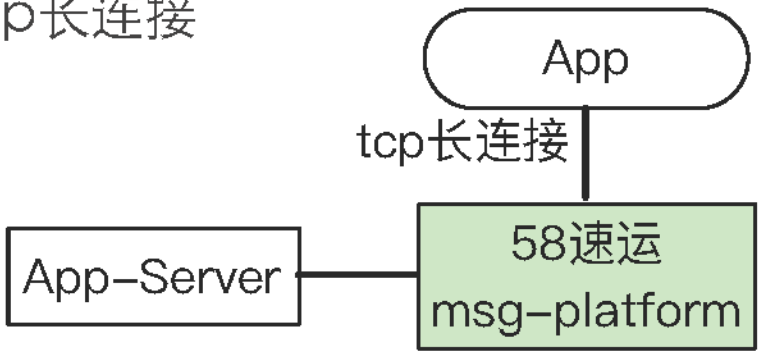
业务方诉求

高性能 高可用 实时 高到达率

消息平台

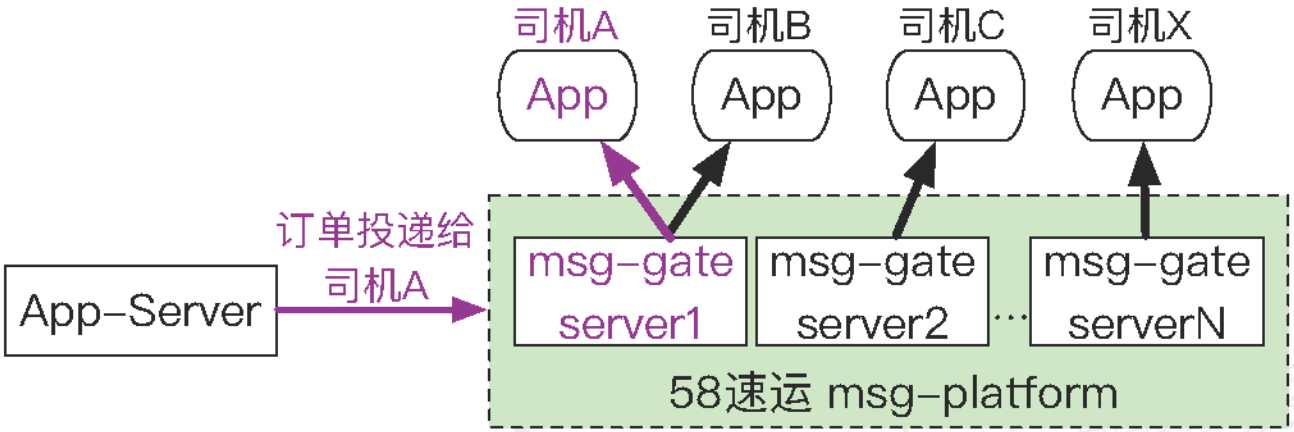
高性能

- 改用tcp长连接



- 潜在问题

  实时订单推送　长连接状态维护

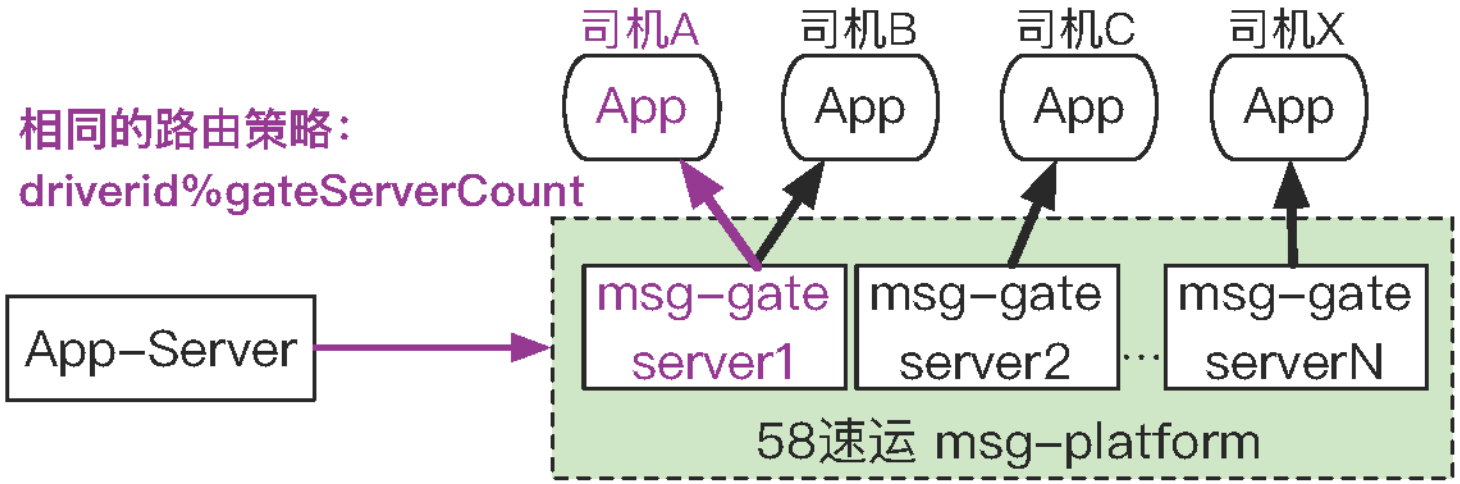


高性能 - 长连接状态维护

- 实时订单推送　长连接状态维护

方案：

1、App 和 App-Server使用相同的路由策略；



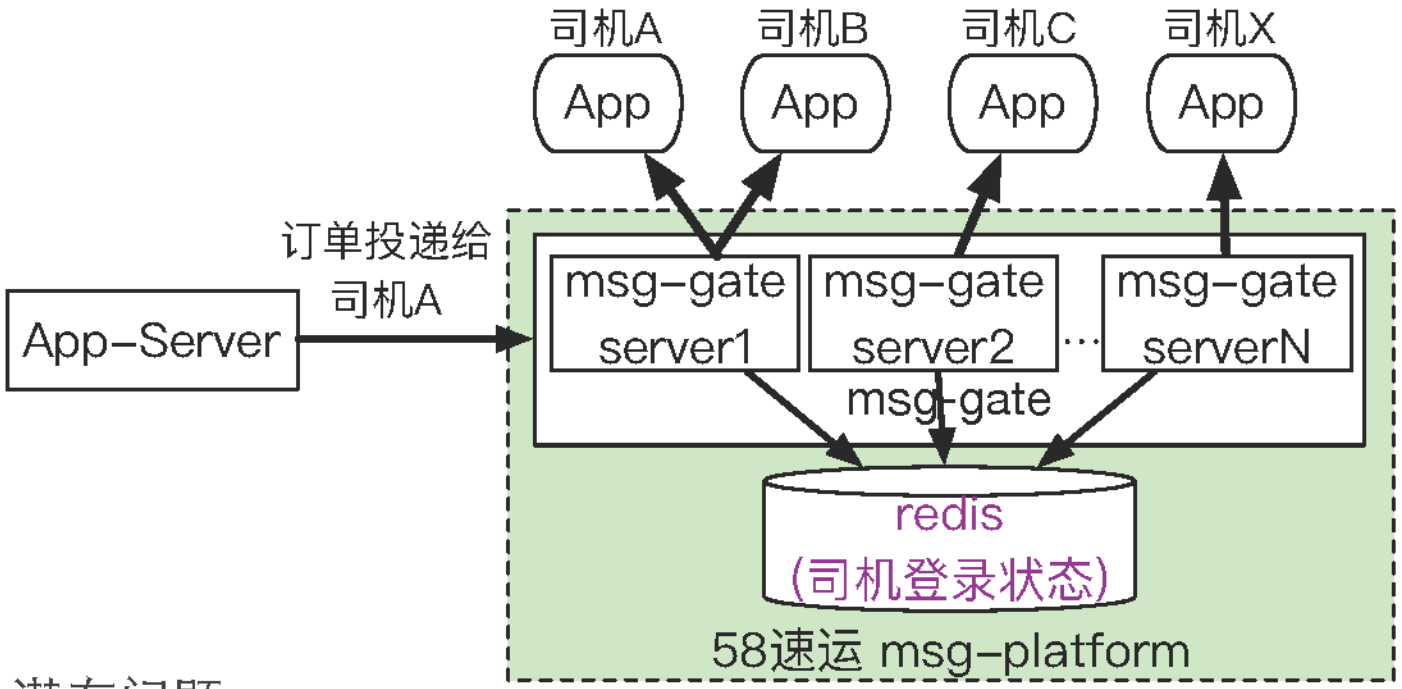
- 潜在问题

msg-gate-server 扩、缩容时，需要通知App端；

路由策略更新时，需要通知App端；

高性能 - 长连接状态存储(58速运实践)

- 司机登录状态存储 <driver_id，{msg-gate ip:port}>
  优点：msg-gate服务无状态



- 潜在问题：

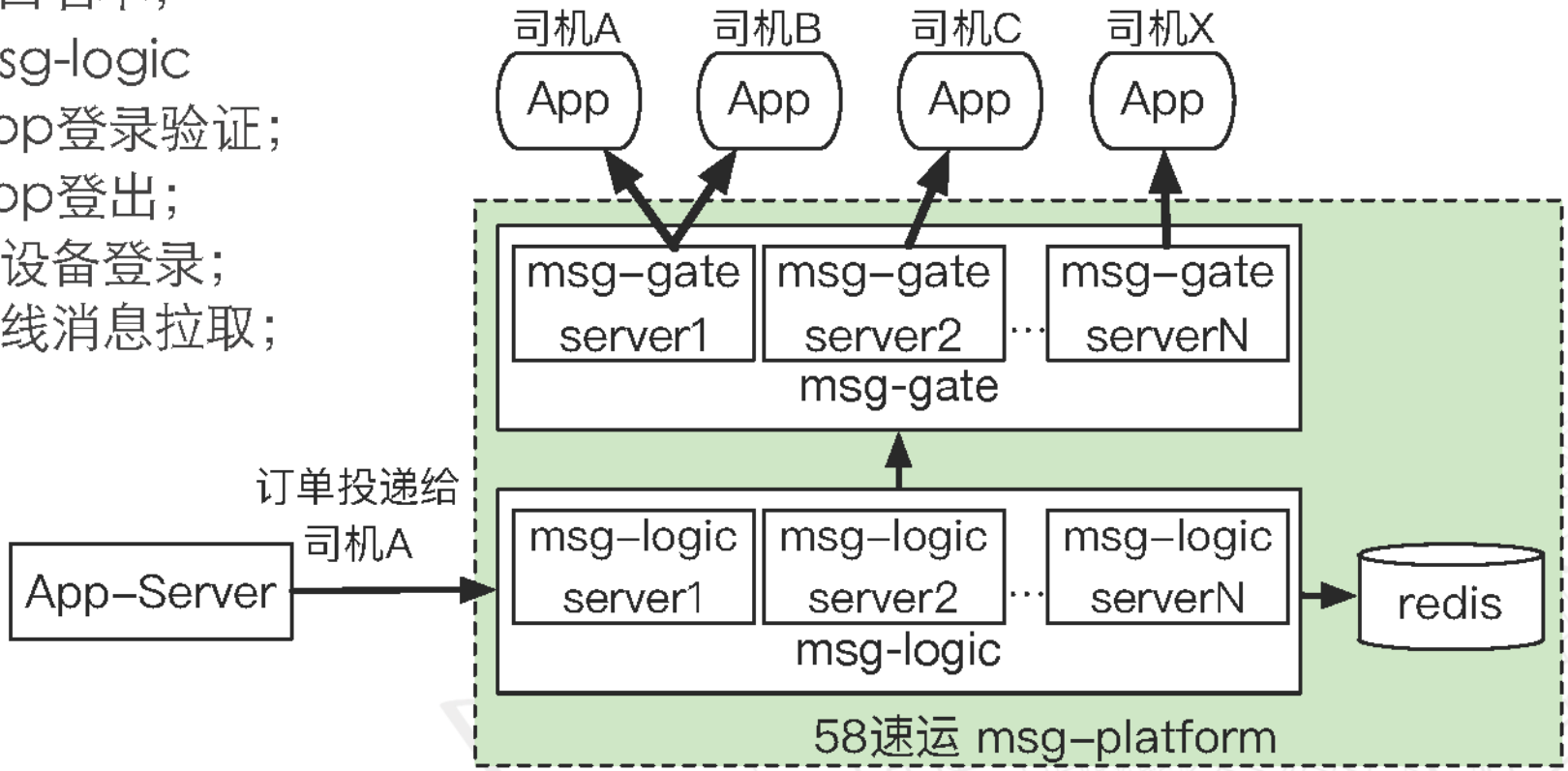
msg-gate多层职责

维持长连接、路由、相关业务(司机登录、登出等)

业务更新相对较频繁，需要msg-gate重启，会导致长连接全部重连

分层架构

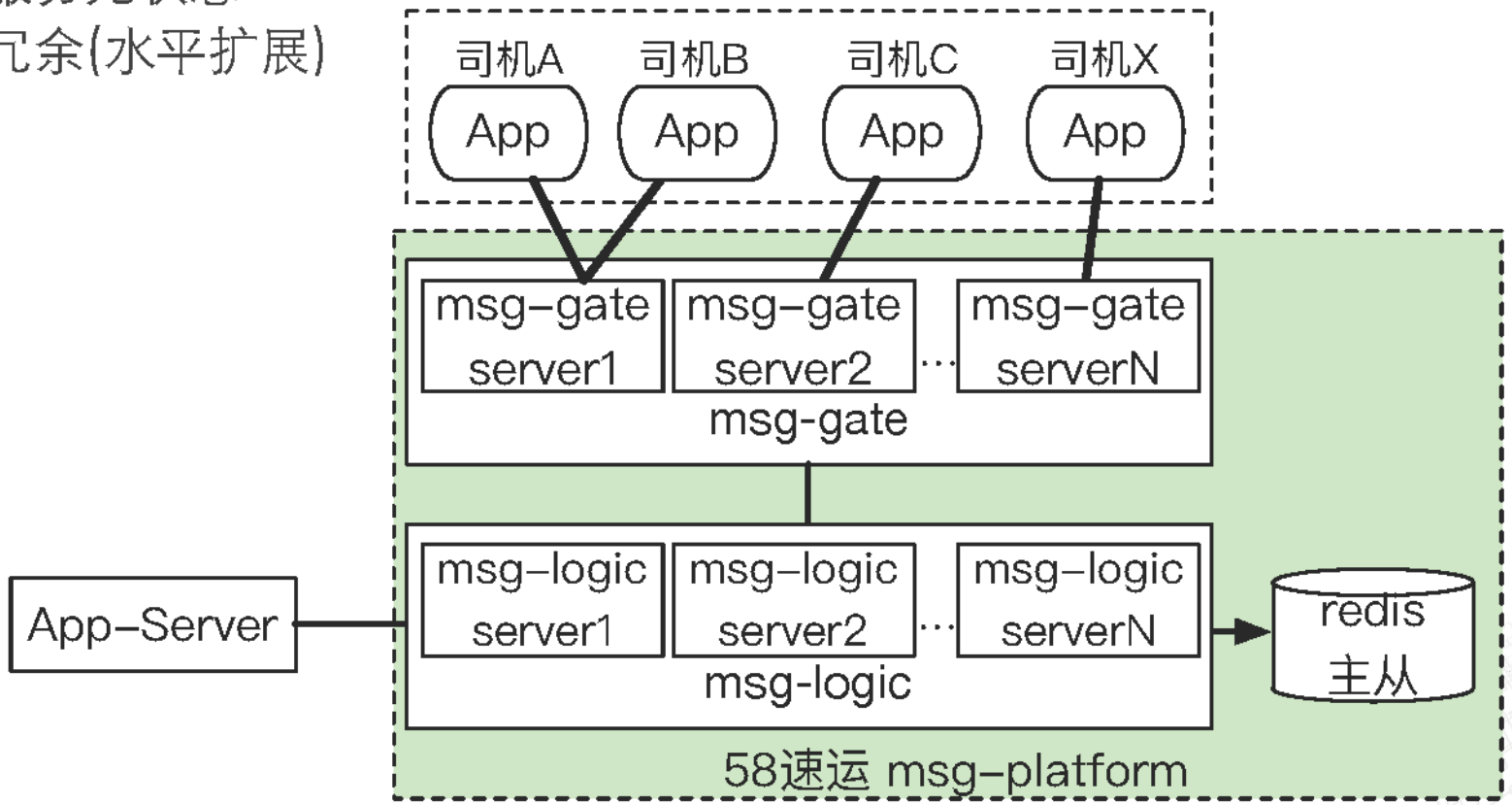
- msg-gate
  百万连接管理；
  流控；
  黑白名单；
- msg-logic
  App登录验证；
  App登出；
  单设备登录；
  离线消息拉取；



高可用
负载均衡

高可用

- 服务无状态
- 冗余(水平扩展)



msg-logic、msg-gate 负载均衡怎么做?

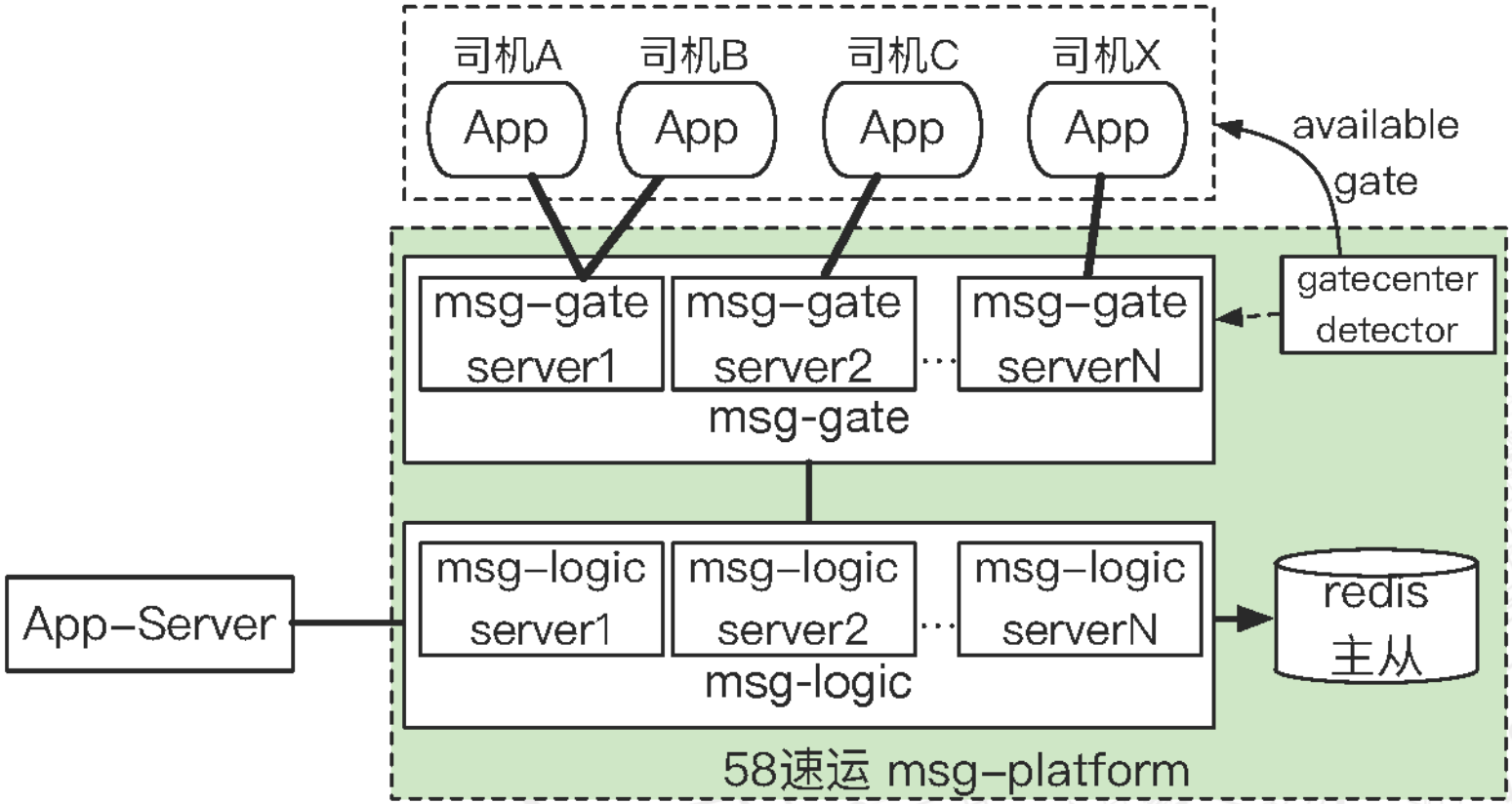
负载均衡

- 实时推送订单给司机

App-Server[客户端] -> msg-logic； msg-logic[路由] -> msg-gate；

- App端上报GPS到App-Server

App->msg-gate[服务端]； msg-gate[客户端]->msg-logic；

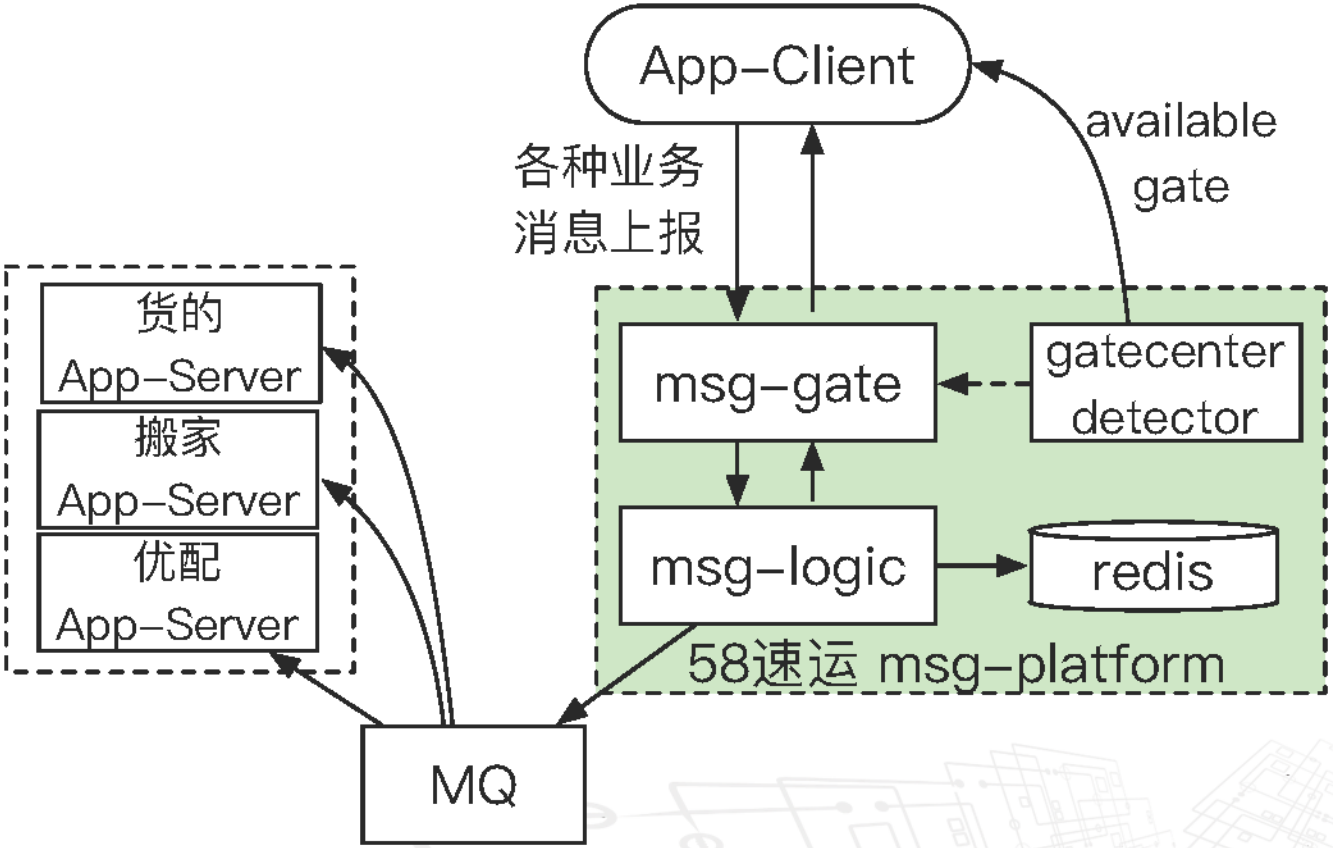


高可扩展性

业务扩展

- 通用消息平台

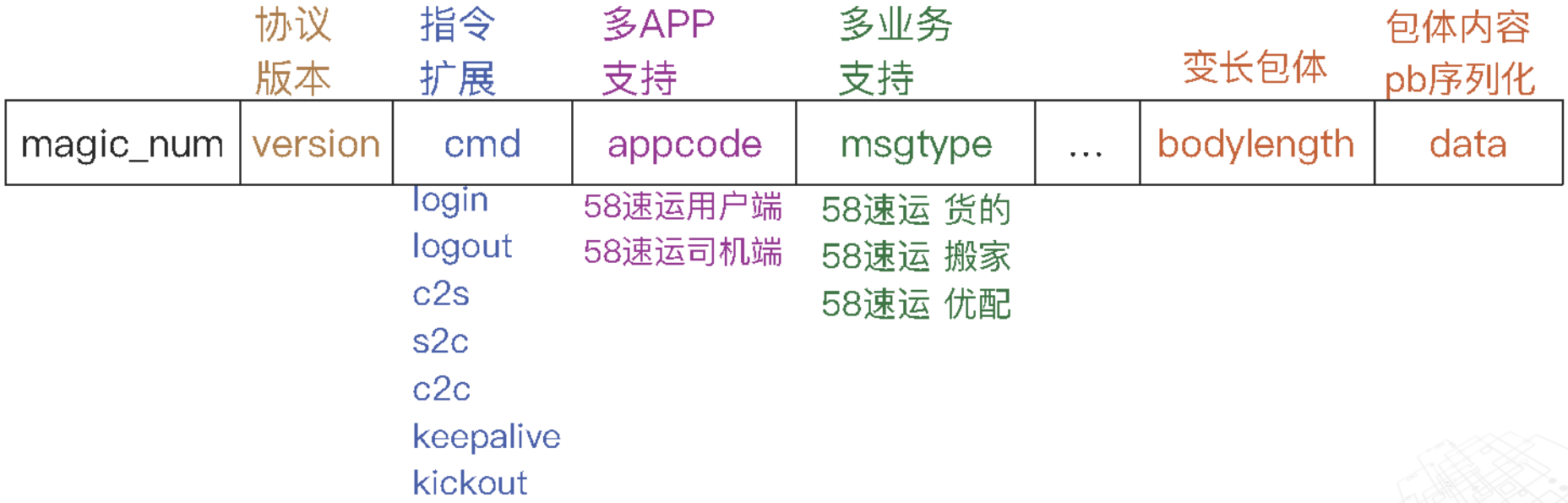
消息平台不关注具体业务



协议扩展

- 定长包头 + cmd + 变长包体

消息平台不关注具体业务



58速运消息平台整体架构

- 分层设计

msg-Gate:连接整流、session管理、攻防
msg-Logic:登录、登出、离线消息拉取(IM)

- 高性能

tcp长连接、redis实时状态存储、异步化

- 高可用

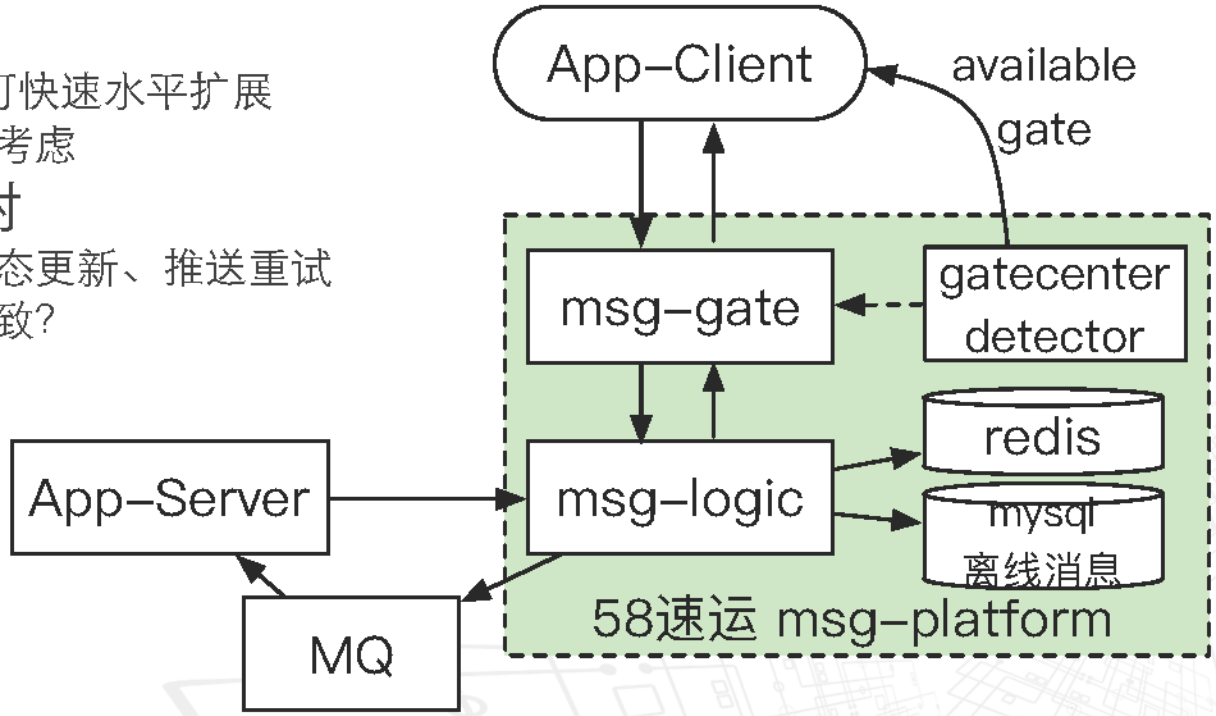
服务无状态+冗余，可快速水平扩展
每个模块都有高可用考虑

- 高到达率、实时

长连接保持、实时状态更新、推送重试
什么时候状态会不一致?

- 高可扩展

通用消息平台



核心业务流程

login(登录)

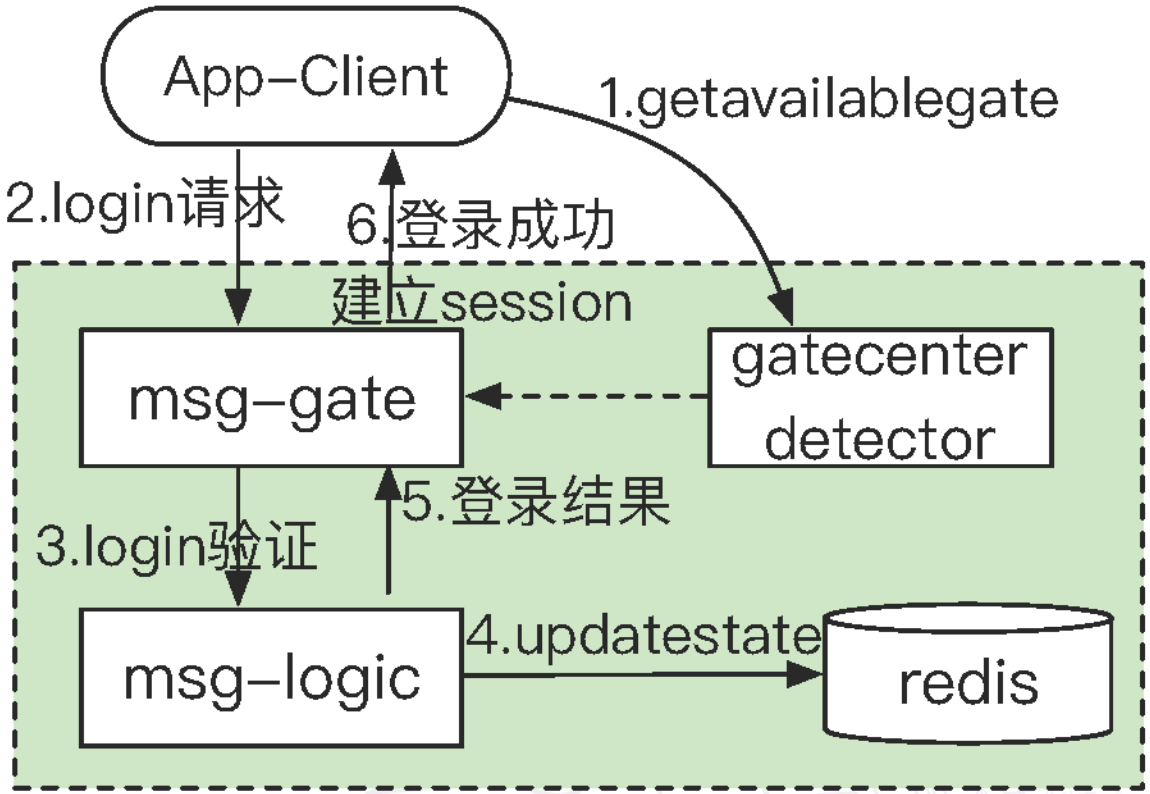
c2s(appClient To appServer)

s2c(appServer To appClient)

c2c(appClient To appClient)

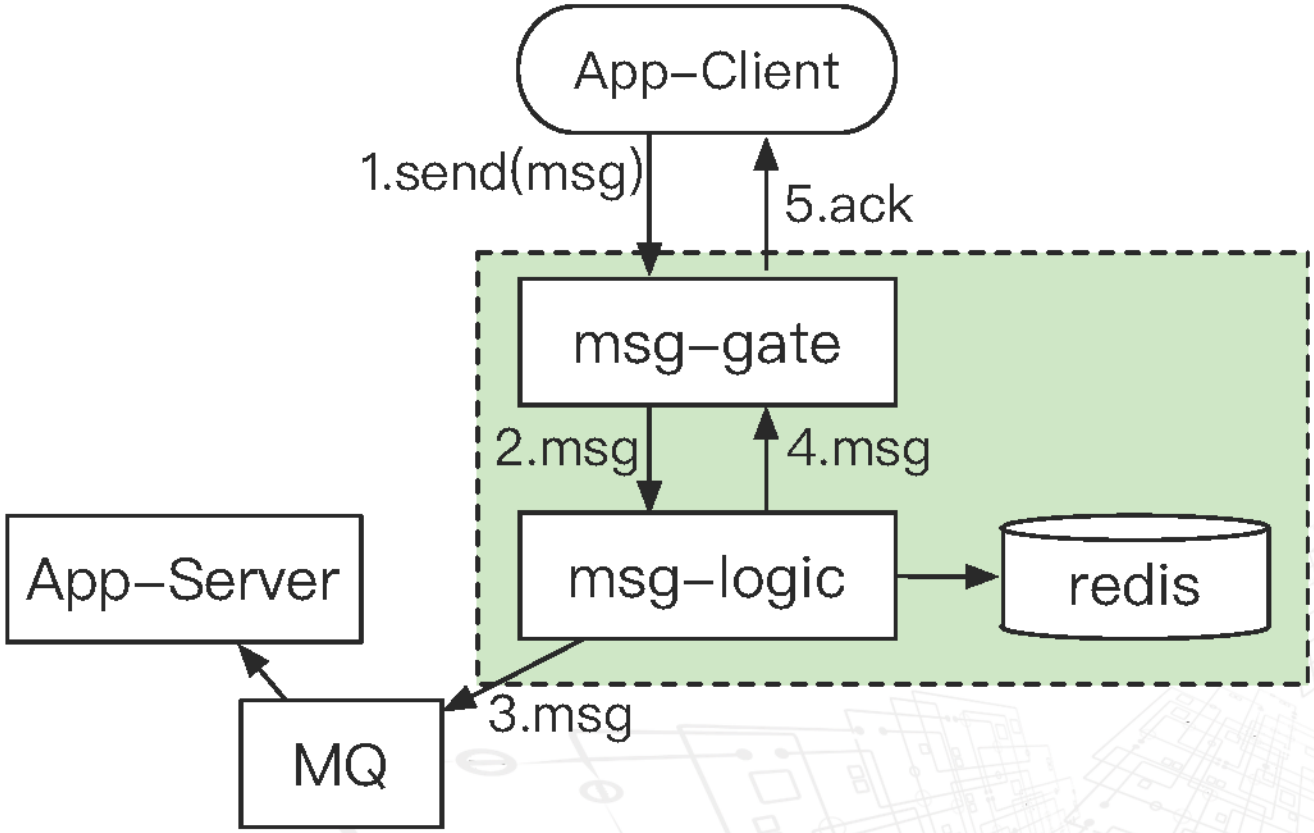
核心业务流程 - login

- AppClient登录消息平台，登录成功后，建立连接session
- gate-center-detector负载均衡



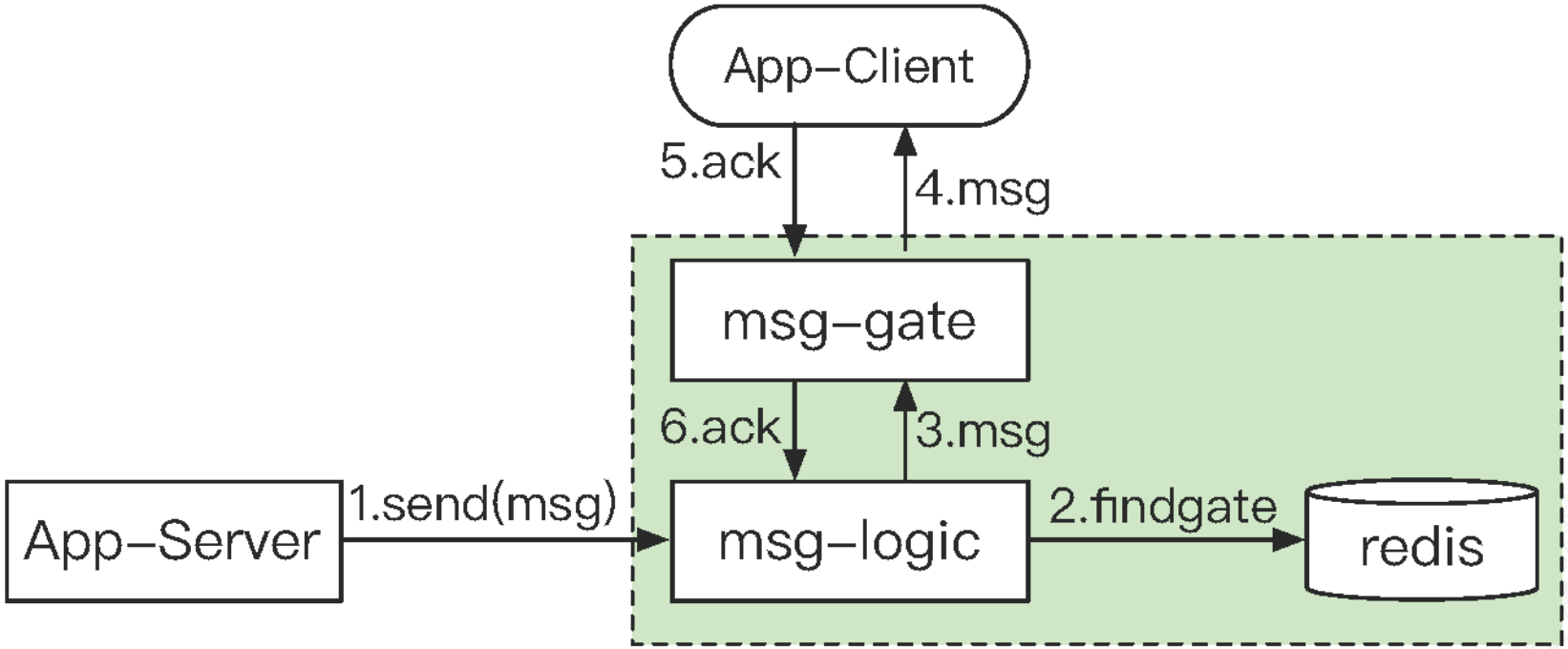
核心业务流程 - C2S

- appClient To appServer
  消息平台与AppServer业务消息解耦
  消息平台不关注具体消息内容



核心业务流程 - S2C消息

- appServer To appClient
  找到msg-Gate进行实时推送
  消息到达率统计



核心业务流程 - C2C消息

- appClient To appClient
  移动端网络环境不稳定下，优化消息发送方体验
  服务端代发消息接收方的ACK消息

- 流程
  1: 发送方发送IM消息;
  2、3: 消息先落地存储;
  4、5: 服务端代发ACK消息;
  6: 查询消息接收方是否在线;
  7、8: 如果在线，则实时推送;
  9: 接收方ack确认收到消息;
  10、11: 服务端从数据库中删除消息;
  12、13: 服务端回复接收方ack;
  (避免step 9重复ack)