# node,port和线程的关系

# 登录时角色对象的创建

# 登出时角色对象的销毁

# 进入副本时角色对象的复制以及退出时对象的覆盖

# **进入副本时线程的切换？**

# **如何解决网络延迟问题**

网络性能指标一：带宽，限制了实时游戏的人数容量

网络性能指标二：延时，决定了实时游戏的最低反应时间

帧间同步：不同客户端每帧显示相同的内容，键盘/时钟数据传到服务器，服务器确认后所有终端做出响应，多用于局域网游戏，比如红警（需要等待客户端），街霸II的网络版（360），可参考 LockStep，TimeWrap算法，**网速要求高，复杂度低**，见我的旧文[帧锁定算法](https://link.zhihu.com/?target=http://www.skywind.me/blog/archives/131" \t "_blank)。

插值同步：不同客户端显示不同步，但是状态同步，常见的Dead Reckoning（或叫导航插值），**效果好，但复杂度高**。常见于竞速类游戏和 FPS游戏。

客户端和服务器是如何互动

游戏中所有的逻辑判定都是由服务器完成的，客户端只负责发送请求和接收服务器的反馈，并把反馈具象化。拿CSGO做例子吧，比如你（玩家A）拿着AK瞄准了玩家B的头开了一枪，那么你的客户端会向服务器发送一个数据包，里面包含了谁（你）拿着什么武器（AK）从什么位置（你在地图上的坐标）向什么方向（角度）开了一枪。服务器收到后进行判定，这一枪的伤害会经过玩家B的头部模型，判定为爆头伤害，数值为104（这个值是瞎编的），因此判定玩家B死亡。然后向所有玩家的客户端通信，更新当前游戏状态，其中会包括玩家A用AK爆头击杀了玩家B，也会包括其他游戏信息，比如玩家的位置等等。你（玩家A）收到通信后再你的GUI上显示你击杀了玩家B，而玩家B则会收到被你击杀的信息。

服务器的通信频率（tick rate）

Hz: 每秒的周期次数(周期/秒),大多数FPS的通信频率在30Hz左右，而CSGO为60Hz,电子竞技比赛时一般服务器的通信频率还会提高到120Hz（因为比赛时大家都是在一个局域网里所以延迟很小高频通信能够更加精确反映游戏内状态）.

10Hz:（0.1秒才更新一次，人类的反应时间差不多略小于0.1秒，即玩家已经可以感觉到其中的不连贯）

CSGO在娱乐模式下通常采用60Hz的频率，也就是说每过1/60秒玩家的客户端就会收到一次新的信息

100ms: 你（玩家A）与服务器之间存在100ms的延迟

客户端的策略

延迟补偿策略: 称为“客户端预测法”。也就是说客户端能够大致预测游戏未来的走向，因此在接收到服务器更新前会把预测到的画面先绘制出来（比如移动，武器的开火效果，弹药计数的变化等等

插帧法: 就是客户端会记录之前一次从服务器收到的信息，然后在接受到下一次通信的时候不立刻更新游戏画面，而是逐渐的更新画面（比如两次通信间玩家B移动了10单位距离，客户端会绘制玩家B以一定的速度移动了这10单位距离，而非立刻绘制玩家B瞬间移动了10单位距离）。 插帧法的问题在于如果玩家并未沿直线运动且其直线路径中有本应不能通过的物体在（比如绕过一堵墙），客户端会绘制出该玩家穿墙而非绕过去的动作。

服务端的策略：

1. “眼不见为净”法。很多时候服务器不去补偿玩家的延迟是一个合理的做法，特别是如果游戏内进行的事件非常多（想想行星边际2里面千人同图混战的情形）。浪费宝贵的服务器资源去补偿个别玩家的延迟是不明智的。这个策略的缺点很明显，就是玩家有可能会对游戏体验不满意。

2.“倒带”法。采用倒带法的服务器会记录刚刚过去一段时间内（比如0.5秒）游戏内的所有信息。当一个有延迟的玩家（比如200ms）向服务器发送一个请求，那么服务器***在处理这个请求的时候***会调取0.2秒前游戏的状态然后进行判定，在把判定结果对所有客户端进行同步。如此一来该玩家的操作虽然有延迟但也能与他/她所看见的画面一致。 该策略的最大问题在于它让不同延迟之间的玩家被迫体验较大的延迟。举个例子，假设游戏里击杀时间（TTK）足够小，玩家A（10ms延迟）和玩家B（延迟200ms）对射，两人都是空血（一次攻击即死），A比B先开火（时间差很小，比如50ms）。玩家A的次攻击很快（10ms后）就得到了处理并记录在服务器中，玩家B被判死亡。然而在200ms后玩家B的请求到达，服务器倒带0.2秒，此时玩家AB都未死亡，因此玩家B的攻击有效，玩家A也被判定为死亡。 如果没有延迟，那么服务器应该会判定玩家B死亡，因此玩家B将无法攻击，玩家A应该存活。换句话说采用倒带法的服务器里如果有一个延迟很大的玩家将会拖累其他低延迟玩家的游戏体验。

# **FPS的意义，网络延迟如何解决**

# **TPC的黏包，调包，以及分包**