15-213

"The course that gives CMU its Zip!"

Concurrent Servers

May 3, 2001

Topics

- Baseline iterative server
- Process-based concurrent server
- Threads-based concurrent server
- `select`-based concurrent server

Error-handling sockets wrappers

To simplify our code, we will use error handling wrappers of the form:

```
int Accept(int s, struct sockaddr *addr, int *addrlen) {
  int rc = accept(s, addr, addrlen);
  if (rc < 0)
    unix_error("Accept");
  return rc;
}
```

```
void unix_error(char *msg) {
  printf("%s: %s\n", msg, strerror(errno));
  exit(0);
};
```

Echo client revisited

```
/*
 * echoclient.c - A simple connection-based echo client
 * usage: echoclient <host> <port>
 */
#include <ics.h>
#define BUFSIZE 1024

int main(int argc, char **argv) {
    int sockfd;                     /* client socket */
    struct sockaddr_in serveraddr; /* server socket addr struct */
    struct hostent *server;         /* server's DNS entry */
    char *hostname;                 /* server's domain name */
    int portno;                     /* server's port number */
    char buf[BUFSIZE];

    /* check command line arguments */
    if (argc != 3) {
       fprintf(stderr,"usage: %s <hostname> <port>\n", argv[0]);
       exit(0);
    }
    hostname = argv[1];
    portno = atoi(argv[2]);
```