Chapter 17
Concurrent Clients

Potential Advantages:
–easier to program (modularity)
–easier to maintain and extend
–improved response time by contacting multiple servers
–improved through put

Key advantage: asynchrony—no strict execution order on tasks

Important goal: avoid deadlock

Principle: the client/user/system should be able to overcome a slow or dead server.
–asynchronous wait
–request cancellation
–tasks not requiring the service should continue
–should not lock up interactive users indefinitely

Non-blocking I/O: Read/write calls normally wait for completion, non-blocking versions return "failure" indication if they would be forced to wait.

Concurrent Client Implementations

Option 1: Client divides into two or more processes that each handle one function.

Very useful with good interprocess communication.

Very useful in symmetric multiprocessor machines.

Option 2: Client consists of a single process that uses select to handle multiple I/O events asynchronously.

Usually very efficient, but very complex programming

Examples:

mail–Concurrent: hands off the message and doesn't wait until it is delivered.

df (disk free)–Non concurrent: waits (forever) for remote directories to report.

Design: either make it concurrent, give it a time-out provision, or allow it to be terminated by the user (interactive programs only).