**GFS**

**Chunk：文件都被分割成固定大小的Chunk**

**Chunkserver：Chunk服务器把Chunk以linux文件的形式保存在本地硬盘上，并且根据指定的Chunk标识和字节范围来读写块数据。**

Chunk服务器把Chunk以linux文件的形式保存在本地硬盘上，并且根据指定的Chunk标识和字节范围来读写块数据。

**Master：GFS分布式文件系统中的总控节点。**

* 执行所有的namespace操作。
* 管理整个系统里所有Chunk的副本
* 创建新Chunk和它的副本
* 决定Chunk的存储位置
* 在所有的Chunk服务器之间进行负载均衡
* 协调各种系统活动以保证Chunk被完全复制
* 回收不再使用的存储空间

Master节点使用**心跳信息**周期地和每个Chunk服务器通讯，发送指令到各个Chunk服务器并接收Chunk服务器的状态信息

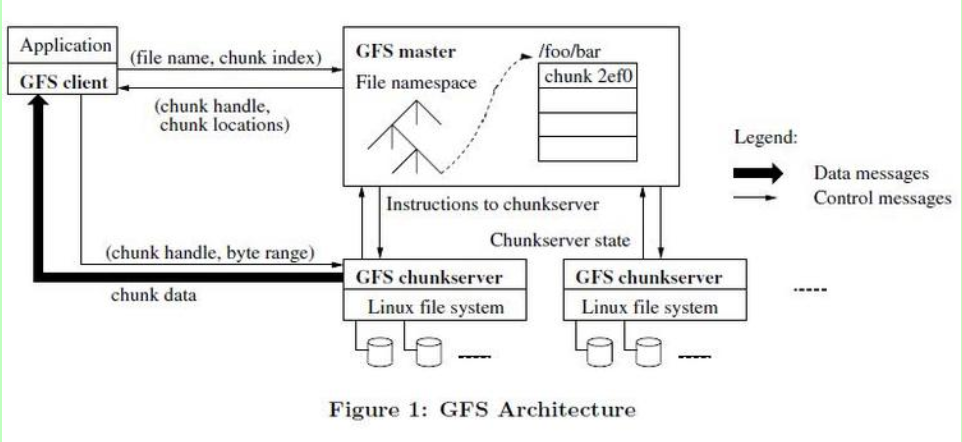
Master存储3种主要类型的**元数据**：

文件和Chunk的命名空间

文件和Chunk的对应关系

每个Chunk副本的存放地点

**Client:** 客户端和Master节点的通信**只获取元数据**，所有的数据操作都是由客户端直接和Chunk服务器进行交互的.如下所示：



**灾难恢复：**

**checkpoint，减少重演操作的日志量。恢复：Master从磁盘上读取最新的Checkpoint文件，重演Checkpoint之后的有限个日志文件便可恢复系统**

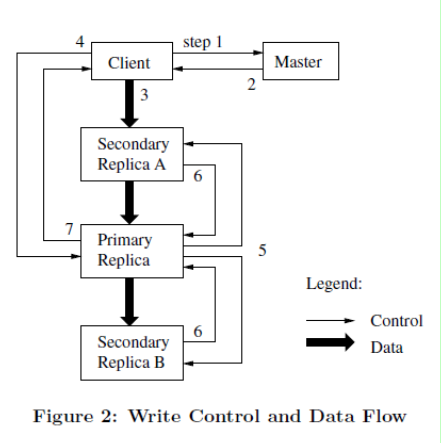
**简单读流程：**

1）客户端将文件名和程序指定的字节偏移根据固定的Chunk大小转换成文件的Chunk索引，进而把文件名和Chunk索引发送给Master节点。

2）Master节点将相应的Chunk标识和副本的位置信息发还给客户端。客户端用文件名和Chunk索引作为key缓存这些信息，之后客户端发送请求（包含Chunk的标识和字节范围）到其中的一个副本Chunkserver处（一般会选择最近的）。

3）该Chunkserver读取chunk数据返回客户端。

**写数据：**



1)客户机向Master节点询问哪个Chunk服务器持有当前租约，以及其它副本的位置。

**如果没有Chunk持有租约，Master就选择其中一个副本建立一个租约**

2）Master将主Chunk的标识符以及其它副本（secondary副本/二级副本）的位置返回给客户机。客户机缓存这些数据以便后续操作。

**此后只有在主Chunk不可用或主Chunk回复信息表明它已不再持有租约时，客户机才重新联系Master节点。**

3）客户机把数据推送到所有的副本上（可按任意的顺序推送）。

**Chunk服务器接收到数据并保存在它的内部LRU缓存中，直到数据被使用或过期交换出去**

4）客户机发送写请求到主Chunk服务器。主Chunk为接收到的所有操作分配连续的序列号（**这些操作可能来自不同的客户机，序列号保证了操作顺序执行**），它以序列号的顺序把操作应用到它自己的本地状态中（**主Chunk顺序执行这些操作，并更新自己的状态**）

5）主Chunk把写请求传递到所有的二级副本。每个二级副本依照主Chunk分配的序列号以相同的顺序执行这些操作。

6）所有的二级副本回复主Chunk“它们已经完成了操作”

7）主Chunk服务器回复客户机（任何副本产生的任何错误都会返回给客户机）。

出现错误时，写操作可能在主Chunk和一些二级副本执行成功。（如果主Chunk上操作失败，操作不会被分配序列号，也不会传递）客户端请求被确认为失败，被修改的region处于不一致状态。客户机代码通过重复执行失败的操作来处理这样的错误。在从头开始重复执行之前，客户机会先尝试几次步骤（3）~（7）。