## 1.1 概念

什么是跨服？

曾经有这么一个人，他一天在玩游戏的时候，发觉到好多朋友都在不同的服务器玩，他知道总有一天会合服，但是等待那天需要多久？于是他产生了这么一个想法，能不能让所有的玩家都进入到一个服务器来一起玩？这样他就不用等到了合服的那天，他就可以随时与自己的朋友一起玩。他于是将这样的想法同很多游戏的朋友说了，结果他发觉还真有不少志同道合的朋友，于是他们开始将这个建议和需求反馈给游戏制作者。

上面只是一个故事，大致可以反应出跨服的来由，其实就一个道理：有需求才有生产。想想在我们工作生活中，难道不就是这样的道理吗？游戏制作者起先觉得这样实现起来有些难度，但是由于反应的人越来越多，他们就开始思考怎样去实现。其实再难的东西也有实现的一天，于是某一天如雨后春笋一般出现在了很多游戏里。

在这里我们要区分下服务器数据互通与跨服的区别，其实服务器数据互通听起来好像就是跨服一样，其实在现实中这个称呼一般是指合服。而跨服的概念，却是指不同服务器的玩家角色出现在同一个服务器的场景中。

## 1.2 难点

跨服最大的难点是如何同步玩家的数据，因为本来玩家的数据是挂载在自己所在服务器上的，不同的服务器所连接的数据库不相同。

将不同服务器的玩家加入同一台服务器上，在游戏服务器中如果产生了数据的改变，我们不可能使用原来正常的逻辑来保存玩家的数据。这其中的原因，自然是这些别的服务器玩家的数据不应该保存在跨服的服务器上，也不太可能去保存，因为保存的时候在库中完全找不到玩家的数据。

其次还有一个难点就是这些跨服的玩家应该进入哪个服务器的问题。玩家首先连接到的是自己所在的服务器，那么跨服的时候又该去哪个服务器？

## 1.3 思路

针对玩家跨服后何去何从的问题，我们有必要规定一个跨服主服务器，也可以称之为报名服务器。玩家报名之后，在报名服务器之上就形成了唯一的房间，直到房间无效。

针对数据保存的情况，需要根据服务器模型架构来决定，在这里我们将使用两种不同的模式进行实现思路的探索。

重量级：中心服务器作为数据的中转中心负责玩家数据的维护。中心服务器上有一个跨服游戏服务器组的配置，该配置里记录了所有改组的所有中心服务器和游戏服务器的信息。所有跨服的服务器启动的时候，需要连接在跨服组中的所有中心服务器，中心服务器接收到消息之后会将连接的服务器信息记录在一张内存表中。（中心服务器作为中转站的功能，它需要记住这些服务器）

玩家通过客户端报名，报名信息首先发送到自己的服务器，玩家所在的服务器收到此消息又将消息推送给中心服务器。此时中心服务器会通过逻辑判断，再找出是否报名服务器已连接，如果报名服务器存在则发送消息到报名服务器上，报名服务器将结果返回到中心服务器上，中心服务器又将此消息转回玩家当前的服务器，当前的服务器才将此结果返回给客户端。同理在玩家进入报名服务器的逻辑，也是一样，都由中心服务器中转信息。

玩家从报名服务器切换场景的时候与没有跨服的服务器的区别，是报名服务器要将玩家切换场景的消息发送到玩家所在服务器组的中心服务器，在这里玩家的连接上的中心服务器ID的属性也就派上了用场。正确的找到了玩家所在的中心服务器，那么接下来的流程就和之前没有任何区别。

轻量级：这里没有了中心服务器，因为它的功能基本上已经放到了游戏服务器里，所以针对轻量级，跨服的服务器之间需要有连接。原理其实与重量级的原理差不多，不过没有了中心服务器的中转，反而更清晰些。

还是一个游戏服务器作为报名服务器，当跨服服务器启动的时候，就会连接跨服组里的所有游戏服务器，并记录下跨服服务器的信息。当玩家报名的时候，当前的服务器就直接将报名消息发送到报名服务器上，报名服务器记录数据并返回给当前服务器，当前服务器将结果返回客户端。

进入场景（需记录返回的场景信息）的时候，也同报名一样的流程，在返回场景的时候，会根据玩家所在服务器ID将玩家送回到当前的服务器。