Myfind Protokoll:

Parallelisierung:

Der Hauptprozess fork()t einen Kindprozess für jedes File, nach dem gesucht werden soll.

```
// start child-processes for each given filename
while(optind < argc)
{
    pid_t pid = fork();

    switch(pid)
    {
    case -1: // error
        std::cerr << "error when forking child process\n";
        waitForAllChildren(startedChildren, fd);
        return EXIT_FAILURE;
    case 0: // child process
        return search(searchPath, argv[optind], iCounter, RCounter, fd);
    }
    ++optind;
    ++startedChildren;
}
```

Kindprozesse suchen parallel nach den gegeben Files. Wenn ein Kindprozess seine Aufgabe erfüllt hat terminiert er mit entsprechendem Rückgabewert.

Synchronisation:

Gestartete Kindprozesse schreiben ihre Ergebnisse in eine unnamed Pipe, die vom Hauptprozess an zentraler Stelle ausgelesen und ausgegeben wird.

```
// read data from pipe until all children close their writing end
while(read(pipe[0], buffer, PIPE_BUF) != 0)
{
    std::cout << buffer;
    memset(buffer, 0, sizeof(buffer));
}
```

Nachdem alle Kindprozesse ihr Ende der Pipe geschlossen haben (ihre Aufgabe abgeschlossen haben), wartet der Hauptprozess darauf, dass alle Kinder terminieren um keine Zombieprozesse zurück zu lassen.

```
// wait for all children to finish
while(finishedChildren < startedChildren)
{
    pid_t pid = wait(NULL);

    if(pid == -1)
    {
        std::cerr << "error when waiting for child process\n";
        return;
    }
    ++finishedChildren;
}
```