FastDFS:

什么是分布式:

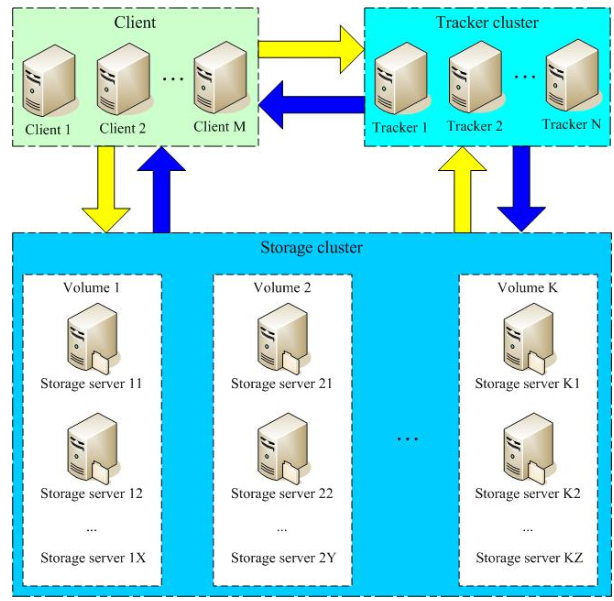
引:单纯通过增加硬盘个数来扩展计算机文件系统的存储容量的方式，在容量大小、容量增长速度、数据备份、数据安全等方面的表现都差强人意。分布式文件系统可以有效解决数据的存储和管理难题：将固定于某个地点的某个文件系统，扩展到任意多个地点/多个文件系统，众多的节点组成一个文件系统网络。每个节点可以分布在不同的地点，通过网络进行节点间的通信和数据传输。人们在使用分布式文件系统时，无需关心数据是存储在哪个节点上、或者是从哪个节点从获取的，只需要像使用本地文件系统一样管理和存储文件系统中的数据。

FastDFS能干什么?

为互联网量身定制，充分考虑了冗余备份、负载均衡、线性扩容等机制，并注重高可用、高性能等指标.使用FastDFS很容易搭建一套高性能的文件服务器集群提供文件上传、下载等服务.

FastDFS 配置略(网上很多):

重点FastDFS 原理:



从图中可以看出,存在tracker和storage两种角色,tracker扮演的就是服务中心,storage 扮演的就是实际工作者.

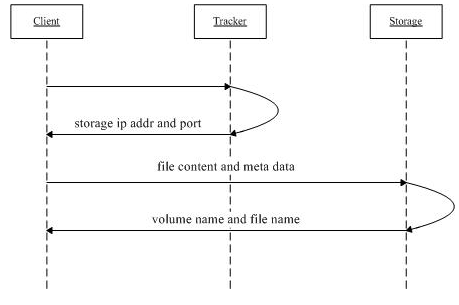
FastDFS中的文件标识分为两个部分：卷名和文件名，二者缺一不可。

存储的节点采用了分组group方式,一个group 就是一小部分存储的空间,多个空间累加起来就是整个服务器中存储的文件容量.group之间是互相独立的.一个group可以由一台或者台存储服务器组成,一个group下的存储下的文件都是相同的.多台group起到容灾,备份的效果.

Tracker:服务中心定时和存储对象storage 进行交互,以便掌握storage目前的状态,同时通过中心实现负载均衡和线性扩容,想想看假如我们的存储都达到了上限,目前想要扩容怎么办?最快的方法必然要添加新的服务者storage,将新的storage服务正确加入后,因为定时交互, 所以tracker可以很快的捕捉,并向外提供.

文件上传:

当主机像用户上传时,用户会向服务中心询问,可有工作者可以接收这份工作,服务中心就会根据当前storage自身的状态进行判断,以分配合适的工作者给我们使用,即将可用的工作者联系方式给我们,由我们自己去沟通联系,当然如果服务中心配置有权重的话,也会在合适的情况下优先推荐工作者(有点类似rpg游戏,优先给你主线任务)具体调度时序图如下:



文件上传时序图

过程如下:

 client询问tracker上传到的storage，不需要附加参数；()

2. tracker返回一台可用的storage；

3. client直接和storage通讯完成文件上传。(传送文件信息,以及元数据信息,所谓的元数据信息是指**描述其它数据的数据.比如描述一个电影库中想要查找到具体的一部电影,通常我们会根据电影的名称,演员名称等,而这些恰恰是电影的组成部分之一.我认为这里的元数据就是文件的名称,文件的类型,文件的上传时间等**)

返回的 group/M00/0C/xxxxxxxxxxxx.xxx

其中 group 是组名,M00是磁盘名,0C是目录名称,xxx.xx是文件名(包含文件类型)

引申:数据元,数据元是最小的数据单位不可分割,说白了就像2进制一样,拆分后系统就不能正确识别了.再比如我们的汉字,拆成笔画后还能在成点,点还能拆?但我们也要明白单独的汉字或者二进制数是不能真正知道意思的通常要由其他的内容辅助,才能知道真正的含义.(更多内容查阅百度吧..深究不好,大致理解就行)

完整的数据元名称=对象类术语+特征类术语+表示类术语+（限定类术语）

其中：

**一个数据元有且仅有一个对象类术语(object class)**：是现实世界或抽象概念中事物的集合，有清楚的边界和含义，因其特性和行为遵循同样的规则而能加以标识。

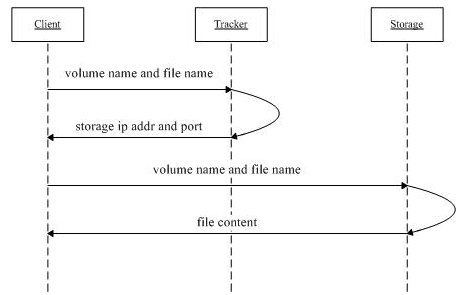
**一个数据元有且仅有一个特征类术语(property)**：是对象类的所有个体所共有的某种性质，也是对象有别于其他成员的依据，是用来区分和识别事物的一种手段。特征类术语是任何一个数据元名称所必须的成分，在数据元概念可完整、准确、无歧义表达的情况下，其他术语可以酌情简略。

**一个数据元有且仅有一个表示类术语(representation)**：是值域、数据类型和表示方式的组合，也包括计量单位和字符集等信息。当表示类术语与特征类术语有重复或部分重复时，可从名称中将冗余词删除。

**限定类术语由专业领域给定，限定类术语是可选的。**

文件下载时:

客户机会向tracker提出请求后,tracker会根据客户机锁提供的fileid(其中包含有:group信息,文件名称等.)tracker 找到对应的端口地址返回给我们,我们就知道具体的位置,然后获取便可.



过程说明:

1. client询问tracker下载文件的storage，参数为文件标识（卷名和文件名）；

2. tracker返回一台可用的storage；

3. client直接和storage通讯完成文件下载。

需要说明的是，client为使用FastDFS服务的调用方，client也应该是一台服务器，它对tracker和storage的调用均为服务器间的调用。

其他说明:

配置文件重点部分详解:

tracker.conf:

# 提供服务的端口

# the tracker server port

port=22122

具体路径地址

# the base path to store data and log files

base\_path=/home/yuqing/fastdfs

# base\_path 目录地址(根目录必须存在,子目录会自动创建)

# 附目录说明:

# tracker server目录及文件结构：

# ${base\_path}

# |\_\_data

# | |\_\_storage\_groups.dat：存储分组信息

# | |\_\_storage\_servers.dat：存储服务器列表

# |\_\_logs

# |\_\_trackerd.log：tracker server日志文件

# 用于提供http服务的端口

# HTTP port on this tracker server

http.server\_port=8080

storage.conf:

# base\_path 目录地址,根目录必须存在 子目录会自动生成 (注 :这里不是上传的文件存放的地址,之前是的,在某个版本后更改了)

# the base path to store data and log files

base\_path=/home/yuqing/fastdfs

# 逐一配置store\_path个路径，索引号基于0。注意配置方法后面有0,1,2 ......，需要配置0到store\_path - 1。

# 如果不配置base\_path0，那边它就和base\_path对应的路径一样。

# store\_path#, based 0, if store\_path0 not exists, it's value is base\_path

# the paths must be exist

store\_path0=/home/yuqing/fastdfs

#store\_path1=/home/yuqing/fastdfs2

#如果有多个则配置多个tracker

tracker\_server=你的服务器公网ip:22122(如192.168.1.1:22122)

FastDFS中，文件更新操作都会优先选择源Storage server，也就是该文件被上传到的那台Storage server。这样的做法不仅避免了文件同步延迟的问题，而且有效地避免了在多台Storage server上更新同一文件可能引起的时序错乱的问题。

Storage server生成的文件名中，包含了源Storage server的IP地址和文件创建时间等字段。文件创建时间为UNIX时间戳，后面称为文件时间戳。从文件名或文件ID中，可以反解出这两个字段。

文件同步采用主动推送的方式。另外，每台*storage server*都会定时向*tracker server*报告它向同组的其他*storage server*同步到的文件时间戳。当*tracker server*收到一台*storage server*的文件同步报告后，它会依次找出该组内各个*storage server*（后称作为*S*）被同步到的文件时间戳最小值，作为*S*的一个属性记录到内存中。

配置文件详情说明:

https://blog.csdn.net/xingjiarong/article/details/50752586