Operating System-Class 3-Feb. 2nd

Notes by Weiwei Gong

Linux coding style: Love P336

Qual Solution

- need to store filename and position
- 1a: need to know which files are open in app
- 1b: need to add OS_OPEN state

```
struct fileinfo{
    char fname[MAXFN];
    int is_app_open;
    int position
};

static struct fileinfo rfile;
static struct fileinfo wfile;
static int open_os.case;

int openread(char *name)
{
    if(rfile.is_app_open)
        return -1;
    strcpy(rfile.fname, name);
    rfile.position = 0;
    rfile.is_app_open = TRUE;
    return 0;
}
```

need to copy filename, much safer -- because pointer can point to another string (caller can change its string buffer)

allocates memory for struct

NONE, READFILE, WRITEFILE

strncpy is better, stay within

```
1a.   int read()
      {
          int ch;
          if(!rfile.is_app_open)
              return -1;
          if(openf(rfile.fname, rfile.position) < 0)
              return -1;
          ch = readf();
          rfile.position++;
          closef();
          return ch;
      }
1b.   int read()
      {
          int ch;
          if(!rfile.is_app_open)
              return -1;
          if(open_os.case == WRITEFILE){
              closef();
              openf(rfile.name, rfile.position);
              open_os.case = READFILE;
          } else if(open_os.case == NONE) {
              openf(rfile.name, rfile.position);
```

```
            open_os.case = READFILE;
        }
        ch = readf();
        rfile.position++;
        return ch;
    }

int closeread(void)
{
    rfile.is_app_open = FALSE;
    if(os_open.case == READFILE){
        closef(rfile.fname);
        os_open.case = NONE;  /* note writefile could be app-open */
    }
}
```

State Transit Diagram

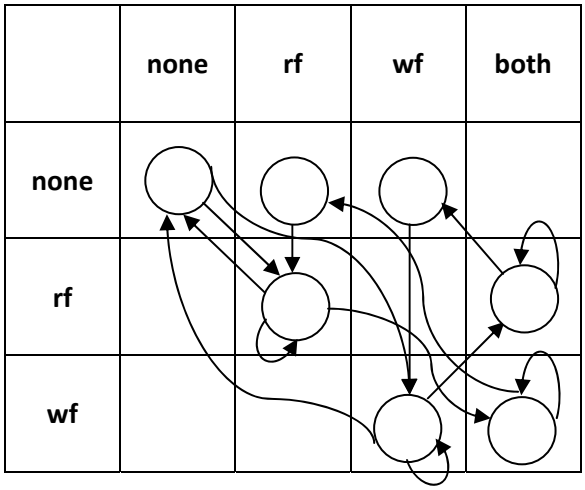
Look at package static variable to see possible states of the system (automatic variables are not important for this, because they are "transient": we are interested only in the effect of the whole call to openread, etc.

rfile.is_app_open:	2 values	} Total: 2*2*3=12
wfile.is_app_open:	2 values	
os_open.case:	3 values	

Using abbreviations: rf=readfile, wf=writefile, we can express these states as follows:

App-open: none, rf, wf, both (can have both open)
OS-open: none, rf, wr (can't have both open)

The whole system is in some (app-open state, OS-open state), starting from (none, none). We can show all the possible states in a grid:



The "extra" states in the first row are caused by closeread OS-closing the readfile while the writefile is app-open, or vice versa.

→ We found 7 states under above code in closeread/closewrite.

- review codes: seeing if it works properly for each of these states
- checking for bad states: system “wedged”, “busy”
- particularly important for OS programming, OS run for days, weeks, months
  - try “uptime” command on Unix/Linux server to see how long the server has been up

Back to Virtual Machine Idea

Process: a program in execution

- OS provide a virtual machine
  - virtual CPU: regular CPU (general registers)
  - address space: appears to be a sequence of memory locations
  - “byte addressed”: each byte of memory has a unique address (can use pointer as unique id’s of memory access)