

分布式共享内存系统

姓名	班级	学号	日期
刘俊杰	103班	21307174	2023.12.25

正文：

1.题目要求

题目：

设计一个分布式文件系统。该文件系统可以是client-server架构，也可以是P2P非集中式架构。要求文件系统具有基本的访问、打开、删除、缓存等功能，同时具有一致性、支持多用户特点。在设计过程中能够体现在分布式课程中学习到的一些机制或者思想，例如Paxos共识、缓存更新机制、访问控制机制、并行扩展等。实现语言不限，要求提交代码和实验报告，实验报告模板稍后在课程网站下载，提交时间为考试之后一周内。

基本要求

基本要求	完成情况
1. 选择编程语言，推荐使用Python或Java	[√]
2. 使用RPC模式进行文件系统节点之间的通信	[√]
3. 实现基本文件操作模型，包括创建、删除、访问等功能	[√]
4. 客户端具备缓存功能，可在本地存储中搜索文件信息	[√]
5. 数据创建多个副本，且多个副本分布在不同物理机器上	[√]
6. 多副本之间能够保持一致性	[√]
7. 支持多用户，文件可以并行读写（包含文件锁）	[√]
8. 提供测试命令或测试用例，并有截屏或video支持	[√]

加分项

加分项	完成情况
1. 加入缓存更新算法	[√]
2. 使用Paxos共识方法或主副本选择算法	[√] (实现了主副本选择算法)
3. 实现访问权限控制	[√]
4. 其他高级功能	[√] (实现了利用多线程来支持多个副本服务器运行)

2.解决思路

(1)使用RPC模式进行文件系统节点之间的通信

为使用RPC模式进行文件系统节点之间的通信,我采用 XML-RPC 协议进行远程过程调用（RPC）通信。**XML-RPC 通信：**

- **原理：** XML-RPC 是一种远程过程调用协议，通过 XML 编码的消息进行通信。客户端可以调用服务器上注册的远程方法，实现远程调用。
- **技术：** 使用 `SimpleXMLRPCServer` 创建 XML-RPC 服务器，并通过注册服务类的实例，使得客户端可以调用服务中的方法。

(2)实现基本文件操作模型，包括创建、删除、访问等功能

实现基本文件操作模型的代码中，运用到了 RPC（远程过程调用）作为通信模式，使得客户端可以调用服务器上的远程过程来实现文件操作。

1. **RPC框架实现远程调用：** XML-RPC 提供了远程调用的机制，使得客户端可以调用服务器上的远程过程。

2. 文件操作：

- 使用 Python 提供的文件操作函数，包括文件的读取、写入、删除以及文件夹的删除。将文件操作封装为 RPC 服务，定义了文件服务的接口，包括上传文件、写文件、读文件、删除文件等操作。这样，客户端可以透明地调用远程服务器上的函数，就像调用本地函数一样。

(3)客户端具备缓存功能，可在本地存储中搜索文件信息

客户端具备缓存功能，缓存中记录了文件的内容、修改时间和上次读时间。

- 若需要读取的文件在缓存中有,则通过服务器获取上次该文件的修改时间，若得到的修改时间不大于缓存中的修改时间，则说明缓存中的文件是最新的，可以直接从缓存中读，不需要请求服务器发送文件内容
- 若需要将文件加入缓存，则需要将文件的内容、修改时间和上次读时间记录到缓存中，若需要更新换出某个文件时，可以依据上次读时间进行LRU缓存更新。

(4)数据创建多个副本

设计了主服务器来管理多个副本,客户端可以先和主服务器器连接后，选择自己需要的副本服务器来提供服务。

(5)多副本之间能够保持一致性

每次任意一台副本服务器下的副本发生变化时，需要先执行副本一致性操作，才能允许客户端继续操作。

(6)支持多用户，文件可以并行读写（包含文件锁）

文件锁机制：

- **原理：** 采用文件锁机制确保同一时刻只有一个线程能够对文件进行写操作，避免多个线程同时写入导致的数据不一致问题。
- **技术：** 使用 `FileLock` 类实现文件锁的获取和释放，利用 `msvcrt.locking` 函数实现文件锁。

(7)加入缓存更新算法

这里加入的缓存更新我选择的是LRU缓存更新算法。在缓存中记录每个文件上次读的时间，每次需要缓存更新且缓存不够时，将上次读时间最早的文件移出缓存，将需要更新的文件加入缓存。

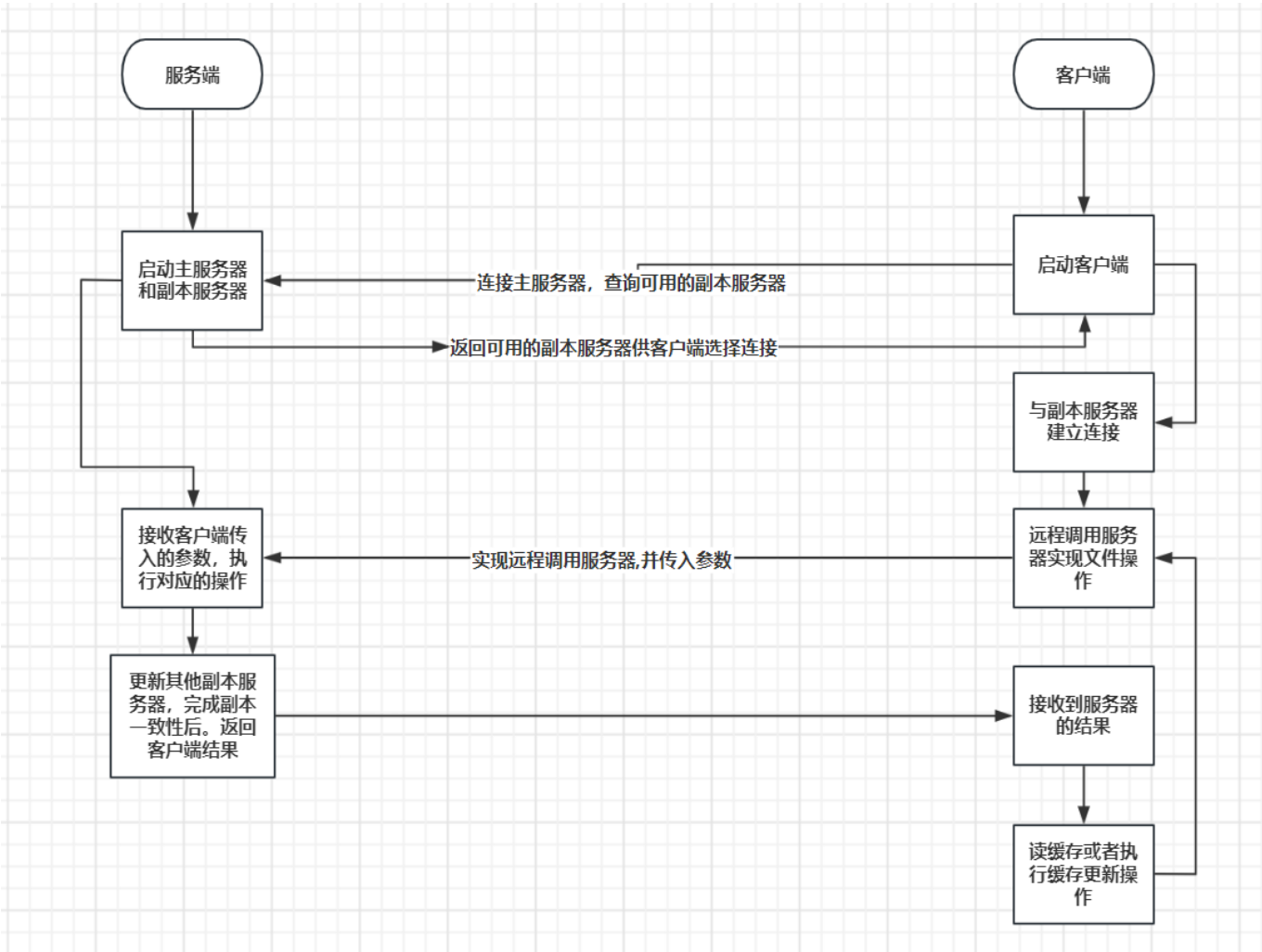
(8)主副本选择算法

因为是在一台机器上运行的文件系统，无法考虑地理位置、通信等因素对客户端选择副本的影响，所以我实现了让客户端自主选择主副本的功能：

- 1.首先使得客户端与主服务器建立RPC连接，主服务器向客户端显示有哪些副本可供选择。
- 2.接着，客户端可输入想选择副本对应的名称。
- 3.最后，将客户端与其选择的副本建立RPC连接来实现主副本选择。

3.实现细节

流程图



细节实现

(1)主副本选择算法

因为是在一台机器上运行的文件系统，无法考虑地理位置、通信等因素对客户端选择副本的影响，所以我实现了让客户端自主选择主副本的功能：

- 1.首先使得客户端与主服务器建立RPC连接，主服务器向客户端显示有哪些副本可供选择。
- 2.接着，客户端可输入想选择副本对应的名称。
- 3.最后，将客户端与其选择的副本建立RPC连接来实现主副本选择。

```
class FileClient: # 客户端
    def __init__(self, server_url):
        # 首先与主服务器连接
        self.server = ServerProxy(server_url)
        self.client_port = self.get_client_port()
        # 获得可用的副本服务器
        replica_dict = self.server.get_replica()
        print("Here are the services with ports:")
        for server, port in replica_dict.items():
            print(f"Server: {server}, Port: {port}")
        # 选择可用的副本服务器并进行连接
        self.replica = input("Choose the name of server you want to connect: ")
        while self.replica not in replica_dict:
            print("This server not Found.")
            self.replica = input("Choose the name of server you want to connect: ")
    )

    replica_port = replica_dict[self.replica]
    server_url = 'http://localhost:' + str(replica_port) + '/RPC2'
    self.server = ServerProxy(server_url)
    # 初始化缓存
    self.filecache = Cache()
```

(2)文件操作

客户端输入命令,并执行对应的操作:

```
def handle(self): # 处理输入的命令
    while True:
        command = input("Enter a command (Type 'help' for available commands): ")
    )

    if command == 'help':
        self.help()
    elif command.startswith('upload_file'):
        _, filename = command.split(' ')
        result = self.upload_file(filename)
        print(result)
    elif command.startswith('download_file'):
        _, filename = command.split(' ')
        result = self.download_file(filename)
        print(result)
    elif command.startswith('delete_file'):
        _, filename = command.split(' ')
        result = self.delete_file(filename)
        print(result)
```

```

elif command.startswith("delete_folder"):
    _, foldername = command.split(" ", 1)
    result = self.delete_folder(foldername)
    print(result)
elif command.startswith("write_file"):
    _, filename = command.split(" ", 1)
    result = self.write_file(filename)
    print(result)
elif command == "list_files":
    self.list_files()
elif command.startswith('read_file'):
    _, filename = command.split(' ')
    self.read_file(filename)
elif command.startswith('port'):
    print(f"Client's local port: {self.client_port}")
    break
elif command.startswith('cache'):
    self.filecache.show_cache()
elif command == "exit":
    print("Exiting the client.")
    break
else:
    print("Invalid command. Type 'help' for available commands.")

```

服务器将文件操作封装为 RPC 服务，客户端可以透明地调用远程服务器上的函数。服务器上的功能:

```

def list_files(self, folder=None): # 列出文件夹和文件
    .....

def upload_file(self, filename, content, updatetime): # 客户端上传文件
    .....

def write_file(self, filename, content, updatetime): # 客户端写文件
    .....

def download_file(self, filename): # 客户端下载文件
    .....

def delete_file(self, filename): # 客户端删除文件
    .....

def delete_folder(self, foldername): # 客户端删除文件夹
    .....

def read_file(self, filename): # 客户端读文件
    .....

```

(3) 客户端具备缓存功能，可在本地存储中搜索文件信息

客户端具备缓存功能，缓存中记录了文件的内容、修改时间和上次读时间。

- 若需要读取的文件在缓存中有,则通过服务器获取上次该文件的修改时间,若得到的修改时间不大于缓存中的修改时间,则说明缓存中的文件是最新的,可以直接从缓存中读,不需要请求服务器发送文件内容
- 若需要将文件加入缓存,则需要将文件的内容、修改时间和上次读时间记录到缓存中,若需要更新换出某个文件时,可以依据上次读时间进行LRU缓存更新。代码实现:

```
def read_file(self, filename): # 读文件
    # 先比较缓存内是否有 且 是否为最新修改过的 如果满足直接读缓存
    filetype = self.filecache.read_file_time(filename)
    if filetype is not False:
        update_time = self.server.get_file_update_time(filename)
        if update_time is not False and update_time <= filetype:
            content = self.filecache.read_file_content(filename)
            print("Reading from cache")
            print(f"Content of '{filename}':\n{content}")
            return
    # 没有则读服务器
    updatetime,content = self.server.read_file(filename)
    if content is not False:
        print(f"Content of '{filename}':\n{content}")
        self.filecache.update_file_LRU(filename,content,updatetime) # 更新缓存
    else:
        print(f"Error: Unable to read file '{filename}' from the server.")
```

(4)数据创建多个副本,利用多线程来启动主服务器和副本服务器服务

我用主服务器来管理(为操作方便,设置主服务器没有文件、不执行文件操作,只负责管理副本),多线程来启动主服务器和副本服务器服务

```
# 创建主服务器
main_server_thread = threading.Thread(target=run_server, args=
(SimpleXMLRPCServer, 'localhost', 8000,"",True))
main_server_thread.start()

# 创建副本服务器1
replica1_thread = threading.Thread(target=run_server, args=
(SimpleXMLRPCServer, 'localhost', 8001,"server1",False))
replica1_thread.start()

# 创建副本服务器2
replica2_thread = threading.Thread(target=run_server, args=
(SimpleXMLRPCServer, 'localhost', 8002,"server2",False))
replica2_thread.start()
```

(5)多副本之间能够保持一致性

每当服务器下的文件发生变化时,调用以下的函数来对其他副本服务器的文件做出同步的修改,一致性完成后才能返回给客户端。

```
def replica_consistency(self, server, file, delete, content=None): # 执行保证副本一致性操作
    if delete:
        print(server, file)
        for server_tmp, addr in replica_addr_dict.items():
            if server_tmp == server:
                continue
            else:
                filename = os.path.join(os.getcwd(), os.path.join(addr,
file))

                print(filename)
                if os.path.isfile(filename):
                    os.remove(filename)
                else:
                    shutil.rmtree(filename)
    else:
        for server_tmp, addr in replica_addr_dict.items():
            if server_tmp == server:
                continue
            else:
                filename = os.path.join(os.getcwd(), os.path.join(addr, file))
                with open(filename, 'w') as file:
                    file.write(content)
```

(6)支持多用户，文件可以并行读写（包含文件锁）

因为实现了rpc远程调用,可以实现多个用户远程调用一个或多个服务器来进行操作,

我使用 FileLock类实现文件锁的获取和释放，利用 msvcrt.locking`函数实现文件锁:

```
class FileLock: # 文件锁
    def __init__(self, folder=None):
        self.directory = os.getcwd()

    def acquire_lock(self, filename): # 获取锁
        lock_file_path = os.path.join(self.directory, f'{filename}.lock')

        while True:
            try:
                lock_file = open(lock_file_path, "w")
                msvcrt.locking(lock_file.fileno(), msvcrt.LK_LOCK, 1)
                return lock_file
            except Exception as e:
                print(f"Acquiring FileLock Error: ({filename}): {e}")
                time.sleep(1) # 等待一段时间后重试

    def release_lock(self, lock_file): #释放锁
        try:
            msvcrt.locking(lock_file.fileno(), msvcrt.LK_UNLCK, 1)
```

```

except Exception as e:
    print(f"Releasing FileLock Error: {e}")
finally:
    lock_file.close()

```

每次写文件之前都要请求锁，若请求不到则会在acquire_lock中继续等待，请求到后执行写操作，写操作执行完后释放锁。

```

def write_file(self, filename, content, updatetime): # 客户端写文件
    filename_tmp = os.path.join(self.server_files_directory, filename)
    lock_file = self.lock.acquire_lock(filename) # 请求锁
    with open(filename_tmp, 'w') as file:
        file.write(content)
    update_file_time(filename, updatetime)
    self.replica_consistency(self.addr, filename, False, content) # 执行保证副本一致性操作
    self.lock.release_lock(lock_file) # 释放锁
    return True

```

(7)缓存更新

利用LRU缓存更新算法，在缓存中记录每个文件上次读的时间，每次需要缓存更新且缓存不够时，将上次读时间最早的文件移出缓存，将需要更新的文件加入缓存。

```

def update_file_LRU(self, filename, content, updatetime): # 更新缓存
    if filename in self.cache: # 更新已在缓存内的文件
        self.cache[filename] = [content, updatetime, time.time()]
        return

    if len(self.cache) < self.size: # 缓存大小足够, 直接加入
        self.cache[filename] = [content, updatetime, time.time()]
    else: # 缓存已满 利用LRU缓存更新算法更新
        min_update_time = time.time()
        min_update_file = None
        for file, ct in self.cache.items():
            if ct[2] < min_update_time:
                min_update_time = ct[1]
                min_update_file = file
        self.cache.pop(min_update_file)
        self.cache[filename] = [content, updatetime, time.time()]

```

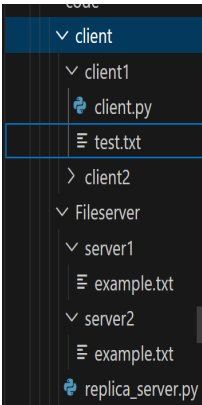
(8)用户权限

用户本地有一个privilege记录用户的权限，权限由服务器分配，每次执行操作前先检查权限，若有权限则执行否则提示不可执行：


```
# 检测权限
if self.check_privilege(command) is False:
    continue
```

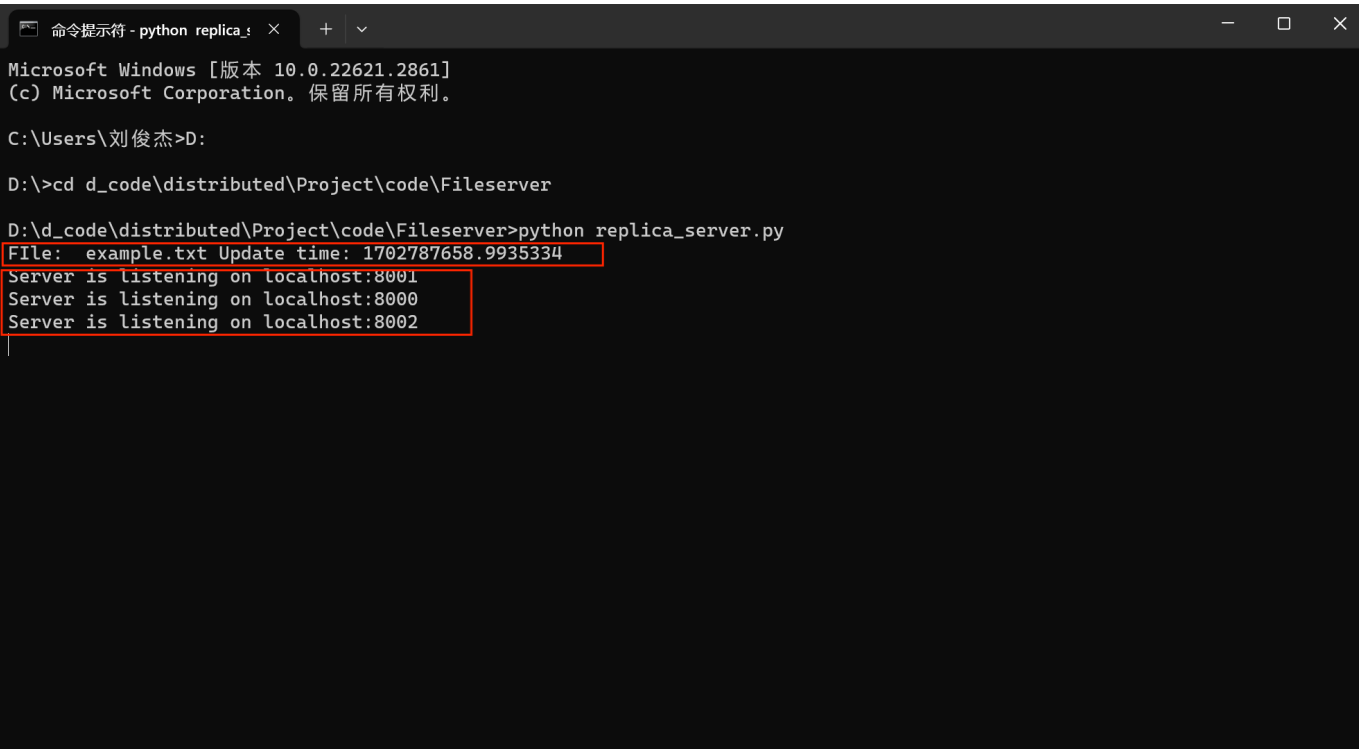
4.运行情况

(1)运行单个客户端测试基本功能



初始文件夹的情况 客户端1下有test.txt 副本服务器server1和副本服务器server2下有example.txt

运行客户端



可以看到主服务器和副本服务器已启动

运行客户端

命令提示符 - python replica_

Microsoft Windows [版本 10.0.22621.2861]
(c) Microsoft Corporation。保留所有权利。

C:\Users\刘俊杰>D:

D:\>cd d_code\distributed\Project\code\Fileserver

D:\d_code\distributed\Project\code\Fileserver>python replica_server.py
File: example.txt Update time: 1702787658.9935334
Server is listening on localhost:8001
Server is listening on localhost:8000
Server is listening on localhost:8002
127.0.0.1 - - [17/Dec/2023 12:35:07] "POST /RPC2 HTTP/1.1" 200 -

命令提示符 - python client.py

Microsoft Windows [版本 10.0.22621.2861]
(c) Microsoft Corporation。保留所有权利。

C:\Users\刘俊杰>D:

D:\>cd d_code\distributed\Project\code\client\client1

D:\d_code\distributed\Project\code\client\client1>python client.py
Here are the services with ports:
Server: server1, Port: 8001
Server: server2, Port: 8002
Choose the name of server you want to connect: |

可以看到可以选择副本服务器进行连接

命令提示符 - python replica_

Microsoft Windows [版本 10.0.22621.2861]
(c) Microsoft Corporation。保留所有权利。

C:\Users\刘俊杰>D:

D:\>cd d_code\distributed\Project\code\Fileserver

D:\d_code\distributed\Project\code\Fileserver>python replica_server.py
File: example.txt Update time: 1702787658.9935334
Server is listening on localhost:8001
Server is listening on localhost:8000
Server is listening on localhost:8002
127.0.0.1 - - [17/Dec/2023 12:35:07] "POST /RPC2 HTTP/1.1" 200 -

命令提示符 - python client.py

Microsoft Windows [版本 10.0.22621.2861]
(c) Microsoft Corporation。保留所有权利。

C:\Users\刘俊杰>D:

D:\>cd d_code\distributed\Project\code\client\client1

D:\d_code\distributed\Project\code\client\client1>python client.py
Here are the services with ports:
Server: server1, Port: 8001
Server: server2, Port: 8002
Choose the name of server you want to connect: server1
Enter a command (Type 'help' for available commands): |

选择好副本服务器，并进行连接

命令提示符 - python replica_

Microsoft Windows [版本 10.0.22621.2861]
(c) Microsoft Corporation。保留所有权利。

C:\Users\刘俊杰>D:

D:\>cd d_code\distributed\Project\code\Fileserver

D:\d_code\distributed\Project\code\Fileserver>python replica_server.py
File: example.txt Update time: 1702787658.9935334
Server is listening on localhost:8001
Server is listening on localhost:8000
Server is listening on localhost:8002
127.0.0.1 - - [17/Dec/2023 12:35:07] "POST /RPC2 HTTP/1.1" 200 -

命令提示符 - python client.py

Microsoft Windows [版本 10.0.22621.2861]
(c) Microsoft Corporation。保留所有权利。

C:\Users\刘俊杰>D:

D:\>cd d_code\distributed\Project\code\client\client1

D:\d_code\distributed\Project\code\client\client1>python client.py
Here are the services with ports:
Server: server1, Port: 8001
Server: server2, Port: 8002
Choose the name of server you want to connect: server1
Enter a command (Type 'help' for available commands): help
Available Commands:
1. upload_file [filename] - Upload a file to the server.
2. download_file [filename] - Download a file from the server.
3. delete_file [filename] - Delete a file on the server.
4. help() - Display available commands.
5. exit() - Exit the client.
6. list_files(): - List files in the server folder.
7. delete_folder [foldername]: - Delete a folder on the server.
8. read_files [file_name]: - Read a file to the server.
9. port : - Display the port of the client.
10. cache : - Display the cache of the client.
11. write : - write a file from the server.
Enter a command (Type 'help' for available commands): |

输入help指令,查看可使用的指令及其作用

命令提示符 - python replica_

Microsoft Windows [版本 10.0.22621.2861]
(c) Microsoft Corporation。保留所有权利。

C:\Users\刘俊杰>D:

D:\>cd d_code\distributed\Project\code\Fileserver

D:\d_code\distributed\Project\code\Fileserver>python replica_server.py
File: example.txt Update time: 1702787658.9935334
Server is listening on localhost:8001
Server is listening on localhost:8000
Server is listening on localhost:8002
127.0.0.1 - - [17/Dec/2023 12:35:07] "POST /RPC2 HTTP/1.1" 200 -
127.0.0.1 - - [17/Dec/2023 12:36:30] "POST /RPC2 HTTP/1.1" 200 -

命令提示符 - python client.py

Microsoft Windows [版本 10.0.22621.2861]
(c) Microsoft Corporation。保留所有权利。

C:\Users\刘俊杰>D:

D:\>cd d_code\distributed\Project\code\client\client1

D:\d_code\distributed\Project\code\client\client1>python client.py
Here are the services with ports:
Server: server1, Port: 8001
Server: server2, Port: 8002
Choose the name of server you want to connect: server1
Enter a command (Type 'help' for available commands): help
Available Commands:
1. upload_file [filename] - Upload a file to the server.
2. download_file [filename] - Download a file from the server.
3. delete_file [filename] - Delete a file on the server.
4. help() - Display available commands.
5. exit() - Exit the client.
6. list_files(): - List files in the server folder.
7. delete_folder [foldername]: - Delete a folder on the server.
8. read_files [file_name]: - Read a file to the server.
9. port : - Display the port of the client.
10. cache : - Display the cache of the client.
11. write : - write a file from the server.
Enter a command (Type 'help' for available commands): list_files
Files and folders on the server:
- example.txt
Enter a command (Type 'help' for available commands): |

输入list_files指令,查看服务器下的文件

```
命令提示符 - python replica. x + -
Microsoft Windows [版本 10.0.22621.2861]
(c) Microsoft Corporation. 保留所有权利。

C:\Users\刘俊杰>D:

D:\>cd d_code\distributed\Project\code\Fileserver

D:\d_code\distributed\Project\code\Fileserver>python replica_server.py
File: example.txt Update time: 1702787658.9935334
Server is listening on localhost:8001
Server is listening on localhost:8000
Server is listening on localhost:8002
127.0.0.1 - - [17/Dec/2023 12:35:07] "POST /RPC2 HTTP/1.1" 200 -
127.0.0.1 - - [17/Dec/2023 12:36:30] "POST /RPC2 HTTP/1.1" 200 -

命令提示符 - python client.py x + -
Microsoft Windows [版本 10.0.22621.2861]
(c) Microsoft Corporation. 保留所有权利。

C:\Users\刘俊杰>D:

D:\>cd d_code\distributed\Project\code\client\client1

D:\d_code\distributed\Project\code\client\client1>python client.py
Here are the services with ports:
Server: server1, Port: 8001
Server: server2, Port: 8002
Choose the name of server you want to connect: server1
Enter a command (Type 'help' for available commands): help
Available Commands:
1. upload_file [filename] - Upload a file to the server.
2. download_file [filename] - Download a file from the server.
3. delete_file [filename] - Delete a file on the server.
4. help() - Display available commands.
5. exit() - Exit the client.
6. list_files(): - List files in the server folder.
7. delete_folder [foldername]: - Delete a folder on the server.
8. read_files [file_name]: - Read a file to the server.
9. port : - Display the port of the client.
10. cache : - Display the cache of the client.
11. write : - write a file from the server.
Enter a command (Type 'help' for available commands): list_files
Files and folders on the server:
- example.txt
Enter a command (Type 'help' for available commands): cache
Enter a command (Type 'help' for available commands):
```

输入cache，可以看到此时的缓存为空，因为没有读取任何文件到缓存中

```
命令提示符 - python replica. x + -
Microsoft Windows [版本 10.0.22621.2861]
(c) Microsoft Corporation. 保留所有权利。

C:\Users\刘俊杰>D:

D:\>cd d_code\distributed\Project\code\Fileserver

D:\d_code\distributed\Project\code\Fileserver>python replica_server.py
File: example.txt Update time: 1702787658.9935334
Server is listening on localhost:8001
Server is listening on localhost:8000
Server is listening on localhost:8002
127.0.0.1 - - [17/Dec/2023 12:35:07] "POST /RPC2 HTTP/1.1" 200 -
127.0.0.1 - - [17/Dec/2023 12:36:30] "POST /RPC2 HTTP/1.1" 200 -
127.0.0.1 - - [17/Dec/2023 12:37:30] "POST /RPC2 HTTP/1.1" 200 -

命令提示符 - python client.py x + -
Microsoft Windows [版本 10.0.22621.2861]
(c) Microsoft Corporation. 保留所有权利。

C:\Users\刘俊杰>D:

D:\>cd d_code\distributed\Project\code\client\client1

D:\d_code\distributed\Project\code\client\client1>python client.py
Here are the services with ports:
Server: server1, Port: 8001
Server: server2, Port: 8002
Choose the name of server you want to connect: server1
Enter a command (Type 'help' for available commands): help
Available Commands:
1. upload_file [filename] - Upload a file to the server.
2. download_file [filename] - Download a file from the server.
3. delete_file [filename] - Delete a file on the server.
4. help() - Display available commands.
5. exit() - Exit the client.
6. list_files(): - List files in the server folder.
7. delete_folder [foldername]: - Delete a folder on the server.
8. read_files [file_name]: - Read a file to the server.
9. port : - Display the port of the client.
10. cache : - Display the cache of the client.
11. write : - write a file from the server.
Enter a command (Type 'help' for available commands): list_files
Files and folders on the server:
- example.txt
Enter a command (Type 'help' for available commands): cache
Enter a command (Type 'help' for available commands): read_file example.txt
Content of 'example.txt':
hello world
Enter a command (Type 'help' for available commands):
```

读取example.txt文件，可以看到打印出了文件内容

```
命令提示符 - python replica. x + -
Microsoft Windows [版本 10.0.22621.2861]
(c) Microsoft Corporation. 保留所有权利。

C:\Users\刘俊杰>D:

D:\>cd d_code\distributed\Project\code\Fileserver

D:\d_code\distributed\Project\code\Fileserver>python replica_server.py
File: example.txt Update time: 1702787658.9935334
Server is listening on localhost:8001
Server is listening on localhost:8000
Server is listening on localhost:8002
127.0.0.1 - - [17/Dec/2023 12:35:07] "POST /RPC2 HTTP/1.1" 200 -
127.0.0.1 - - [17/Dec/2023 12:36:30] "POST /RPC2 HTTP/1.1" 200 -
127.0.0.1 - - [17/Dec/2023 12:37:30] "POST /RPC2 HTTP/1.1" 200 -

命令提示符 - python client.py x + -
Microsoft Windows [版本 10.0.22621.2861]
(c) Microsoft Corporation. 保留所有权利。

C:\Users\刘俊杰>D:

D:\>cd d_code\distributed\Project\code\client\client1

D:\d_code\distributed\Project\code\client\client1>python client.py
Here are the services with ports:
Server: server1, Port: 8001
Server: server2, Port: 8002
Choose the name of server you want to connect: server1
Enter a command (Type 'help' for available commands): help
Available Commands:
1. upload_file [filename] - Upload a file to the server.
2. download_file [filename] - Download a file from the server.
3. delete_file [filename] - Delete a file on the server.
4. help() - Display available commands.
5. exit() - Exit the client.
6. list_files(): - List files in the server folder.
7. delete_folder [foldername]: - Delete a folder on the server.
8. read_files [file_name]: - Read a file to the server.
9. port : - Display the port of the client.
10. cache : - Display the cache of the client.
11. write : - write a file from the server.
Enter a command (Type 'help' for available commands): list_files
Files and folders on the server:
- example.txt
Enter a command (Type 'help' for available commands): cache
Enter a command (Type 'help' for available commands): read_file example.txt
Content of 'example.txt':
hello world
Enter a command (Type 'help' for available commands): cache
file: example.txt update_time: 1702787658.9935334 read_time: 1702787658.9935334
Enter a command (Type 'help' for available commands):
```

此时再查看cache，可以看到缓存中已有了example.txt

```
命令提示符 - python replica.py x + -
Microsoft Windows [版本 10.0.22621.2861]
(c) Microsoft Corporation. 保留所有权利。

C:\Users\刘俊杰>D:

D:\>cd d_code\distributed\Project\code\Fileserver

D:\d_code\distributed\Project\code\Fileserver>python replica_server.py
File: example.txt Update time: 1702787658.9935334
Server is listening on localhost:8001
Server is listening on localhost:8000
Server is listening on localhost:8002
127.0.0.1 - - [17/Dec/2023 12:35:07] "POST /RPC2 HTTP/1.1" 200 -
127.0.0.1 - - [17/Dec/2023 12:36:30] "POST /RPC2 HTTP/1.1" 200 -
127.0.0.1 - - [17/Dec/2023 12:37:30] "POST /RPC2 HTTP/1.1" 200 -
127.0.0.1 - - [17/Dec/2023 12:38:16] "POST /RPC2 HTTP/1.1" 200 -

命令提示符 - python client.py x + -
Microsoft Windows [版本 10.0.22621.2861]
(c) Microsoft Corporation. 保留所有权利。

C:\Users\刘俊杰>D:

D:\>cd d_code\distributed\Project\code\client\client1

D:\d_code\distributed\Project\code\client\client1>python client.py
Here are the services with ports:
Server: server1, Port: 8001
Server: server2, Port: 8002
Choose the name of server you want to connect: server1
Enter a command (Type 'help' for available commands): help
Available Commands:
1. upload_file [filename] - Upload a file to the server.
2. download_file [filename] - Download a file from the server.
3. delete_file [filename] - Delete a file on the server.
4. help() - Display available commands.
5. exit() - Exit the client.
6. list_files(): - List files in the server folder.
7. delete_folder [foldername]: - Delete a folder on the server.
8. read_files [file_name]: - Read a file to the server.
9. port : - Display the port of the client.
10. cache : - Display the cache of the client.
11. write : - write a file from the server.
Enter a command (Type 'help' for available commands): list_files
Files and folders on the server:
- example.txt
Enter a command (Type 'help' for available commands): cache
Enter a command (Type 'help' for available commands): read_file example.txt
Content of 'example.txt':
hello world
Enter a command (Type 'help' for available commands): cache
file: example.txt update_time: 1702787658.9935334 read_time: 1702787658.9935
334
Enter a command (Type 'help' for available commands): read_file example.txt
Reading from cache
Content of 'example.txt':
hello world
Enter a command (Type 'help' for available commands):
```

再读取example.txt文件,可以看到此时是从缓存中读的

```
命令提示符 - python replica.py x + -
Microsoft Windows [版本 10.0.22621.2861]
(c) Microsoft Corporation. 保留所有权利。

C:\Users\刘俊杰>D:

D:\>cd d_code\distributed\Project\code\Fileserver

D:\d_code\distributed\Project\code\Fileserver>python replica_server.py
File: example.txt Update time: 1702787658.9935334
Server is listening on localhost:8001
Server is listening on localhost:8000
Server is listening on localhost:8002
127.0.0.1 - - [17/Dec/2023 12:35:07] "POST /RPC2 HTTP/1.1" 200 -
127.0.0.1 - - [17/Dec/2023 12:36:30] "POST /RPC2 HTTP/1.1" 200 -
127.0.0.1 - - [17/Dec/2023 12:37:30] "POST /RPC2 HTTP/1.1" 200 -
127.0.0.1 - - [17/Dec/2023 12:38:16] "POST /RPC2 HTTP/1.1" 200 -

命令提示符 - python client.py x + -
(c) Microsoft Corporation. 保留所有权利。

C:\Users\刘俊杰>D:

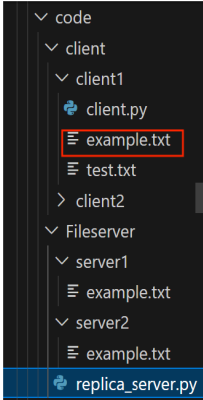
D:\>cd d_code\distributed\Project\code\client\client1

D:\d_code\distributed\Project\code\client\client1>python client.py
Here are the services with ports:
Server: server1, Port: 8001
Server: server2, Port: 8002
Choose the name of server you want to connect: server1
Enter a command (Type 'help' for available commands): help
Available Commands:
1. upload_file [filename] - Upload a file to the server.
2. download_file [filename] - Download a file from the server.
3. delete_file [filename] - Delete a file on the server.
4. help() - Display available commands.
5. exit() - Exit the client.
6. list_files(): - List files in the server folder.
7. delete_folder [foldername]: - Delete a folder on the server.
8. read_files [file_name]: - Read a file to the server.
9. port : - Display the port of the client.
10. cache : - Display the cache of the client.
11. write : - write a file from the server.
Enter a command (Type 'help' for available commands): list_files
Files and folders on the server:
- example.txt
Enter a command (Type 'help' for available commands): cache
Enter a command (Type 'help' for available commands): read_file example.txt
Content of 'example.txt':
hello world
Enter a command (Type 'help' for available commands): cache
file: example.txt update_time: 1702787658.9935334 read_time: 1702787658.9935
334
Enter a command (Type 'help' for available commands): read_file example.txt
Reading from cache
Content of 'example.txt':
hello world
Enter a command (Type 'help' for available commands): cache
file: example.txt update_time: 1702787658.9935334 read_time: 1702787894.7894
936
Enter a command (Type 'help' for available commands):
```

此时再查看cache，可以看到缓存中example.txt上次读时间被更新

命令提示符 - python replica_...
Microsoft Windows [版本 10.0.22621.2861]
(c) Microsoft Corporation. 保留所有权利。
C:\Users\刘俊杰>D:
D:\>cd d_code\distributed\Project\code\Fileserver
D:\d_code\distributed\Project\code\Fileserver>python replica_server.py
File: example.txt Update time: 1702787658.9935334
Server is listening on localhost:8001
Server is listening on localhost:8000
Server is listening on localhost:8002
127.0.0.1 - - [17/Dec/2023 12:35:07] "POST /RPC2 HTTP/1.1" 200 -
127.0.0.1 - - [17/Dec/2023 12:36:30] "POST /RPC2 HTTP/1.1" 200 -
127.0.0.1 - - [17/Dec/2023 12:37:30] "POST /RPC2 HTTP/1.1" 200 -
127.0.0.1 - - [17/Dec/2023 12:38:16] "POST /RPC2 HTTP/1.1" 200 -
127.0.0.1 - - [17/Dec/2023 12:39:55] "POST /RPC2 HTTP/1.1" 200 -

命令提示符 - python client.py
D:\d_code\distributed\Project\code\client\client1>python client.py
Here are the services with ports:
Server: server1, Port: 8001
Server: server2, Port: 8002
Choose the name of server you want to connect: server1
Enter a command (Type 'help' for available commands): help
Available Commands:
1. upload_file [filename] - Upload a file to the server.
2. download_file [filename] - Download a file from the server.
3. delete_file [filename] - Delete a file on the server.
4. help() - Display available commands.
5. exit() - Exit the client.
6. list_files(): - List files in the server folder.
7. delete_folder [foldername]: - Delete a folder on the server.
8. read_files [file_name]: - Read a file to the server.
9. port : - Display the port of the client.
10. cache : - Display the cache of the client.
11. write : - write a file from the server.
Enter a command (Type 'help' for available commands): list_files
Files and folders on the server:
- example.txt
Enter a command (Type 'help' for available commands): cache
Enter a command (Type 'help' for available commands): read_file example.txt
Content of 'example.txt':
hello world
Enter a command (Type 'help' for available commands): cache
file: example.txt update_time: 1702787658.9935334 read_time: 1702787658.9935334
Enter a command (Type 'help' for available commands): read_file example.txt
Reading from cache
Content of 'example.txt':
hello world
Enter a command (Type 'help' for available commands): cache
file: example.txt update_time: 1702787658.9935334 read_time: 1702787894.7894936
Enter a command (Type 'help' for available commands): download example.txt
Invalid command. Type 'help' for available commands.
Enter a command (Type 'help' for available commands): download_file example.txt
{'success': True, 'content': 'hello world'}
Enter a command (Type 'help' for available commands):



下载example.txt文件

可以看到client1文件夹下成功下载了example.txt

```
命令提示符 - python replica.py x + - x
Microsoft Windows [版本 10.0.22621.2861]
(c) Microsoft Corporation. 保留所有权利。

C:\Users\刘俊杰>D:
D:\>cd d_code\distributed\Project\code\Fileserver
D:\d_code\distributed\Project\code\Fileserver>python replica_server.py
File: example.txt Update time: 1702787658.9935334
Server is listening on localhost:8001
Server is listening on localhost:8000
Server is listening on localhost:8002
127.0.0.1 - - [17/Dec/2023 12:35:07] "POST /RPC2 HTTP/1.1" 200 -
127.0.0.1 - - [17/Dec/2023 12:36:30] "POST /RPC2 HTTP/1.1" 200 -
127.0.0.1 - - [17/Dec/2023 12:37:30] "POST /RPC2 HTTP/1.1" 200 -
127.0.0.1 - - [17/Dec/2023 12:38:16] "POST /RPC2 HTTP/1.1" 200 -
127.0.0.1 - - [17/Dec/2023 12:39:55] "POST /RPC2 HTTP/1.1" 200 -

命令提示符 - python client.py x + - x
Server: server1, Port: 8001
Server: server2, Port: 8002
Choose the name of server you want to connect: server1
Enter a command (Type 'help' for available commands): help
Available Commands:
1. upload_file [filename] - Upload a file to the server.
2. download_file [filename] - Download a file from the server.
3. delete_file [filename] - Delete a file on the server.
4. help() - Display available commands.
5. exit() - Exit the client.
6. list_files(): - List files in the server folder.
7. delete_folder [foldername]: - Delete a folder on the server.
8. read_files [file_name]: - Read a file to the server.
9. port : - Display the port of the client.
10. cache : - Display the cache of the client.
11. write : - write a file from the server.
Enter a command (Type 'help' for available commands): list_files
Files and folders on the server:
- example.txt
Enter a command (Type 'help' for available commands): cache
Enter a command (Type 'help' for available commands): read_file example.txt
Content of 'example.txt':
hello world
Enter a command (Type 'help' for available commands): cache
file: example.txt update_time: 1702787658.9935334 read_time: 1702787658.9935334
Enter a command (Type 'help' for available commands): read_file example.txt
Reading from cache
Content of 'example.txt':
hello world
Enter a command (Type 'help' for available commands): cache
file: example.txt update_time: 1702787658.9935334 read_time: 1702787894.7894936
Enter a command (Type 'help' for available commands): download example.txt
Invalid command. Type 'help' for available commands.
Enter a command (Type 'help' for available commands): download_file example.txt
{'success': True, 'content': 'hello world'}
Enter a command (Type 'help' for available commands): cache
file: example.txt update_time: 1702787658.9935334 read_time: 1702787658.9935334
Enter a command (Type 'help' for available commands):
```

此时再查看cache，可以看到缓存中example.txt上次读时间被更新

命令提示符 - python replica.py

Microsoft Windows [版本 10.0.22621.2861]
(c) Microsoft Corporation. 保留所有权利。

C:\Users\刘俊杰>D:

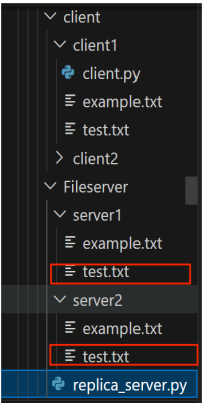
D:\>cd d_code\distributed\Project\code\Fileserver

D:\d_code\distributed\Project\code\Fileserver>python replica_server.py
File: example.txt Update time: 1702787658.9935334
Server is listening on localhost:8001
Server is listening on localhost:8000
Server is listening on localhost:8002
127.0.0.1 - - [17/Dec/2023 12:35:07] "POST /RPC2 HTTP/1.1" 200 -
127.0.0.1 - - [17/Dec/2023 12:36:30] "POST /RPC2 HTTP/1.1" 200 -
127.0.0.1 - - [17/Dec/2023 12:37:30] "POST /RPC2 HTTP/1.1" 200 -
127.0.0.1 - - [17/Dec/2023 12:38:16] "POST /RPC2 HTTP/1.1" 200 -
127.0.0.1 - - [17/Dec/2023 12:39:55] "POST /RPC2 HTTP/1.1" 200 -
127.0.0.1 - - [17/Dec/2023 12:41:10] "POST /RPC2 HTTP/1.1" 200 -
|

命令提示符 - python client.py

Available Commands:
1. upload_file [filename] - Upload a file to the server.
2. download_file [filename] - Download a file from the server.
3. delete_file [filename] - Delete a file on the server.
4. help() - Display available commands.
5. exit() - Exit the client.
6. list_files(): - List files in the server folder.
7. delete_folder [foldername]: - Delete a folder on the server.
8. read_files [file_name]: - Read a file to the server.
9. port : - Display the port of the client.
10. cache : - Display the cache of the client.
11. write : - write a file from the server.
Enter a command (Type 'help' for available commands): list_files
Files and folders on the server:
- example.txt
Enter a command (Type 'help' for available commands): cache
Enter a command (Type 'help' for available commands): read_file example.txt
Content of 'example.txt':
hello world
Enter a command (Type 'help' for available commands): cache
file: example.txt update_time: 1702787658.9935334 read_time: 1702787658.9935334
Enter a command (Type 'help' for available commands): read_file example.txt
Reading from cache
Content of 'example.txt':
hello world
Enter a command (Type 'help' for available commands): cache
file: example.txt update_time: 1702787658.9935334 read_time: 1702787894.7894936
Enter a command (Type 'help' for available commands): download example.txt
Invalid command. Type 'help' for available commands.
Enter a command (Type 'help' for available commands): download_file example.txt
{'success': True, 'content': 'hello world'}
Enter a command (Type 'help' for available commands): cache
file: example.txt update_time: 1702787658.9935334 read_time: 1702787658.9935334
Enter a command (Type 'help' for available commands): upload test.txt
Invalid command. Type 'help' for available commands.
Enter a command (Type 'help' for available commands): upload_file test.txt
True
Enter a command (Type 'help' for available commands): |

上传test.txt文件



可以看到不止server1上上传了test.txt，同时server2上也有test.txt，这是实现了副本一致性的效果

命令提示符 - python replica_

Microsoft Windows [版本 10.0.22621.2861]
(c) Microsoft Corporation. 保留所有权利。

C:\Users\刘俊杰>D:

D:\>cd d_code\distributed\Project\code\Fileserver

D:\d_code\distributed\Project\code\Fileserver>python replica_server.py
File: example.txt Update time: 1702787658.9935334
Server is listening on localhost:8001
Server is listening on localhost:8000
Server is listening on localhost:8002
127.0.0.1 - - [17/Dec/2023 12:35:07] "POST /RPC2 HTTP/1.1" 200 -
127.0.0.1 - - [17/Dec/2023 12:36:30] "POST /RPC2 HTTP/1.1" 200 -
127.0.0.1 - - [17/Dec/2023 12:37:30] "POST /RPC2 HTTP/1.1" 200 -
127.0.0.1 - - [17/Dec/2023 12:38:16] "POST /RPC2 HTTP/1.1" 200 -
127.0.0.1 - - [17/Dec/2023 12:39:55] "POST /RPC2 HTTP/1.1" 200 -
127.0.0.1 - - [17/Dec/2023 12:41:10] "POST /RPC2 HTTP/1.1" 200 -
D:\d_code\distributed\Project\code\Fileserver
server1 test.txt
D:\d_code\distributed\Project\code\Fileserver\server2\test.txt
127.0.0.1 - - [17/Dec/2023 12:41:47] "POST /RPC2 HTTP/1.1" 200 -
|

命令提示符 - python client.py

2. download_file [filename] - Download a file from the server.
3. delete_file [filename] - Delete a file on the server.
4. help() - Display available commands.
5. exit() - Exit the client.
6. list_files(): - List files in the server folder.
7. delete_folder [foldername]: - Delete a folder on the server.
8. read_files [file_name]: - Read a file to the server.
9. port : - Display the port of the client.
10. cache : - Display the cache of the client.
11. write : - write a file from the server.
Enter a command (Type 'help' for available commands): list_files
Files and folders on the server:
- example.txt
Enter a command (Type 'help' for available commands): cache
Enter a command (Type 'help' for available commands): read_file example.txt
Content of 'example.txt':
hello world
Enter a command (Type 'help' for available commands): cache
file: example.txt update_time: 1702787658.9935334 read_time: 1702787658.9935334
Enter a command (Type 'help' for available commands): read_file example.txt
Reading from cache
Content of 'example.txt':
hello world
Enter a command (Type 'help' for available commands): cache
file: example.txt update_time: 1702787658.9935334 read_time: 1702787894.7894936
Enter a command (Type 'help' for available commands): download example.txt
Invalid command. Type 'help' for available commands.
Enter a command (Type 'help' for available commands): download_file example.txt
{'success': True, 'content': 'hello world'}
Enter a command (Type 'help' for available commands): cache
file: example.txt update_time: 1702787658.9935334 read_time: 1702787658.9935334
Enter a command (Type 'help' for available commands): upload test.txt
Invalid command. Type 'help' for available commands.
Enter a command (Type 'help' for available commands): upload_file test.txt
True
Enter a command (Type 'help' for available commands): delete_file test.txt
{'success': True, 'message': "File 'test.txt' deleted from server."}
Enter a command (Type 'help' for available commands):

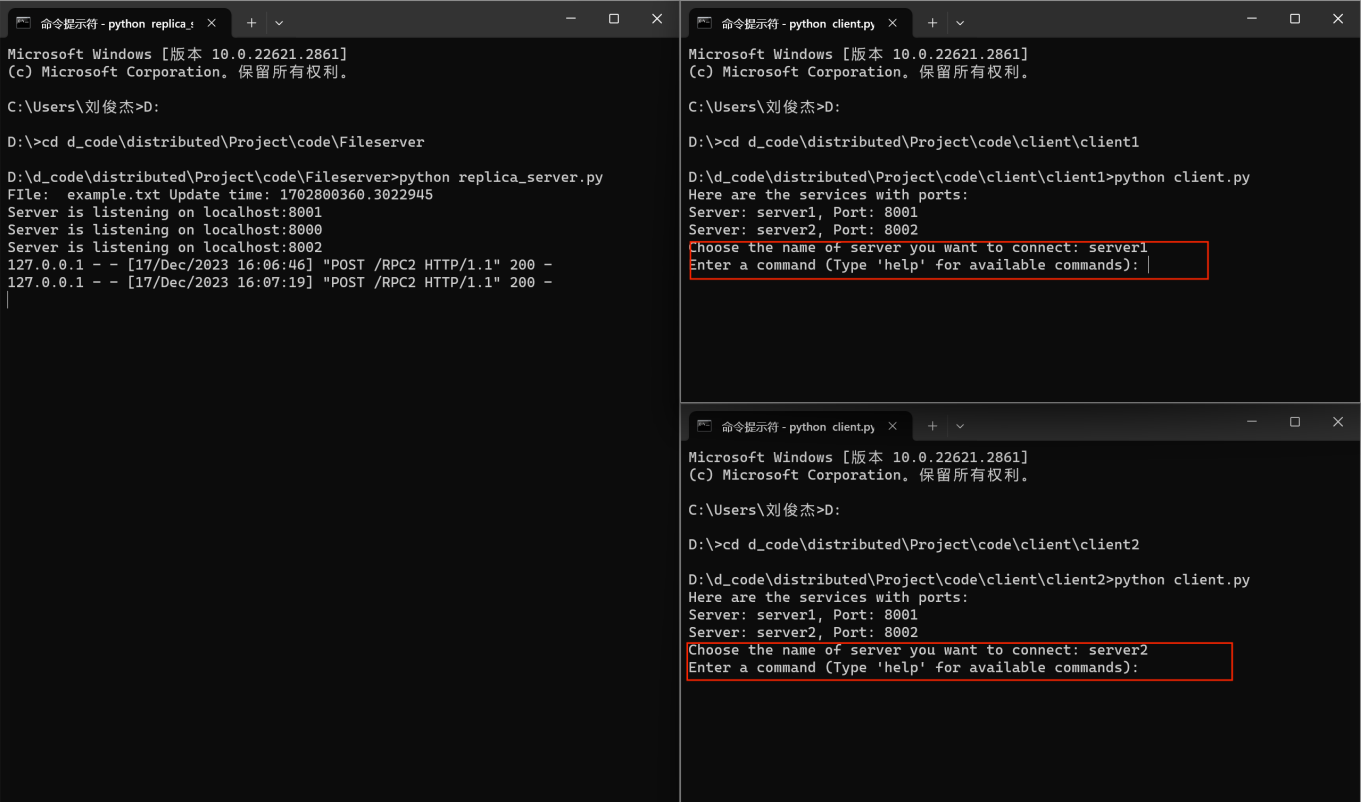
删除test.txt文件

可以看到删除文件操作也实现了副本一致性，server1和server2下的test.txt都被删除

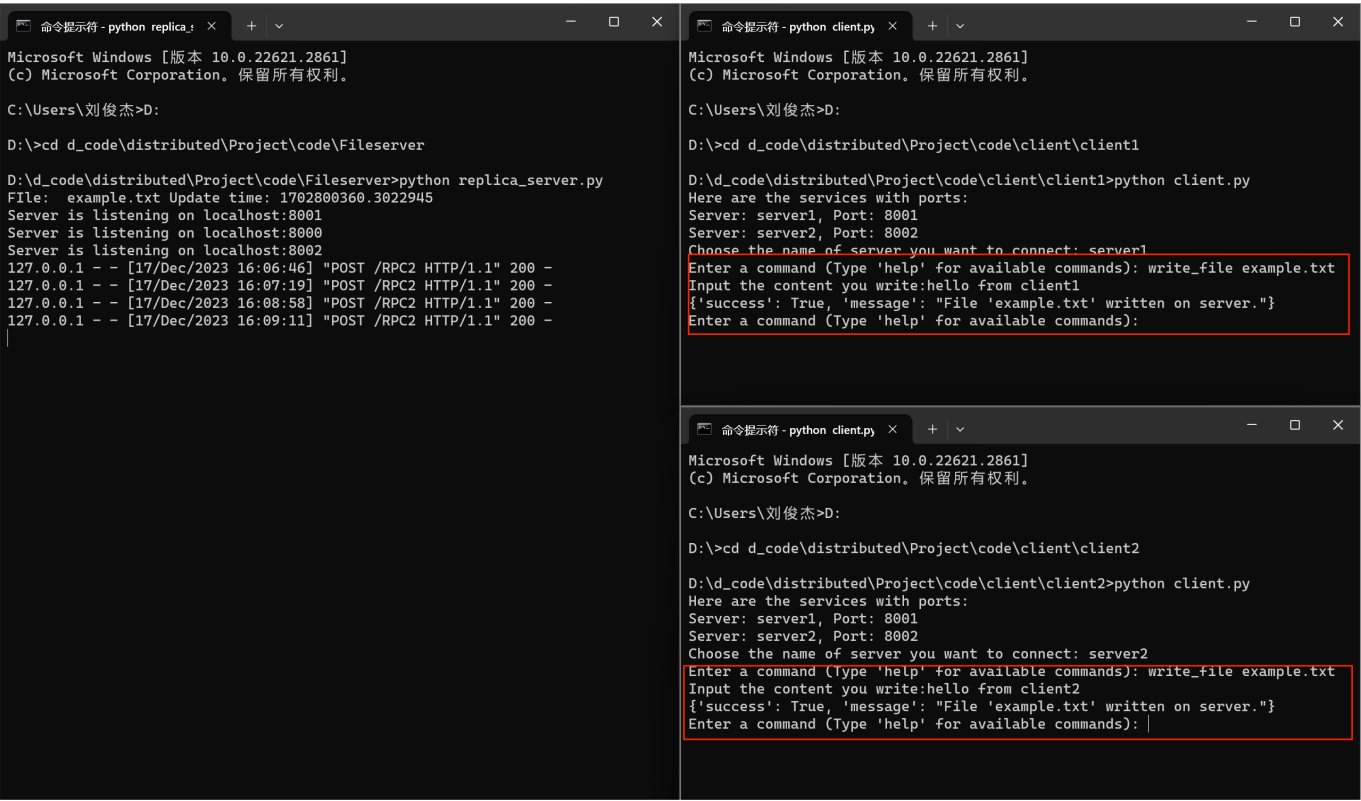
(2)运行两个客户端测试并行写(这里的测试不够完善,在后后面的问题中会说明)

运行两个客户端,client1连接server1,client2连接server2

16 / 20



同时运行两个客户端对example.txt执行写操作



观察两个副本服务器下的example.txt

```
distributed > Project > code > Fileserver > server1 > ≡ example.txt
```

```
1 hello from client2
```

```
distributed > Project > code > Fileserver > server2 > ≡ example.txt
```

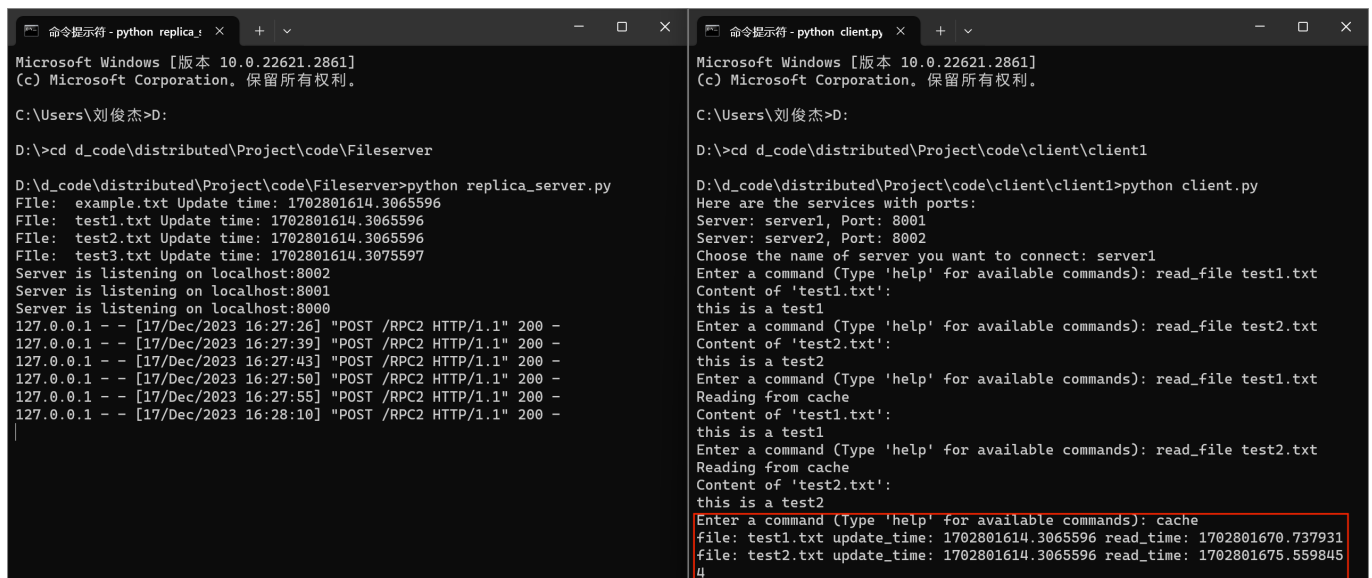
```
1 hello from client2
```

可以看到写入的结果是client2输入的结果,没有发生写冲突

(3)运行一个客户端测试LRU缓存更新

首先将缓存区大小设为2个文件

按先读test1.txt后读test2.txt的顺序进行两次



```
命令提示符 - python replica.py x + -
Microsoft Windows [版本 10.0.22621.2861]
(c) Microsoft Corporation. 保留所有权利。

C:\Users\刘俊杰>D:
D:\>cd d_code\distributed\Project\code\Fileserver
D:\d_code\distributed\Project\code\Fileserver>python replica_server.py
File: example.txt Update time: 1702801614.3065596
File: test1.txt Update time: 1702801614.3065596
File: test2.txt Update time: 1702801614.3065596
File: test3.txt Update time: 1702801614.3075597
Server is listening on localhost:8002
Server is listening on localhost:8001
Server is listening on localhost:8000
127.0.0.1 - - [17/Dec/2023 16:27:26] "POST /RPC2 HTTP/1.1" 200 -
127.0.0.1 - - [17/Dec/2023 16:27:39] "POST /RPC2 HTTP/1.1" 200 -
127.0.0.1 - - [17/Dec/2023 16:27:43] "POST /RPC2 HTTP/1.1" 200 -
127.0.0.1 - - [17/Dec/2023 16:27:50] "POST /RPC2 HTTP/1.1" 200 -
127.0.0.1 - - [17/Dec/2023 16:27:55] "POST /RPC2 HTTP/1.1" 200 -
127.0.0.1 - - [17/Dec/2023 16:28:10] "POST /RPC2 HTTP/1.1" 200 -

命令提示符 - python client.py x + -
Microsoft Windows [版本 10.0.22621.2861]
(c) Microsoft Corporation. 保留所有权利。

C:\Users\刘俊杰>D:
D:\>cd d_code\distributed\Project\code\client\client1
D:\d_code\distributed\Project\code\client\client1>python client.py
Here are the services with ports:
Server: server1, Port: 8001
Server: server2, Port: 8002
Choose the name of server you want to connect: server1
Enter a command (Type 'help' for available commands): read_file test1.txt
Content of 'test1.txt':
this is a test1
Enter a command (Type 'help' for available commands): read_file test2.txt
Content of 'test2.txt':
this is a test2
Enter a command (Type 'help' for available commands): read_file test1.txt
Reading from cache
Content of 'test1.txt':
this is a test1
Enter a command (Type 'help' for available commands): read_file test2.txt
Reading from cache
Content of 'test2.txt':
this is a test2
Enter a command (Type 'help' for available commands): cache
file: test1.txt update_time: 1702801614.3065596 read_time: 1702801670.737931
file: test2.txt update_time: 1702801614.3065596 read_time: 1702801675.559845
4
```

可以看到缓存中test2.txt上次读的时间比test1.txt晚

读test3.txt, 观察缓存更新

可以看到上次读时间最早的test1.txt被换出缓冲区,test3.txt被换入缓存区

(4)权限测试

以上测试都是赋予用户所有权限 现在只赋予读的权限:

```
D:\d_code\distributed\Project\code2\client\client1>python client.py
Here are the services with ports:
Server: server1, Port: 8001
Server: server2, Port: 8002
Choose the name of server you want to connect: server1
Enter a command (Type 'help' for available commands): list_files
Files and folders on the server:
- example.txt
- test1.txt
- test2.txt
- test3.txt
Enter a command (Type 'help' for available commands): read example.txt
Invalid command. Type 'help' for available commands.
Enter a command (Type 'help' for available commands): read_file example.txt
Content of 'example.txt':
hello from client2
Enter a command (Type 'help' for available commands): download_file test1.tx
t
You don't have the privilege!
Enter a command (Type 'help' for available commands):
```

可以看到用户成功读文件了,但是没有获得下载文件的权限 下载文件失败

5.遇到的问题

(1)没有很好地测试并行写

问题所在

因为这里实现的写文件操作是客户端输入内容,然后远程调用服务端的写函数,将内容传给服务端,在服务端处来实现写。而不是像我们平时使用的文档编辑一样,能打开文件之后再进行编辑最后保存。这样就会造成写文件从输入内容结束到写文件完成都是极短时间内完成的,无法很好地检测并行写是否造成了写冲突。

解决方法

还未解决

(2)缓存更新的时间问题

问题所在

采用的缓存更新算法LRU,我用updatetime来记录修改时间(用来判断是否为最新版本),用readtime来记录上次读的时间(来实现LRU更新),虽然两者都是用time()来赋值,但是updatetime是服务器time()的时间,readtime是客户端time()的时间,两者的时间记录不一致。

解决方法

将两者独立开来,updatetime只与服务器上的修改时间比较,赋值通过服务端的返回时间赋值,readtime只与客户端上的时间比较,赋值在客户端用time()赋值。在客户端修改文件时,时间更改也只在服务端修改,缓存中的修改时间不改,到需要读文件时,将该文件时间与服务端文件修改时间比较,就能知道此文件不是最新的。

(3)解决文件夹中又有文件夹的问题

问题所在

文件夹中又有文件夹，文件存放方式复杂，就会造成管理的文件中有文件重名问题。

解决方法

首先利用了递归的方法来搜索所用文件,这样能保证每个文件都能被搜索到，同时用相对地址来作为每个文件的键，避免文件重名问题。

```
def list_files(self, folder=None): # 列出文件夹和文件
    try:
        if folder is None:
            folder_path = self.server_files_directory
        else:
            folder_path = os.path.join(self.server_files_directory, folder)

        files = os.listdir(folder_path)
        file_dict = {}
        # print(files)
        for file in files:
            file_path_tmp = os.path.join(folder_path, file)
            if os.path.isfile(file_path_tmp):
                file_dict[file] = file
            elif os.path.isdir(file_path_tmp):
                subfolder_files = self.list_files(os.path.join(folder_path,
file))

                file_dict[file] = subfolder_files

        return file_dict
    except FileNotFoundError:
        return False
```

6.总结

通过这次的课程设计，我深入了解了分布式文件系统的原理和技术，包括节点通信、一致性、缓存机制等，使我对分布式系统的工作方式有了更清晰的认识。

我学会使用RPC进行节点间通信是一项关键的技能。通过实现基于XML-RPC的通信，我更好地理解远程过程调用的原理，并学到了如何在分布式系统中进行有效的通信；学会管理多个副本、实现一致性操作是设计分布式系统的关键。这让我认识到在分布式环境下，数据的一致性是需要特别关注和处理的问题；实现文件锁机制让我了解了在多用户并行读写的情境下，如何通过文件锁来保证数据的完整性和一致性，对分布式文件系统的正确运作至关重要；学会使用缓存机制并实现LRU算法，我对缓存的优化和更新有了更深刻的理解，这有助于提高系统的性能和效率。

在设计分布式文件系统的过程中，不仅学到了理论知识，还学会了如何将理论知识应用到实际问题中，提升了解决问题的能力。在系统设计的过程中，我意识到构建分布式系统是不不断调整和优化是一个渐进的过程。通过发现问题、修改设计并实践，我更好地理解分布式文件系统和持续改进的重要性。