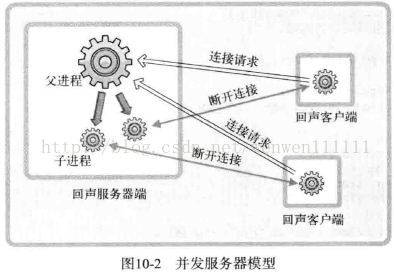
1.基于进程的并发服务器模型



a. 父进程通过调用accept受理连接请求；（连接请求套接字）

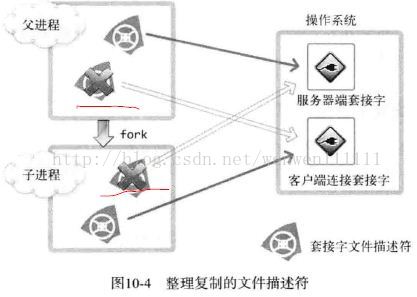
b. 获取套接字文件描述符创建并传递给子进程；

//子进程复制父进程的所有资源

c. 子进程利用传递来的文件描述符提供服务。

1. #include <stdio.h>
2. #include <stdlib.h>
3. #include <string.h>
4. #include <unistd.h>
5. #include <arpa/inet.h>
6. #include <signal.h>
7. #include <sys/wait.h>
8. #include <sys/socket.h>
10. #define BUF\_SIZE    30
11. **void** error\_handling(**char** \*message);
12. **void** read\_childproc(**int** sig);
14. **int** main(**int** argc, **char** \*argv[]){
15. **int** serv\_sock,clnt\_sock;
16. pid\_t pid;
17. **struct** sigaction act;
18. **char** buf[BUF\_SIZE];
19. **int** str\_len,state;
20. **struct** sockaddr\_in serv\_adr;
21. **struct** sockaddr\_in clnt\_adr;
22. socklen\_t adr\_sz;
24. **if**(argc != 2){
25. printf("Usage : %s <port>\n",argv[0]);
26. exit(1);
27. }
29. act.sa\_handler = read\_childproc;
30. sigemptyset(&act.sa\_mask);
31. act.sa\_flags = 0;
32. state = sigaction(SIGCHLD,&act,0);
34. serv\_sock = socket(PF\_INET,SOCK\_STREAM,0);
35. **if**(serv\_sock == -1){
36. error\_handling("socket error");
37. }
39. memset(&serv\_adr,0,**sizeof**(serv\_adr));
40. serv\_adr.sin\_family=AF\_INET;
41. serv\_adr.sin\_addr.s\_addr=htonl(INADDR\_ANY);
42. serv\_adr.sin\_port=htons(atoi(argv[1]));
44. **if**(bind(serv\_sock,(**struct** sockaddr\*)&serv\_adr,**sizeof**(serv\_adr)) == -1){
45. error\_handling("bind() error");
46. }
48. **if**(listen(serv\_sock,5) == -1){
49. error\_handling("listen() error");
50. }
52. **while**(1){
53. adr\_sz = **sizeof**(clnt\_adr);
54. clnt\_sock = accept(serv\_sock,(**struct** sockaddr \*)&clnt\_adr,&adr\_sz);
55. **if**(clnt\_sock == -1){
56. **continue**;
57. }**else**{
58. printf("new client ...\n");
59. }
61. pid = fork();
63. **if**(pid == -1){
64. close(clnt\_sock);
65. **continue**;
66. }**else** **if**(pid == 0){
67. close(serv\_sock);
68. **while**((str\_len = read(clnt\_sock,buf,BUF\_SIZE)) != 0){
69. write(clnt\_sock,buf,str\_len);
70. }
72. close(clnt\_sock);
73. puts("client disconnected...");
74. **return** 0;
75. }**else**{
76. close(clnt\_sock);
77. }
78. }
79. close(serv\_sock);
81. **return** 0;
82. }
84. **void** error\_handling(**char** \*message){
86. fputs(message,stderr);
87. fputs("\n",stderr);
88. exit(1);
89. }
91. **void** read\_childproc(**int** sig){
92. pid\_t pid;
93. **int** status;
94. pid = waitpid(-1,&status, WNOHANG);
95. printf("removed proc id : %d \n",pid);
96. }

示例中fork复制文件描述符，父进程将2个套接字文件描述符复制给了子进程，套接字不会被复制（可以理解为套接字属于操作系统，不属于进程）



在调用fork之后需要将无关的套接字文件描述符关闭。