大三暑假大作业中期检查报告

袁晓丹

项目名称：多核操作系统JOS实现

指导教师:陈海波、王彭

项目团队成员	洪扬、郑浩南、袁晓丹

项目进展情况：(以下仅列举个人)

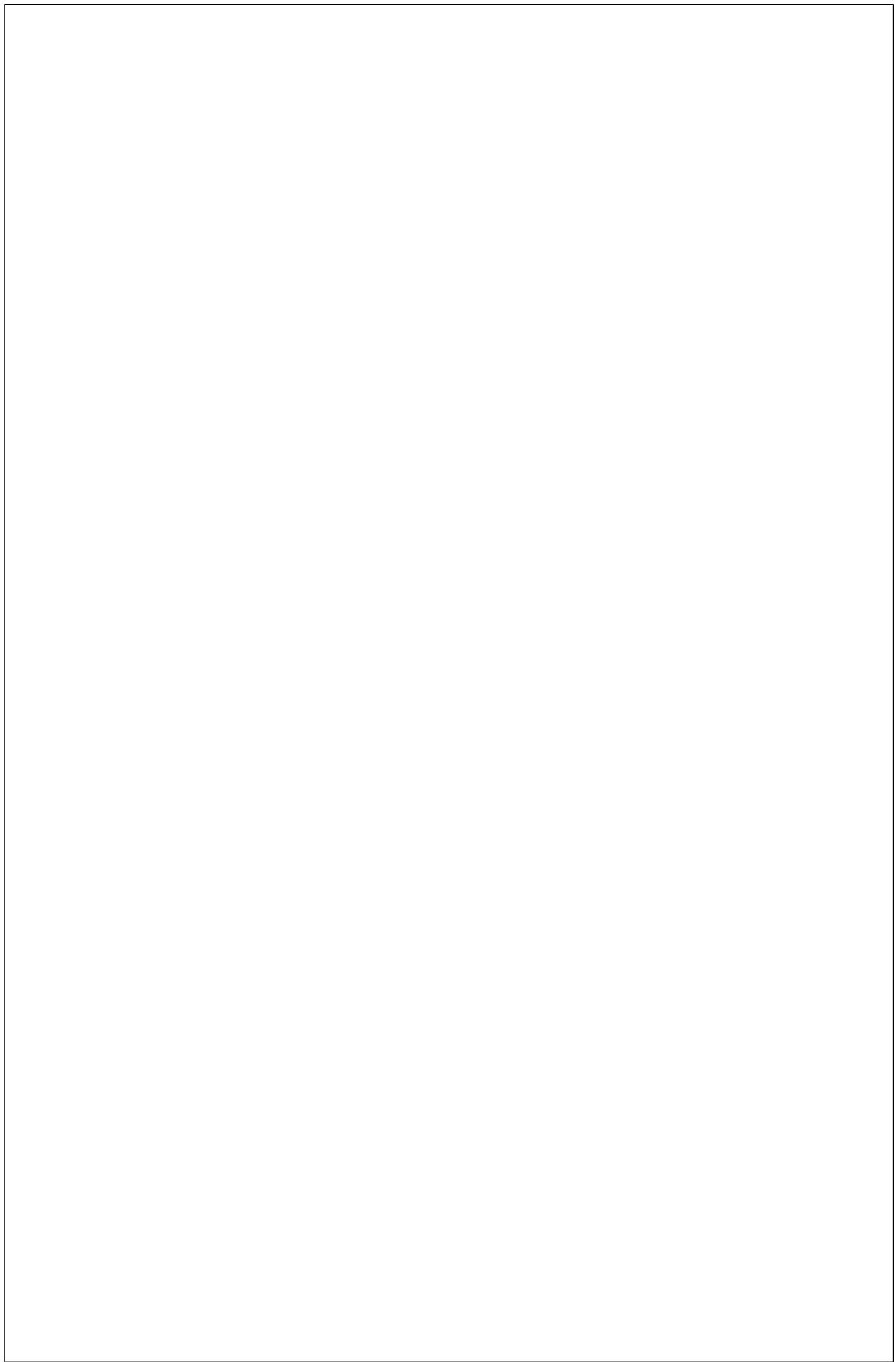
2012.6.25 – 2012.6.29: Finished exercises before round-robin in lab 4 with Zhen, and get a better revision to proceed

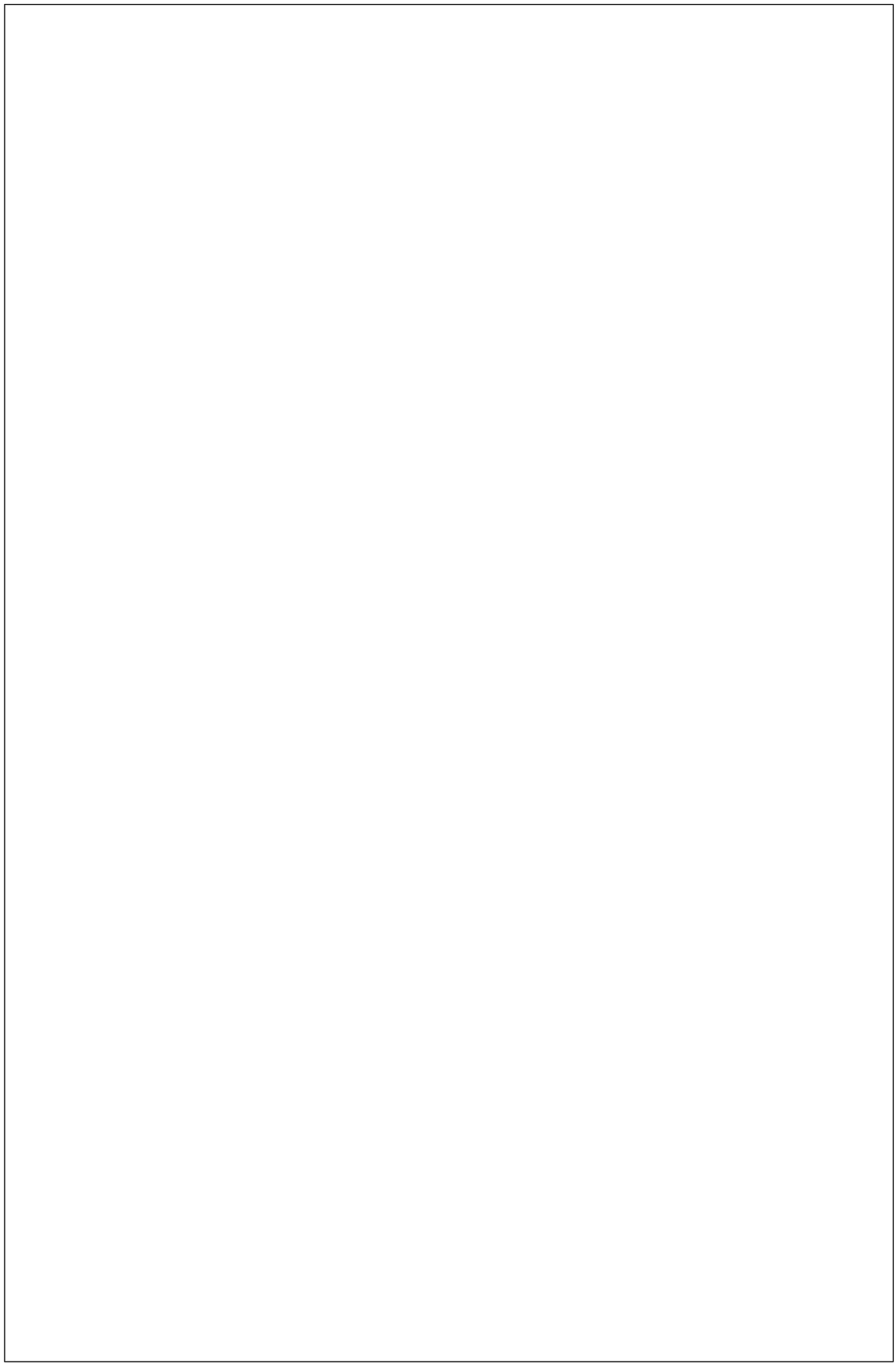
2012.7.02 – 2012.7.06: Finished all part A in lab4 with Zhen

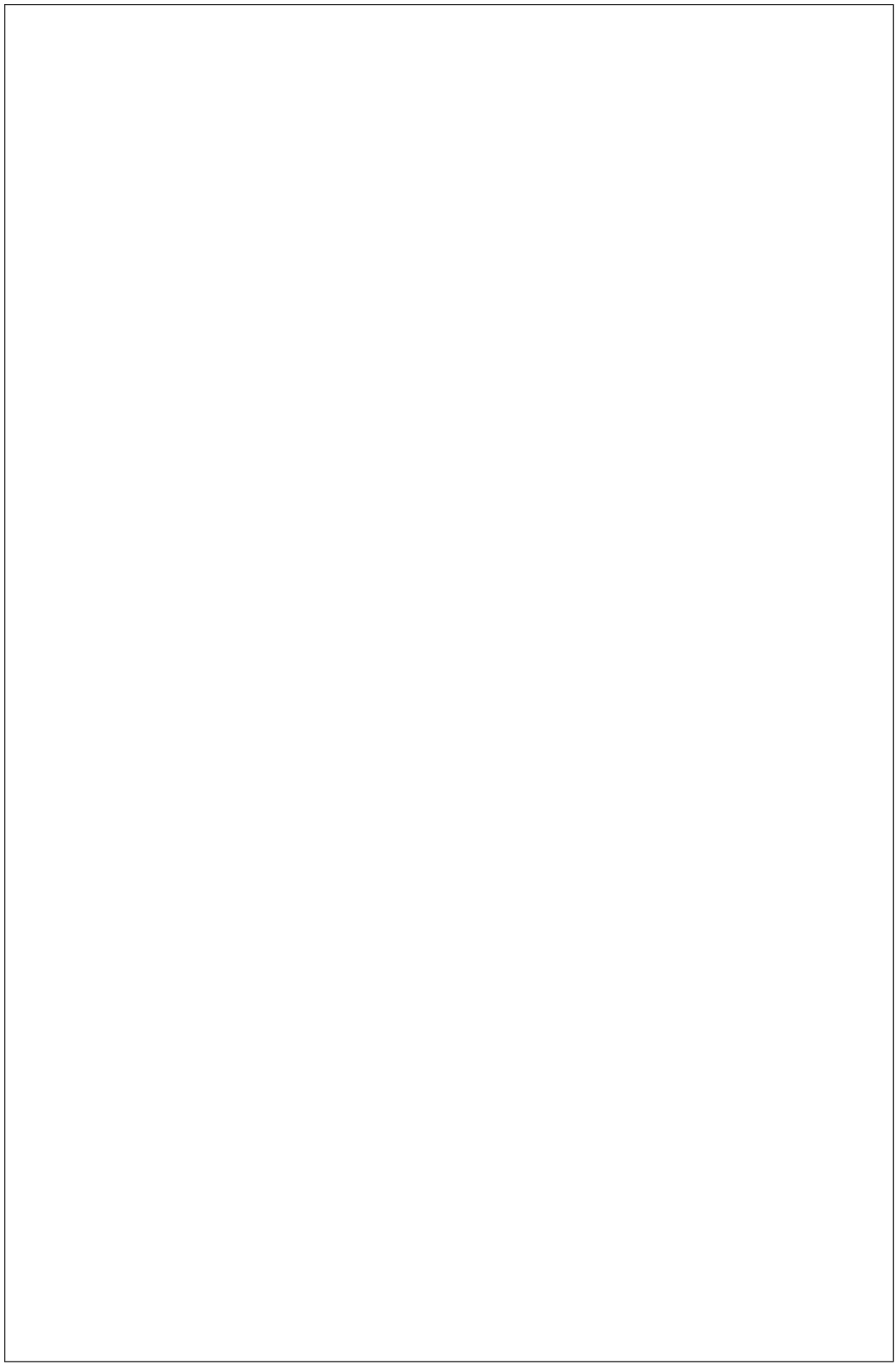
2012.7.09 – 2012.7.13: Parallel programming begined: reviewed fs's basics,
merged files between lab4/lab5,
and finished ex1—disk access in lab 5

2012.7.16 – 2012.7.20: Finished ex2~4—block cache, bitmap, operations in lab 5,
Went deeply into *RPC*,
fixed a IPC bug in lab 4 and finished ex5~8
—Client/Server File System Access and Operations in lab 5

2012.7.23 – 2012.7.27: Finished last ex—spawning in lab 5, parallel programming ended,
I tried to do ex7 in lab 6, and made a plan of my future work
—Make the file system fault tolerant: using journaling







已取得的阶段性成果：

完成三个实验，分别是多核、文件系统、网络驱动部分代码

存在的问题及解决思路：

主要集中在 lab 5 里

1. 还需要做其他事情来保证"当进程切换时，I/O 权限设置的保存与恢复正常"吗?

答：这里我们需要了解下环境状态是怎么设置的。寄存器的恢复是在 env pop tf ()中完成的,里面好像没有恢复 eflags,但是 env pop tf ()的过程里 popal 指令从栈中恢复了所有的通用寄存器,然后是在 iret 指令中恢复了 eip, cs 以及 eflags 寄存器。

2. 因为一般机器硬盘显然不止 3GB,但是一个 32 位机器虚拟地址只有 4GB 的地址空间,所以这里 JOS 的做法是什么?

答：因为 JOS 支持的磁盘大小最大在 3GB 左右,所以我们可以使用类似 lab4 中实现 fork 的 COW 页面机制,也就是

1）用文件系统服务进程的虚拟地址空间(4GB)对应到磁盘的地址空间上(3GB)
2）初始文件系统服务进程里什么页面都没映射,如果要访问一个磁盘的地址空间,则发生页错误

3）在页错误处理程序中,在内存中申请一个块的空间映射到相应的文件系统虚拟地址上,然后去实际的物理磁盘上读取这个区域的东西到这个内存区域上,然后恢复文件系统服务进程。
这样就使用用户进程的机制完成了对于物理磁盘的读写机制,并且尽量少节省了内存。

3. 我们这里只对 Super 指针的地址进行了赋值,那实际内存区域里存的东西是什么时候被初始化的呢?

答：diskaddr 定义在 fs/bc.c 中
。可见 Super 的位置是虚拟地址空间的第一块,当这个块被访问的时候,自然会使用磁盘块缓存机制读取到用户空间中,读取来源是 IDE 磁盘。

这个 IDE 磁盘是哪里来的呢?在 JOS 里是以镜像文件由 QEMU 模拟成 IDE 磁

盘的,产生方式在 fs/Makefrag 里有详细过程。
将 fs/fsformat.c 编译成可执行文件(以下行数可能有异，重意即可)
(a) 第 32 行:将 fs/fsformat.c 编译成可执行文件
(b) 第 38 行:使用 fs/fsformat 可执行文件产生镜像文件 fs/clean-fs.img
(c) 第 43 行:将 fs/clean-fs.img 复制成真正的磁盘镜像文件 fs/fs.img

可以看到这个 fs/fsformat 接受了 1024 和一堆文件系统的目标文件生成了 clean-fs.img,那么我们好奇这个 fs/fsformat 是干吗的呢?打开 fs/fsformat.c 来看看。

fs/fsformat.c 是一个创建磁盘的工具,我们就看看它的主函数:

(a) 使用 opendisk 创建一个磁盘文件,超级块在这里被初始化
(b) 使用 startdir 创建根目录,并初始化超级块
(c) 使用 writefile 将目标文件写入磁盘映像
(d) 使用 finishdir 将根目录写入磁盘映像
(e) 使用 finishdisk 将块位图设置为正确的值,完成磁盘映像的创建

其中 opendisk 就是我们初始化 Super 超级块的地方。
这里创建了映像文件,并为 Super 和 bitmap 分配好了空间(在文件中留出
相应大小的空间)

4. 一个搞错的概念: File 结构中无论是 f direct 还是 f.indirect,他们存储 的都是指向的物理磁盘块的编号!如果要对指向的磁盘块进行读写,那么必须用 diskaddr 转换成文件系统地址空间后才可以进行相应的操作。

5. 即使完成了 spawning 部分，实际上还是没有完全知道 spawn 函数的流程，探究一下?

答:

1)从文件系统打开对应的文件,准备从文件中读取 ELF 内容信息;

2)使用 exofork 创建子进程;

3)为子进程初始化堆栈空间(a) 因为对于该进程还有相应要传入的参数,所以要将这些参数合理的安排进用户栈中(b) 将子进程的 esp 栈顶指针设置为合适的位置;

4)将文件对应的 ELF 文件载入到子进程的地址空间中,记住在这里要为他们分配物理页面;

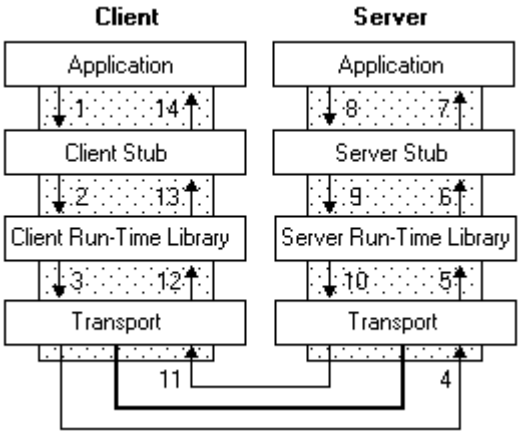
5)设置子进程的各个寄存器,eip 等于 ELF 文件的入口地址;

6)设置子进程为可运行;

6. let's go deeply into RPC

我找到了这里的资料（http: //msdn.microsoft.com/en-us/library/aa374358(v=VS.85).aspx ）。

The RPC tools make it appear to users as though a client directly calls a procedure located in a remote server program. The client and server each have their own address spaces; that is, each has its own memory resource allocated to data used by the procedure. The following figure illustrates the RPC architecture.



As the illustration shows, the client application calls a local stub procedure instead of the actual code implementing the procedure. Stubs are compiled and linked with the client application. Instead of containing the actual code that implements the remote procedure, the client stub code:

1. Retrieves the required parameters from the client address space.

2. Translates the parameters as needed into a standard NDR format for transmission over the network.

3. Calls functions in the RPC client run-time library to send the request and its parameters to the server.

The server performs the following steps to call the remote procedure.

1. The server RPC run-time library functions accept the request and call the server stub procedure.

2. The server stub retrieves the parameters from the network buffer and converts them from the network
transmission format to the format the server needs.

3. The server stub calls the actual procedure on the server.

The remote procedure then runs, possibly generating output parameters and a return value. When the
remote procedure is complete, a similar sequence of steps returns the data to the client.

1. The remote procedure returns its data to the server stub.

2. The server stub converts output parameters to the format required for transmission over the network
and returns them to the RPC run-time library functions.

3. The server RPC run-time library functions transmit the data on the network to the client computer.

The client completes the process by accepting the data over the network and returning it to the calling
function.

1. The client RPC run-time library receives the remote-procedure return values and returns them to the
client stub.

2. The client stub converts the data from its NDR to the format used by the client computer. The stub

writes data into the client memory and returns the result to the calling program on the client.

3. The calling procedure continues as if the procedure had been called on the same computer.

The run-time libraries are provided in two parts: an import library, which is linked with the application
and the RPC run-time library, which is implemented as a dynamic-link library (DLL).

The server application contains calls to the server run-time library functions

which register the server's interface and allow the server to accept remote procedure calls. The server application also contains the application-specific remote procedures that are called by the client applications.

这个介绍大致能看清楚 **RPC** 的工作原理。对应到 **JOS**,他们之间的联系是:

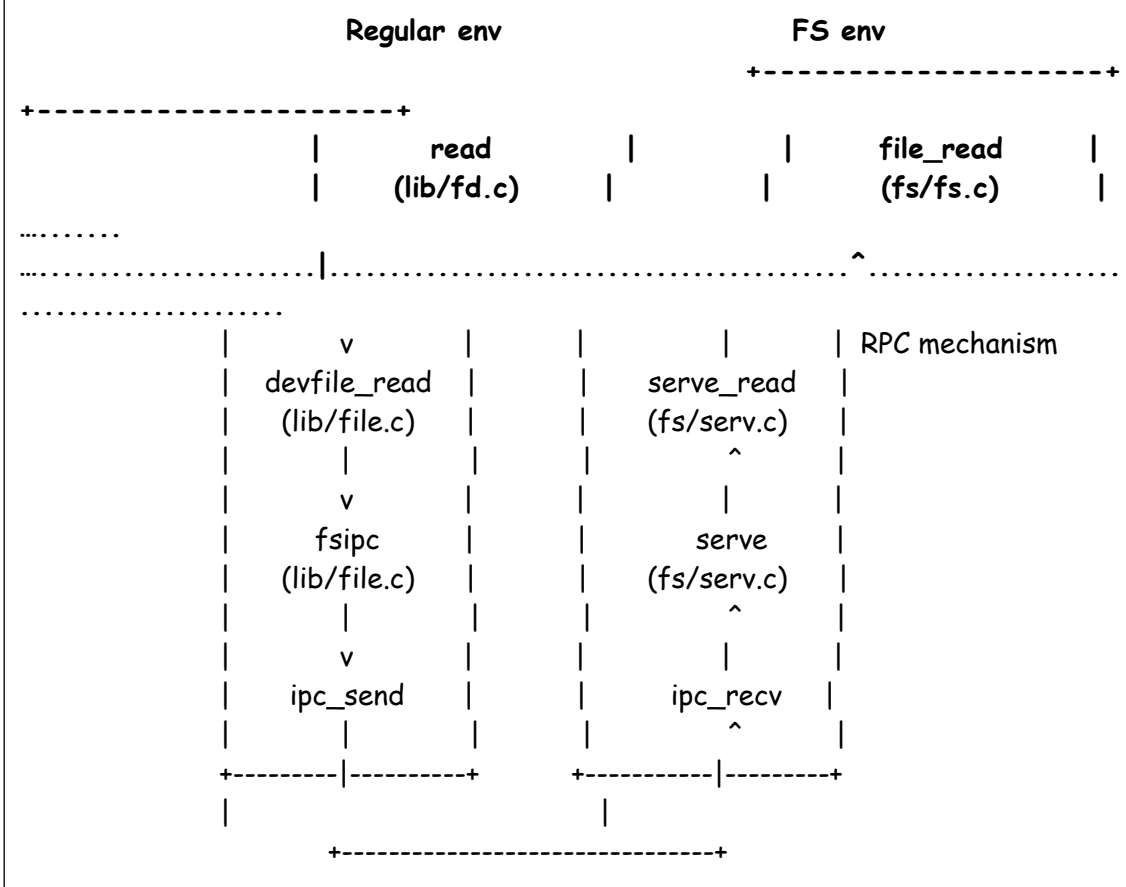
• **RPC** 的最底层传输层可以是 **network**, 这里我们 **JOS** 只是在 **IPC** 上 实现的 **RPC**

• **JOS** 中的 **Server Stub** 即 **fs/serv.c**,**Server Run-Time Library** 即 **fs/fs.c**

• **JOS** 中的 **Client Stub** 即 **lib/fd.c**,用于封装文件传输的细节,实际上一个文件可以对应一个实际文件,也可以是 **Socket**,**Pipe** 之类的

• **JOS** 中的 **Client Run-Time Library** 即 **lib/file.c**,在这里对应真实文件系统的函数调用和数据传输

整个 **C/S** 架构的文件系统调用机制:



下一阶段的计划：

Make the file system fault tolerant: using journaling or soft updates, etc. And try to finish some challenges in labs if time is enough

指导教师意见：

指导教师签名：_____

年　月　日