Rust分布式服务器开发兴趣题

设想一个2D的游戏世界，范围不超过(-1000000,-1000000)-(1000000,1000000)，单位：m。

游戏世界中仅有一种对象：用户

```
struct Vector2 {
    x: f32,
    y: f32,
}

struct User {
    id: u64,
    position: Vector2,
    velocity: Vector2,
    money: u64,
}
```

设计一个游戏服务器，支持：

1. 用户登录：login(id: u64)
    a. 如果用户从未登录过，服务器创建用户，position=随机，velocity=0，money=0，不返回任何信息
    b. 否则不返回任何信息
2. 用户登出：logout(id: u64)
3. 设置速度：set_velocity(id: u64, velocity: Vector2)，修改用户的速度velocity，velocity.len不会超过20m/s
4. 撒钱：aoe(id: u64, radius: f32, m: u64)，用户周围Radius半径的圆内的所有用户身上的money+=m，自身的钱不变，radius不超过50m
5. 查询：query(min: Vector2, max: Vector2)-Vec<User>，查询(xmin,ymin)-(xmax,ymax)范围内的所有用户信息，query的范围可能很大，甚至是整个世界，但客户端的预期是，query的耗时和query结果的数据量呈近似线性关系

服务器启动后，接受客户端的请求（可能会有多个客户端连接服务器，但每个客户端仅代表一个用户），并对用户的行为进行模拟。每0.02秒为一个step，每个step中，模拟：position+=0.02*velocity;

服务器是一个分布式系统，需要考虑：

1. 高可用
2. 负载均衡
3. 可观测性

注意，用户移动的行为模式和真实世界的人的行为模式接近，速度不高，但是有聚集效应。

典型的测试场景是，同时连接的用户数量是单节点负载能力的100倍。