```
type kind = enum(READ, WRITE, CLOSE);
chan open(string fname; int clientID);
chan access[n](int kind, types of other arguments);
chan open_reply[m](int serverID);  # server id or error
chan access_reply[m](types of results);  # data, error, ...

process File_Server[i = 0 to n-1] {
  string fname; int clientID;
  kind k; variables for other arguments;
  bool more = false;
  variables for local buffer, cache, etc.;
  while (true) {
    receive open(fname, clientID);
    open file fname; if successful then:
    send open_reply[clientID](i); more = true;
    while (more) {
      receive access[i](k, other arguments);
      if (k == READ)
        process read request;
      else if (k == WRITE)
        process write request;
      else   # k == CLOSE
        { close the file; more = false; }
      send access_reply[clientID](results);
    }
  }
}

process Client[j = 0 to m-1] {
  int serverID;  declarations of other variables;
  send open("foo", j);    # open file "foo"
  receive open_reply[j](serverID);  # get back server id
  # use file then close it by executing the following
  send access[serverID](access arguments);
  receive access_reply[j](results);
   ...
}
```

Figure 7.10 File servers and clients.