# 设计文档

## 实现

本次lab1利用已有的windows文件系统，通过程序在对应的”bm-xx”文件夹（对应一个block manager）下创建”xx.data”，”xx.meta”文件来表示block。block有size、indexId、blockmanager三个成员。而每个block manager对应的文件夹下还有一个id\_count文件来记录block的下一个indexId。

而Block manager主要通过属性block的arraylist来记录属于本manager的block。newBlock()函数调用newEmptyBlock()函数，再将信息写入block对应的data和meta文件。

File在读取信息的时候使用read()函数，通过meta文件找到各个block以汇总信息。write()函数利用StringOperation中对于byte[]的分割操作，变成二维数组，从而得到与块等大的单个byte[]，进而写入对应block。File中需要维护的指针使用pos属性来保持。①读取的时候，获取指针pos，接着根据length获取对应数量block的所有bytes。最后在根据pos和length对这个byte[]进行裁剪以获得最终read的结果。②写入数据的时候同样获取pos，计算出需要变更的block，由于block内容不可变，所以，从pos所在block开始往后的块从meta文件中删去，但不去理会，而再创建新的block将byte写入。写入完成后将meta信息进行修改。

File manager只需要负责索引和创建File能力即可，实现比较简单。

## 接口接入

①为实现接口中的方法，我创建了FileUtil类，主要在已有文件系统层级提供读写的便利。②另外，我还创建了StringOperation类来帮助处理一些字符串和byte数组。③在File接口的实现中，我另外增加了以下几个方法：getFileMetaStr()、getBlkMngName(String str)、getBlkIdStr(String str)、getValidBlk(String lineData)、randomCommon(**int** min, **int** max, **int** n)。

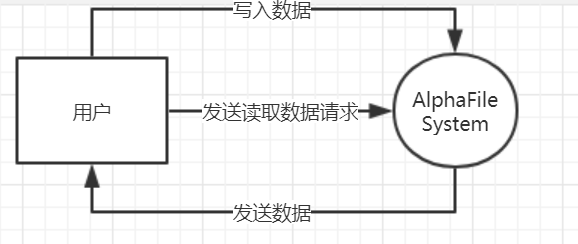
其中getFileMetaStr()和getBlkMngName(String str)、getBlkIdStr(String str)主要用于获取文件保存File的meta信息的文件路径和根据其中的信息进行分割获得block manager name和block id对应的字符串。getValidBlk(String lineData)用于根据FileMeta内容寻找checksum一致、对应block manager存在且对应block存在其中的Block，否则返回null。randomCommon(**int** min, **int** max, **int** n)生成两个随机数，用于Dupication的时候随机选择两个block manager存储对应data。

有了以上的一些辅助类，我们在实现block的read函数的时候就可以调用FileUtil的相关静态函数直接读取。实现block manager的newEmptyBlock(int blockSize)函数的时候首先维护一个id\_count文件和自身的blocks数组，创建对应data和meta文件。newBlock(byte[] b)调用上述方法后操作FileUtil写入b中内容到对应文件，并修改meta信息。实现File则需要维护一个光标，read（int length）和write（byte[] b）需要充分利用光标来实现一些细节，不过好在前面提到的三个辅助类以及为此做好了准备。

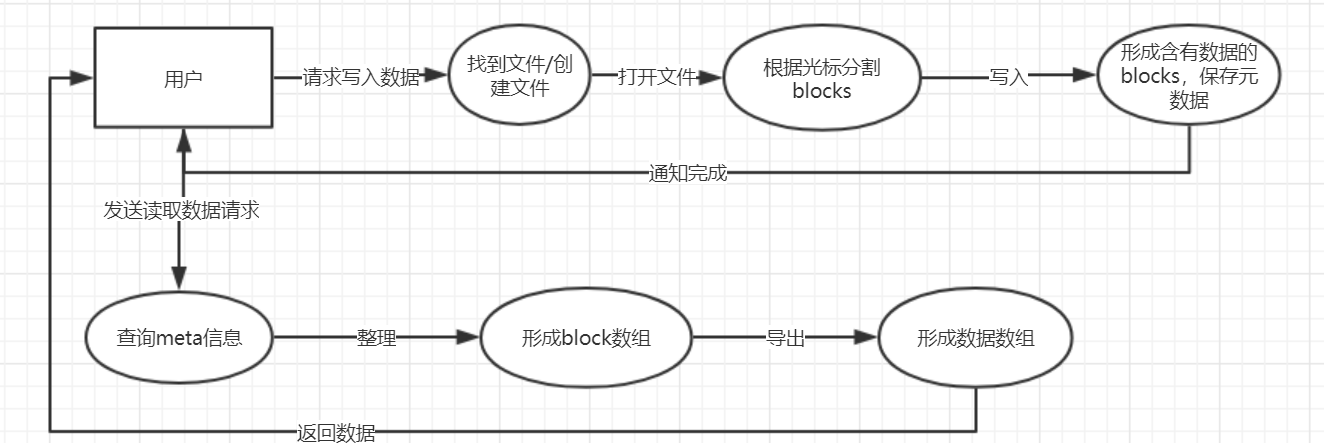
## Scene

在File中读取数据首先调用java.io.File来访问某个File对象所属的meta文件，读取元数据。根据元数据判断所要读取的字节数是否超出，并进行调整（如果超出则改为最大值）。根据logic block的信息组成一个block数组，读取所需长度的字节并返回。

0级流程图：



1级流程图：



2级流程图

