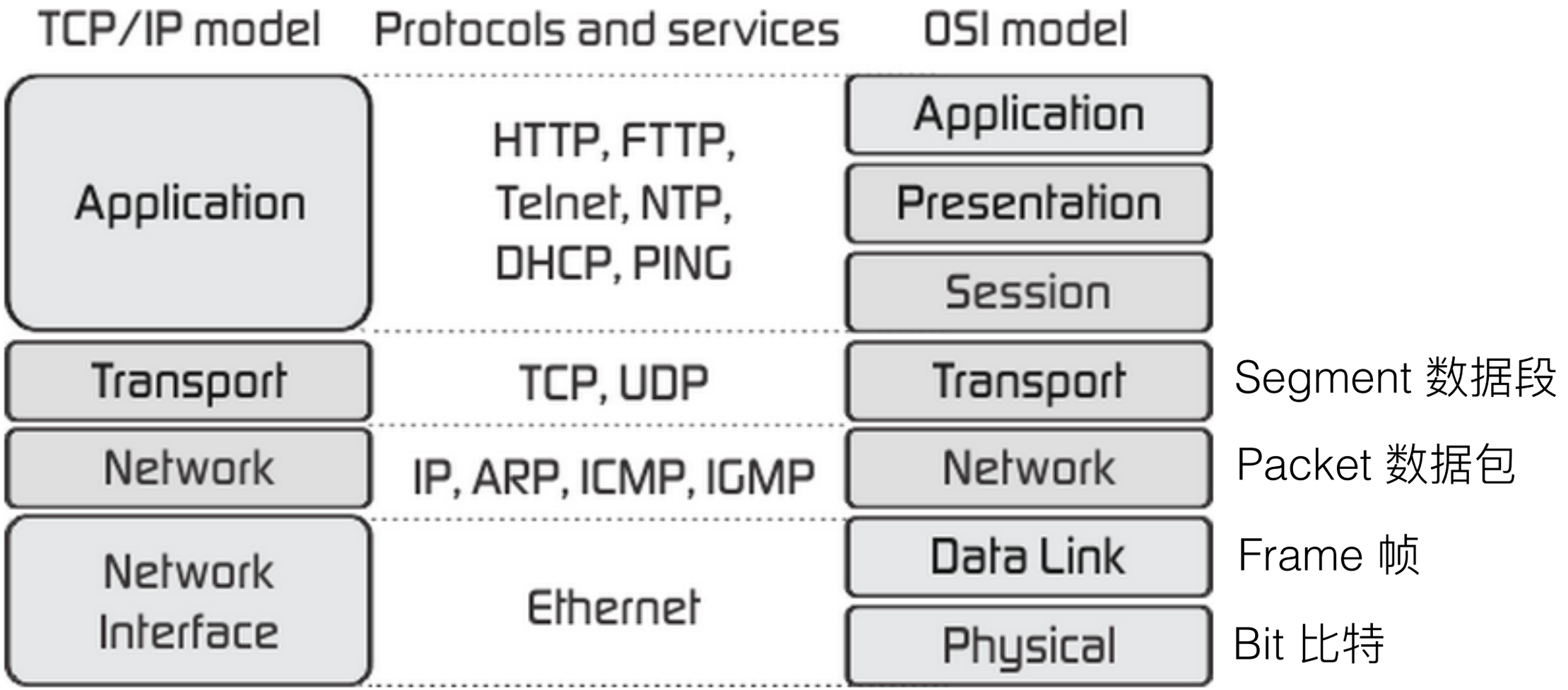
习题课

章 阳
yangzhang@whut.edu.cn
http://yzhang.org

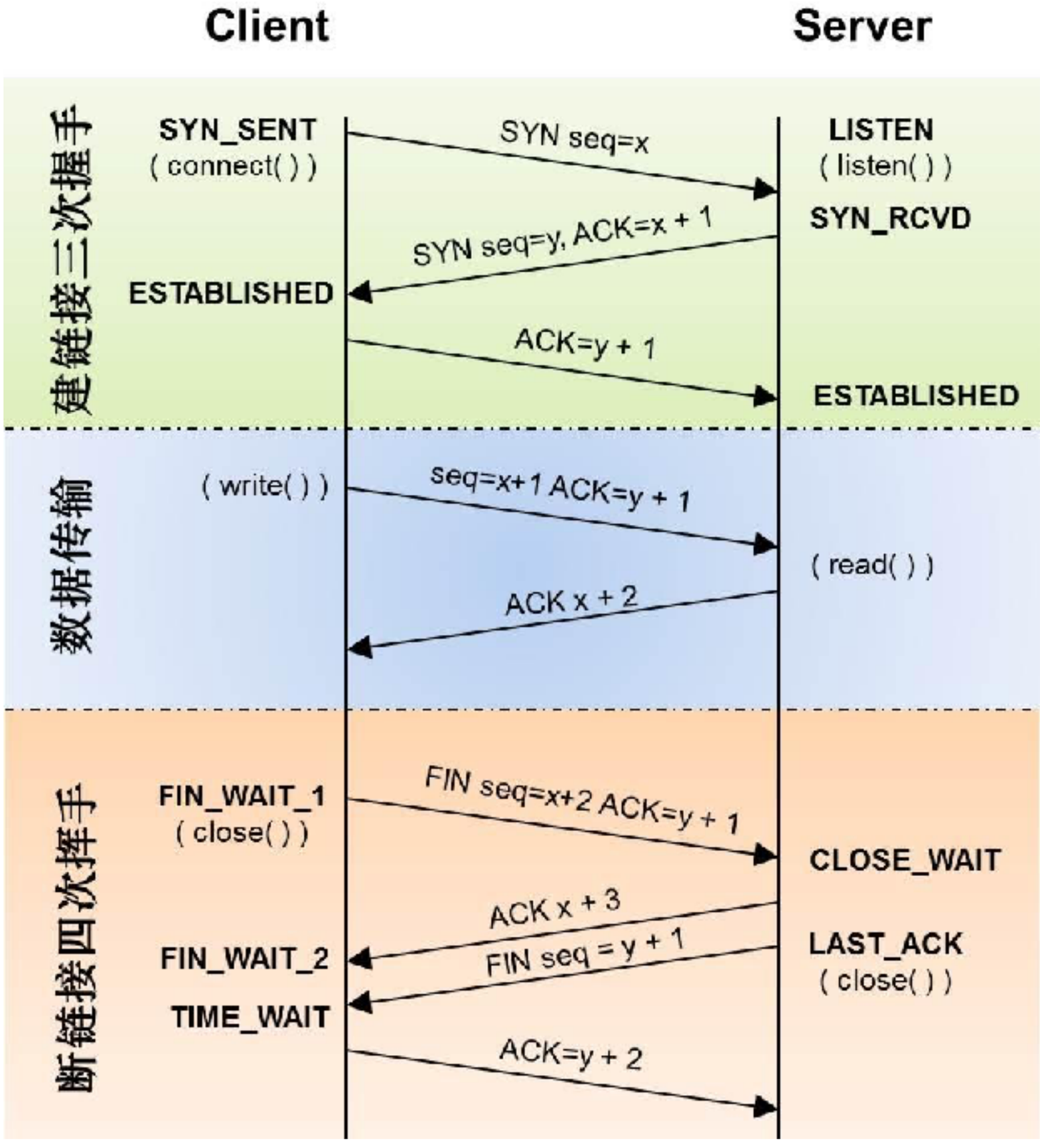
- 1.物理地址（MAC）存在于_____层，IP地址存在于_____层，可以将IP地址分为_____和主机号。


TCP/IP model
Protocols and services
OSI model
Application
HTTP, FTTP,
Telnet, NTP,
DHCP, PING
Application
Presentation
Session
Transport
TCP, UDP
Transport
Segment 数据段
Network
IP, ARP, ICMP, IGMP
Network
Packet 数据包
Network
Interface
Ethernet
Data Link
Frame 帧
Physical
Bit 比特

- 1.物理地址（MAC）存在于_____层，IP地址存在于_____层，可以将IP地址分为_____和主机号。
- （数据链路，网络，网络号）

- 2.要实现网络服务的可靠性需要提供：_____、超时、重传和_____。

- 分析：设计一个网络协议，或一种网络服务的时候，检错-超时-重传-序号，各有什么作用?


Client
Server
建链接三次握手
SYN_SENT
(connect( ))
SYN seq=x
LISTEN
(listen( ))
SYN_RCVD
SYN seq=y, ACK=x + 1
ESTABLISHED
ACK=y + 1
ESTABLISHED
数据传输
(write( ))
seq=x+1 ACK=y + 1
(read( ))
ACK x + 2
断链接四次挥手
FIN_WAIT_1
(close( ))
FIN seq=x+2 ACK=y + 1
CLOSE_WAIT
ACK x + 3
FIN_WAIT_2
FIN seq = y + 1
LAST_ACK
(close( ))
TIME_WAIT
ACK=y + 2

- 2.要实现网络服务的可靠性需要提供：_____、超时、重传和_____。
- （检错，序号）

- 3.发起对等通信的应用程序称为_____，等待接收客户通信请求的程序称为_____。

- （客户/客户端，服务器）

- 对等通信（P2P，Peer-to-Peer）

- 4.在TCP/IP使用中，_____的模式占有主导地位，其动机来源于_____问题。

- （客户/服务器、通信汇聚点）

- 分析：汇聚点问题（Rendezvous dilemma）
  - 空间上的：公园里两人互相寻找
  - 时间上的：快慢通信节点
  - “君生我未生，我生君已老，君恨我生迟，我恨君生早。”

- 5.在UNIX系统中创建新进程，需要调用系统函数_____。
- fork()
- windows下CreateProcess()
- fork的三种返回值，-1, 0, PID（进程ID）

- 6.TCP/IP协议定义的端点地址包括_____和_____。
- （IP地址，端口号）
- :8080, :21

- 7.不保存任何状态信息的服务器称为_____服务器，反之则称为_____服务器。

- （无状态，有状态）

- 分析

  - 无状态服务器：如WEB服务器，HTTP服务

  - 有状态服务器：如游戏服务器

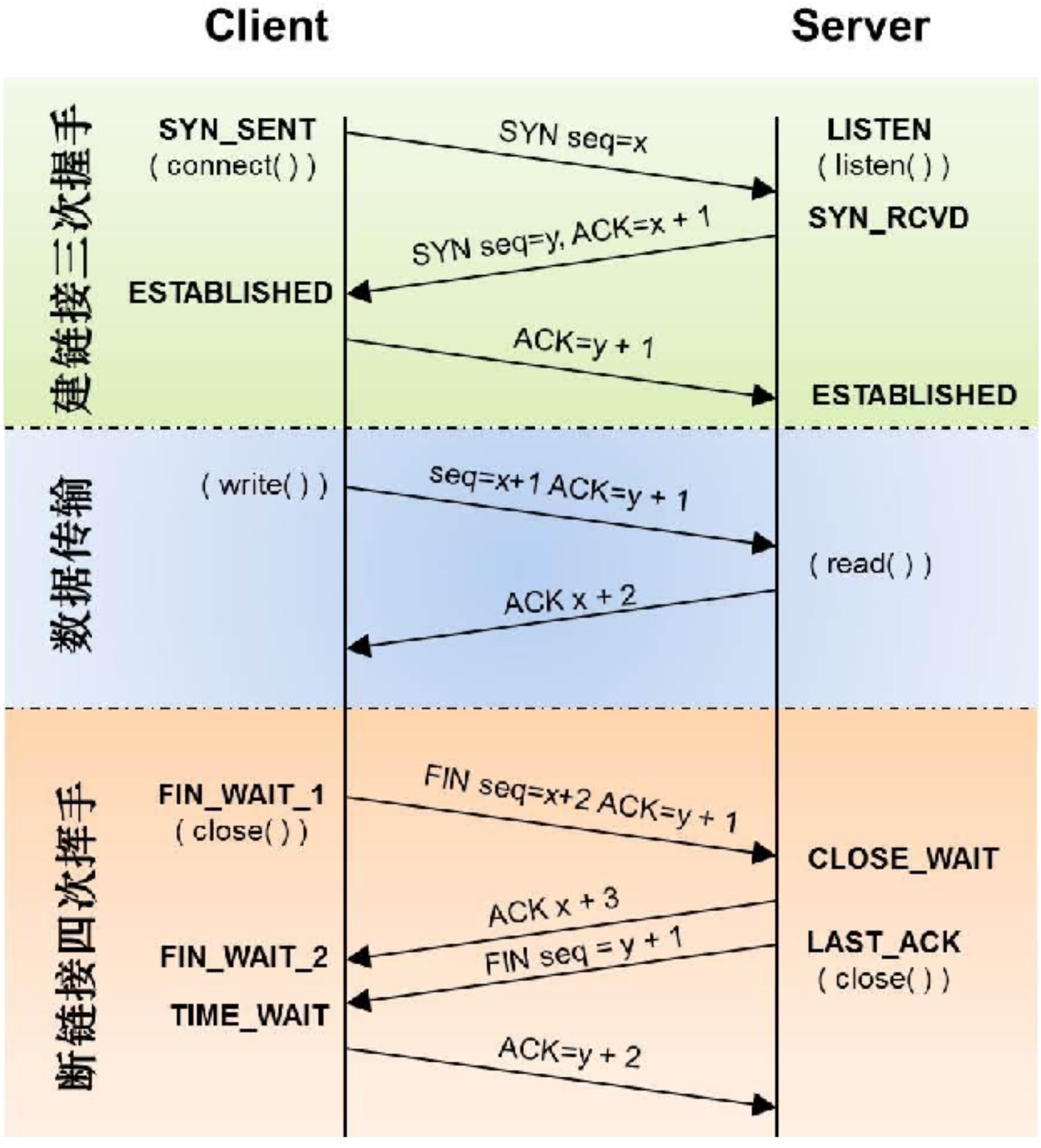
- 8. _____是指真正的或表面的同时计算，一个单处理机多用户的计算机可以通过_____机制实现表面的同时计算，而在多处理机下可以实现真正的同时计算。

- （并发，复用）

- 9. TCP提供面向_____的服务，而UDP提供_____的服务。

- （连接，无连接）

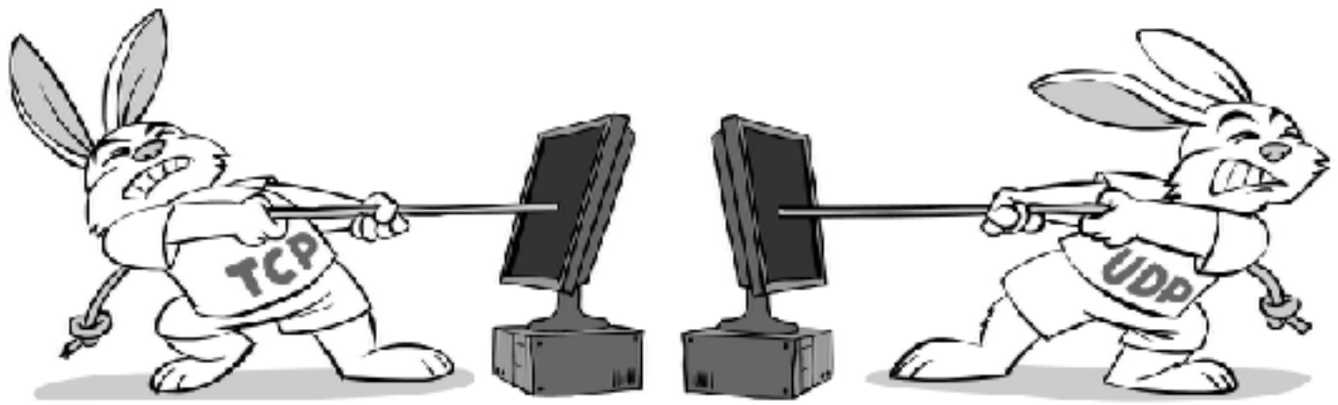
- 分析：TCP-虚连接（虚电路），UDP-尽力发送


Client
Server
建链接三次握手
SYN_SENT
(connect( ))
SYN seq=x
LISTEN
(listen( ))
SYN_RCVD
SYN seq=y, ACK=x + 1
ESTABLISHED
ACK=y + 1
ESTABLISHED
数据传输
(write( ))
seq=x+1 ACK=y + 1
(read( ))
ACK x + 2
断链接四次挥手
FIN_WAIT_1
(close( ))
FIN seq=x+2 ACK=y + 1
CLOSE_WAIT
ACK x + 3
FIN_WAIT_2
FIN seq = y + 1
LAST_ACK
(close( ))
TIME_WAIT
ACK=y + 2

- 判断题：

- 1. 有些场合下只能使用UDP协议进行网络通信 ☑

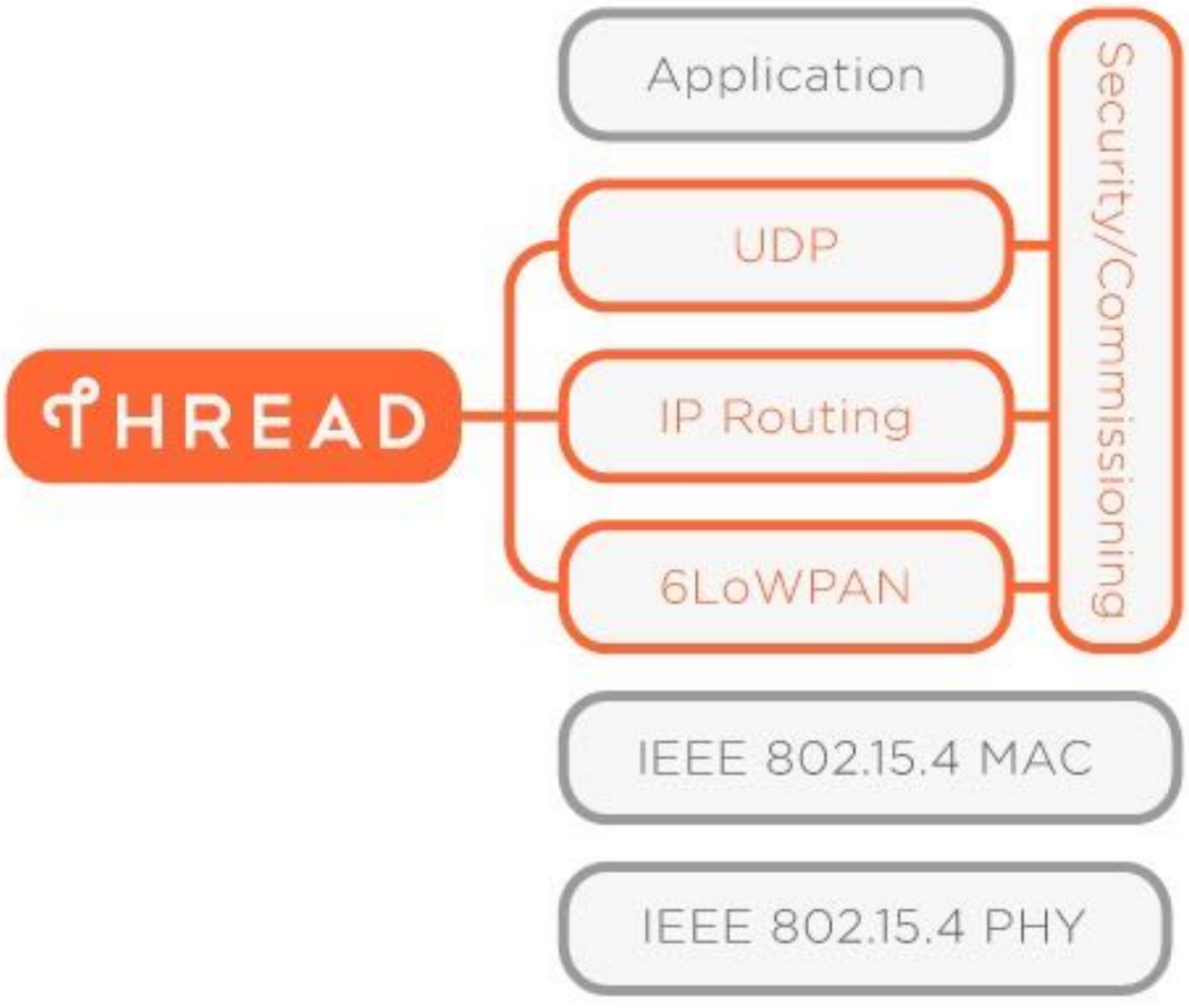
- 提示：内网IM通信；DNS域名解析；VANET；……



Copyright: wilddogbaas

- UDP≠弱
  - UDP不用握手，不提供流控、拥塞等功能，传输不可靠，因此有时更简单有效。（奥卡姆剃刀？）
  - 随着网络环境变好，UDP其实也够了？（<5% packet loss + retransmission by application layer, NDN…）
  - UDP协议简单，冗余功能少，提升空间大

- 物联网与UDP


Application
UDP
IP Routing
6LoWPAN
Security/Commissioning
THREAD
IEEE 802.15.4 MAC
IEEE 802.15.4 PHY

- 判断：

- 2. 服务器使用并发处理可以完全防止死锁?

  - 分析：产生死锁的四个条件：

    - 互斥条件：一个资源被一个进程使用
    - 请求与保持条件：不放弃已获得资源
    - 不剥夺条件：不强行剥夺已获得资源
    - 循环等待条件：进程循环等待释放资源

- 2. 服务器使用并发处理可以完全防止死锁？ ☒

- 判断：

- 3. 发起对等通信的应用程序为服务器 ☒

- 分析：客户端

- 判断
- 4. TCP/IP标准规定了通信双方在什么时间以及用什么方式交互 ☒

- TCP/IP包含了什么?
  - IP：在网络层
    - 数据传输单元和格式，数据递交方法和路由
  - TCP：在传输层
    - 数据包检查、超时检测、处理
    - 面向连接的服务
- UDP
  - 无连接的服务

- 判断：

- 5. 客户程序可以将服务器的IP地址或域名说明为常量 ☑

- 分析：

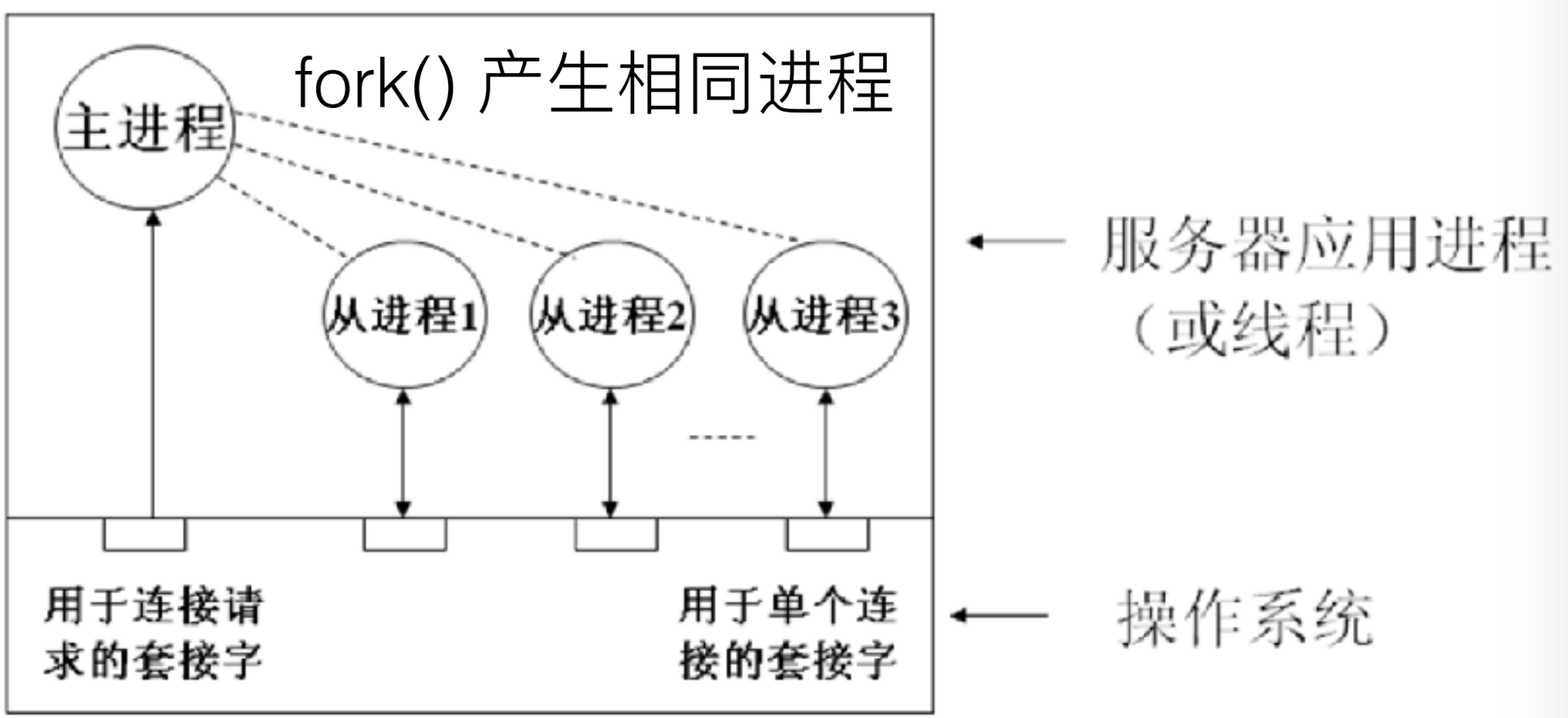
```
const string UNIV_HOMEPAGE_URL= “http://
www.whut.edu.cn”
```

- 判断：

- 6. TCP提供流量控制和拥塞控制☑

- 分析：TCP有，UDP没有

- 7. 并发的、面向连接的服务器可以有n个不同的进程
  ☒



- 8. 只能在TCP通信时使用connect系统调用☒
- 分析：TCP和UDP均调用connect()建立连接。区别是?

- TCP与UDP调用connect()的区别
  - TCP调用connect - 已知
  - UDP调用connect：只记录对方IP和端口，只探测调用时连接错误
- UDP不再用sendto，而直接用send、write - 一个“已连接的无连接”
- UDP可以多次调用connect用于重新连接新的IP和端口，TCP不行

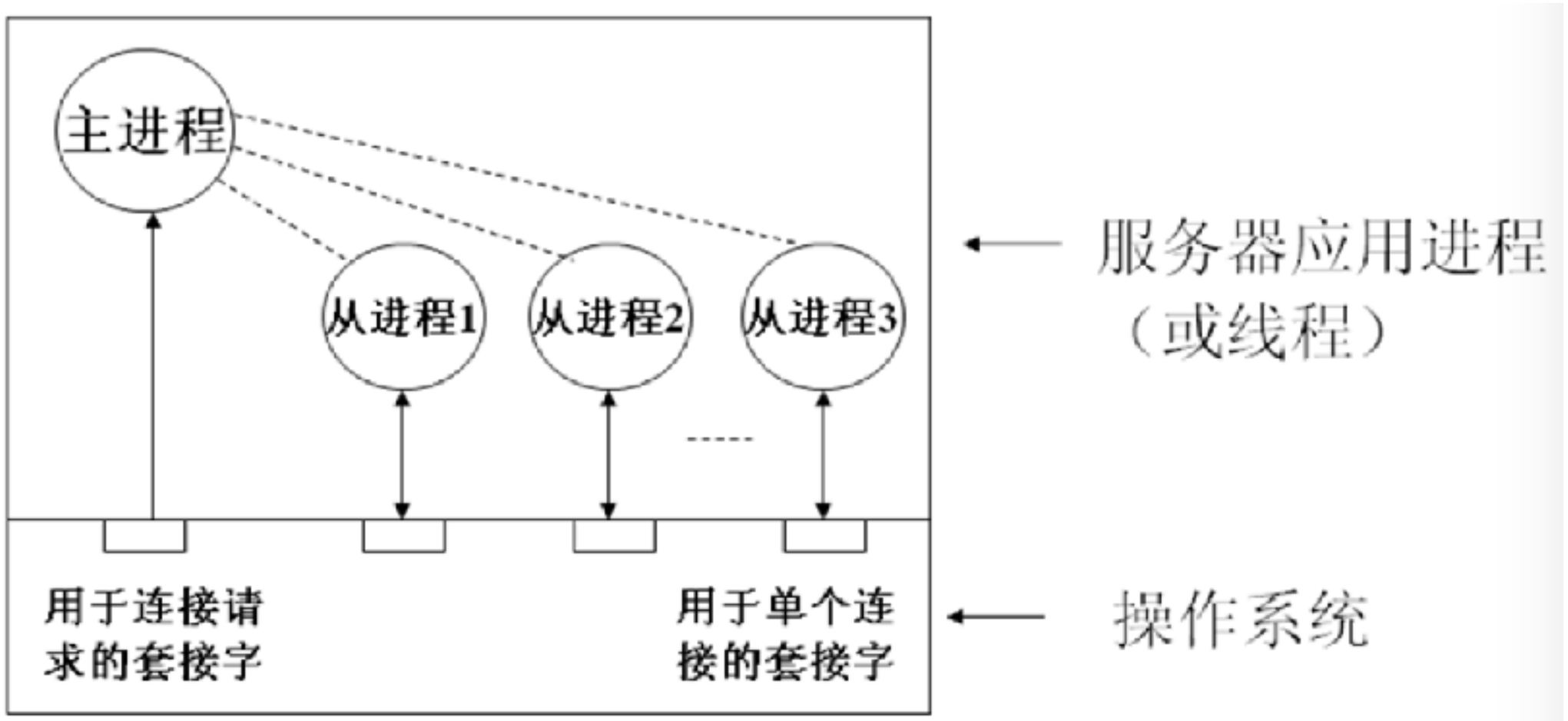
- 判断
- 9. TCP/IP地址族可以表示为PF_INET ☑?
  - 分析：在头文件netinet/in.h，Berkeley规范
    - PF=Protocol Family （BSD里=AF）
    - AF=Address Family
    - AF_INET, AF_INET6, PF_INET（incl. TCP, UDP）, PF_INET6

- 10. 面向连接的服务易于编程。⊠?
  - 分析：更加复杂还是更加简单?
    - 需要保持连接、有额外控制开销，如差错控制、流量控制
    - 保证投递；按序提交

- 问答：
- 1. 请给出并发的面向连接服务器（多进程）设计算法，图示出进程结构，并说明这种类型的服务器的优缺点。

- 面向连接的服务器在多个连接之间实现并发（不是在各个请求之间）
  - 主1、创建套接字并将其绑定到所提供服务的熟知地址上。让该套接字保持为面向连接
  - 主2、将该端口设置为被动模式
  - 主3、反复调用accept以便接收来自客户的下一个连接请求，并创建新的从线程或者进程来处理响应
  - 从1、由主线程传递来的连接请求开始
  - 从2、用该连接与客户进行交互；读取请求并发回响应
  - 从3、关闭连接并退出

- 优点：一个时刻可以处理多个请求，性能好
- 缺点：实现复杂，难以构建和设计



- 2. 试分析面向连接的服务器和无连接的服务器各自的优缺点。

- 回顾：面向连接和无连接
  - 面向连接：是电话系统服务模式的抽象，即每一次完整的数据传输都要经过建立连接、使用连接、终止连接的过程。
  - 无连接：是邮政系统服务的抽象，每个分组都携带完整的目的地址，各分组在系统中独立传送。

- 试分析**面向连接**的服务器和**无连接**的服务器各自的优缺点。
- 面向连接的服务的优点：
  - 1. 易于编程
  - 2. 自动处理分组丢失，分组失序
  - 3. 自动验证数据差错，处理连接状态
- 面向连接的服务的缺点：
  - 1. 对每个连接都有一个单独的套接字，耗费更多的资源，维护开销大
  - 2. 在空闲的连接上不发送任何分组
  - 3. 始终运行的服务器会因为客户的崩溃，导致无用套接字的过多而耗尽资源
- 无连接服务器优点：没有资源耗尽问题
- 缺点：需要自己完成可靠通信问题，必要时，需要一种自适应重传的复杂技术，需要程序员具有相当的专业知识。对于可靠通信的场合，尽量使用TCP

- 3. 将一组大程序分解为一系列的子程序/过程的好处是什么？试分析，在客户程序的设计实现时，为什么为什么要先抽象为connectTCP(machine, service)和connectUDP(machine, service)两个模块，而这两个模块又具有共同的底层模块connectsock?

- 将一组大程序分解为一系列的子程序/过程的好处是什么?

  - 一个模块化的程序比一个等价的单个程序容易理解、排错和修改。如果程序员认真的设计了过程，还可以在其他程序中重新使用这些过程。另外，仔细选择过程可以使程序更容易移植。

  - 过程通过将细节隐藏起来，提高了程序员所用语言的级别。构造客户和服务器的时候，使用网络服务的程序包括了一大堆枯燥的细节，（如端点地址等）使用过程来隐藏细节将减少出错的机会。

- 使用过程（以及它所提供的较高级的的操作）可以避免重复，使用者可以在许多程序中使用他们，不需要再考虑实现的细节。

- 试分析，在客户程序的设计实现时，为什么为什么要先抽象为connectTCP(machine, service)和connectUDP(machine, service)两个模块，而这两个模块又具有共同的底层模块connectsock?

    - 多数代码需要分配套接字、绑定地址并构成网络连接，这些重复出现因而可以重用；TCP/IP网络是异种网互联，代码需要运行在不同机器的体系结构上，因而便于移植。

    - 客户端应用程序请求服务只能通过传输层，而传输层有两种协议TCP和UDP，因此抽象的时候可分为两种情况。

    - 连接需要指明服务器的IP地址和服务类型（端口号）。客户端采用TCP和UDP共同的步骤都是获得套接字ID，因此可以考虑将获得套接字ID的过程合并，仅仅采用参数来标明到底采用何种传输层协议。同时仍然需要传递的参数为服务器的IP地址和服务类型（端口号），所以将底层共用一个过程connectsock。

- 4. 在I/O复用模型的关键是熟练掌握select函数，该函数的原型是

```
int select(int maxfd, fd_set *readset, fd_set
*writeset, fd_set *exceptset, const struct timeval
*timeout);
```

- 1、 请详细解释select函数的参数的意义，以及执行结果。
- 2、如何利用select函数构造一个最简单的多协议服务器，同时提供TCP和UDP服务？画出进程结构图。

```
int select(int maxfd, fd_set *readset, fd_set
*writeset, fd_set *exceptset, const struct timeval
*timeout);
```

- select函数功能：
  - 非阻塞（与recv，recvfrom区别）
  - 对套接口集合扫描（fd文件描述字file descriptor）
  - 若扫描有消息，则阻塞处理，反之则返回

```
int select(int maxfd, fd_set *readset, fd_set
*writeset, fd_set *exceptset, const struct timeval
*timeout);
```

- maxfd参数是所以监视的描述字中最大的描述字加1
- 中间三个参数分别表示监视的不同条件的描述字集合：readset为读描述字集合，writeset为写描述字集合，execptset为异常描述字集合。
- timeout参数为select函数最长等待时间。

```
int select(int maxfd, fd_set *readset, fd_set *writeset, fd_set
*exceptset, const struct timeval *timeout);
```

- Select函数有三种执行情况：
  - 永远等待下去：仅在有一个或以上描述字准备好i/o才返回，为此，我们将timeout设置为**空指针**。
  - 等待固定时间：在有一个描述字准备好时返回，但不超过由timeout参数指定的秒数和微秒数。
  - 根本不等待，检查描述字后立即返回，这称为轮询。这种情况下，timeout必须指向结构timeval，且**定时器的值必须为0**。

```
int select(int maxfd, fd_set *readset, fd_set *writeset,
fd_set *exceptset, const struct timeval *timeout);
```

- select执行结果（返回值）
  - 如果在指定超时值到达之前有一个或多个描述字满足条件，则函数返回值大于零；
  - 如果超时时间到时，没有描述字满足条件，函数返回值为0；
  - 如果select函数阻塞过程中，发生错误，函数返回值为－1；

- 如何利用select函数构造一个最简单的多协议服务器，同时提供TCP和UDP服务？画出进程结构图。

- 要点：
  - 多协议服务器可以由一个执行线程构成，该线程既可以在TCP也可以在UDP上使用异步IO来处理通信。
  - 服务器最初打开两个套接字，一个使用无连接的传输，一个使用面向连接的传输，使用异步IO等待两个套接字之一就绪。