于流的数据传输，用于客户端与 Datanode 之间的文件数据块传递。

从模型的角度出发，可以分析出 HDFS 在设计和实现上的主要特征。

- 采用专用的服务器提供目录服务。Namenode 对文件元数据进行管理，能够维护统一的文件名字空间供用户访问以及从全局上对系统进行控制，提高了系统的透明性和可扩展性；同时，Namenode 不承担文件内容的供给，减轻了节点压力。
- 采用数目众多的服务器提供扁平文件服务。多个 Datanode 可以同时为用户提供文件数据块服务，它们分布广泛并互为备份，提高了系统在节点级的可靠性，因此，单个节点可以由普通的 PC 服务器担当，有利于降低系统成本。

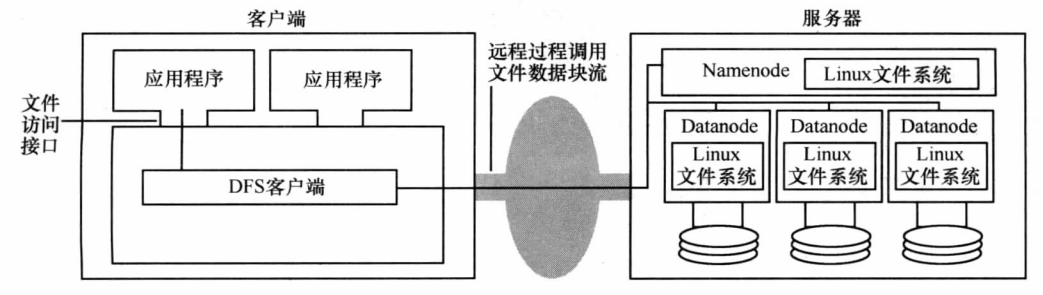

图 3 HDFS 的文件服务模型

· 采用文件数据分块和数据块复制机制。文件数据被划为多个数据块，有利于对其内容进行查找和定位，同时数据块的多个拷贝能够提高系统在文件级和数据块级的可靠性；同时，分布在不同 Datanode 上的数据块可以被并行访问，改善了访问性能。

· 采用多种通信机制。根据通信对象和传输内容的不同，分别提供了远程过程调用和数据流两种通信方式，实现了带外控制，提高了访问性能。

4 文件系统比较

NFS(network file system)[3]是最早商用并流行至今的分布式文件系统。NFS 遵循了如图 1 所示的分布式文件系统模型，在其系统组成中，每台服务器都部署了 NFS 的客户端模块与服务器侧的服务模块，因此 NFS 的客户端和服务器是对称的，即它们都可以成为访问远程文件的客户端或者是提供远程文件访问服务的服务器，以此实现集群内文件资源的共享。NFS 的客户端与服务器之间通过无状态的 NFS 协议进行通信，这是一组用于提供给用户通过网络对远程服务器上的目录和文件进行操作的远程过程调用。在实际应用中，需要访问远程文件的客户端对远程服务器的主机名、远程文件目录的路径及其在客户端本地显示的名称等信息进行设置，然后通过将远程服务器上的文件系统目录挂接到本地文件系统的方式，实现在本地对远程文件的访问。在服务器侧，目录服务和扁平文件服务被统一提供，例如，在文件被创建的时候，其文本名字及所在目录将作为参数一起传递给服务器。NFS 在实际应用中获得了业界的广泛支持，其中用于通信的 NFS 协议已经成为国际互联网标准。虽然它的初始设计是基于 UNIX 操作系统的，但是当前已经被集成在多种操作系统的内核中。

NFS 体现了之前主流分布式文件系统的典型特征，HDFS 则代表了新兴的面向云计算的分布式文件系统，它们在满足系统设计需求方面各有侧重，具体的比较见表 2。

HDFS 很好地满足了系统设计的透明性需求，在系统的扩展性、可靠性等方面具有优势，而这也正是云计算对文件服务的主要需求，因此，HDFS 以及与之具有相似设计思想的 GFS、KFS(KOSMOS distributed file system)等分布式文件系统在当前发挥了巨大的作用并拥有广阔的前景。以这类新兴的分布式文件系统为基础，分布式计算模型、分布式数据库等云计算关键技术也有了新的发展，例如，MapReduce 计算架构能够部署在 HDFS 系统上实现将计算步骤推向数据节点的计算模式，HBase 分布式数据库可以借助 HDFS 的管理能力而无需传统的数据库管理系统。

表 2 NFS 和 HDFS 的比较

		NFS	HDFS
系统透明性	访问透明	提供与传统的本地文件系统访问类似的接口，实现访问透明	未提供与传统的 POSIX 完全兼容的接口，访问透明性不高
	位置透明	远程文件系统的目录直接挂接在客户端文件系统上，支持位置透明	通过文件文本名字和统一资源标识符访问文件，支持位置透明
	移动透明	没有统一的名字空间，文件不能在服务器侧移动，不支持移动透明	Namenode 节点管理和维护统一的文件名字空间，实现移动透明
	性能透明	提供服务器侧和客户端的缓存机制，文件访问性能较高	支持多个文件数据块的并行访问，文件访问性能高
	扩展透明	需要用户手工设置远程文件访问的挂接位置，扩展不够透明	根据负载规模增删 Datanode，无需用户参与，实现扩展透明
文件并发更新		使用网络文件锁管理机制进行并发控制，但会受 NFS 协议的影响	不支持文件的并发更新，采用租期机制限定对文件数据块的访问
文件内容复制		读文件可被复制到多台服务器上，但是不支持可写文件的复制	文件块被自动复制为多份拷贝并被分散存放到多个 Datanode 上
文件数据一致		采用单一拷贝机制，定时审验客户端缓存中的数据信息	采用简单的一致性协议，通过原子化操作对文件进行修改
文件服务容错		采用 NFS 无状态协议，异常处理和容错功能完全由客户端实现	在 Datanode、文件、数据块等多个层次上提供冗余，实现容错
文件服务兼容		支持具有多种操作系统和硬件平台的客户端和服务器	支持运行 Linux 操作系统的标准 PC 服务器作为客户端和服务器
文件服务安全		采用安全的远程过程调用机制保证用户权限识别及数据安全传输	采用基于安全套接字的网络安全机制实现文件的安全访问

但是,从表 2 中也可以发现 HDFS 当前存在的一些问题, 如当文件服务的负载过大时,NFS 能够以增加新的服务器的方式缓解文件服务的压力,而 HDFS 虽然可以通过增加 Datanode 对存储容量进行动态扩容,但是系统中惟一的用于管控系统统一名字空间的 Namenode 会成为性能和可用性的瓶颈。对于这一问题有多种改进思路。例如,可以通过部署 Namenode 集群的方式供客户端从多台服务器上获得文件的元数据[4],但是 Namenode 节点间的高效同步机制会增加系统复杂度;还可以采用在线动态生成文件元数据而无需专用管理节点的方式[5],但是其性能和效率还有待提高。

另外,还需要特别注意的是,HDFS 虽然在当前的面向云计算(主要是互联网领域的相关应用)的应用场景中取得了较大的成功,但是它也具有一定的局限性,例如不适合用于进行低延时的文件数据交互等。因此,文件服务的提供者需要根据实际的用户需求,从文件系统的扁平文件服务、目录服务和客户端等方面的技术要点进行综合考察,才能最终选择出真正适用的文件系统。

5 结束语

通过对 HDFS 提供文件服务的体系架构进行建模分析,可以发现,无论是在服务器侧提供的扁平文件服务和目录服务,还是在客户端运行的客户端模块,HDFS 在设计和实现时都充分考虑了海量数据在存储和管理等方面的需求,并最终获得了系统的高可扩展性、高可靠性和高性能,满足了云计算领域的相关用户需求。因此,HDFS 一直以来都受到了学术界和产业界的广泛关注,拥有广阔的应用前景。

但是,通过对比、分析也可以发现,HDFS 在一些方面仍然有待完善,可以向传统的分布式文件系统借鉴相关的方法和技术。同时,HDFS 也不是万能的,文件服务的提供者必须根据实际需要选择合适的文件系统。

参考文献

1 Hadoop community. Hadoop distributed file system, http://hadoop.apache.org/hdfs, 2010

2 George C, Jean D, Tim K. Distributed systems: concepts and design(3rd Edition). Addison-Wesley Publishers Limited, 2000

3 Russel S, David G, Steve K, *et al*. Design and implementation of the Sun network file system. Artech House, 1988

4 Wang F, Qiu J, Yang J, *et al*. Hadoop high availability through metadata replication. In: Proceeding of the First International Workshop on Cloud Data Management, Hong Kong, China, November 2009

5 Gluster community. Gluster file system, http://www.gluster.org, 2010

Modeling and Analysis of Hadoop Distributed File System

Wang Feng, Lei Baohua

(China Telecom Corporation Limited Beijing Research Institute, Beijing 100035, China)

Abstract Hadoop distributed file system(HDFS) is inspired by the Google file system paper. It gets great attention and is applied widely. In this paper, from a theoretical perspective of system design, we study user requirements and modeling method of distributed file system, and create a model for analyzing the architecture characteristics of HDFS. Through the comparison of HDFS and traditional distributed file system, several advantages of HDFS (e.g. high scalability and high availability) are identified. This paper is helpful to research the design principles of HDFS more deeply, and provides some advices for design and implementation of Cloud Computing-oriented distributed file system.

Key words hadoop distributed file system, system modeling, cloud computing (**收稿日期**: 2010-11-29)