# FPS实现方式

## 前后端通讯采用方案：

采用帧同步通讯方案。

通讯协议采用UDP协议来保证数据传输速度。

网络拓扑结构采用星形拓扑结构，来管理客户端的连接。

## 前后端交互协议：

采用谷歌的PB协议。

## 数据存储

采用mysql关系型数据库进行数据存储。

## 游戏引擎

采用unity3d 2018。

## 编程语言

后端使用java的异步服务器。

前端使用c#+lua的形式来编写客户端。

## 游戏资源

游戏资源来自UnityAssetStore中的免费资源、以及爱给网上的免费资源、以及Mixamo等。

## 采用UDP协议需要解决的问题

由于UDP传输数据不会建立链接，也没有保证数据的时序性、数据是否丢包、根据数据的大小对数据进行组合或拆分等问题；所以我们需要解决的问题：

1. 数据是否丢失。
2. 数据的时序性。
3. 根据数据大小对数据进行粘包、拆包。

对游戏中的操作指令按优先级进行分类：

需要分析游戏中不同操作对应的优先级，并且服务器发送数据时根据确认的优先级进行数据的发送。

实现可靠的UDP：

1. 在发送数据的时候添加重传定时器，保证丢失的数据会被重传。重传的定时器可以定时回调，发送重传的数据，也支持将接受到ACK的数据从定时器中取出。
2. 定时器的间隔时间：
   1. 设置一个固定的重传时间
   2. 最合理的，针对每条传输链路设置不同的重传时间，rto时间。

查找rto时间的方法：

我们获取数据从发送到收到确认信息的时间差： 在发送的数据头里添加当前发送时间戳，另一端接受到数据后，需要将收到的时间戳放入ACK数据中一起送回。 此时在接受到的ACK中就存在该消息的准确发送时间，所以我们可以获得准确的rrt时间。 我们按照TCP的标准方法《CP/TPxian详解卷一, P465》，计算rto时间。

当接受到ACK确认时，我们需要将确认的数据从计时器中移出。