

1230666

购票系统

架构设计报告

BY 王梦麟
03/30/2017



功能需求

- 车票操作：查询余票；车票购买、改签、退票
- 账户管理：查询订单；增删常用乘车人
- 列车时刻：提供列车运行时刻表、列车正晚点信息
- 外部服务：通过公开接口对第三方系统提供服务

质量属性与约束

- 质量属性
- 性能、安全性、可用性
- 可延展性、互操作性、易用性
- 约束：
- 感觉这一页PPT没卵用
- 感觉下面的场景画图更好看？

ASR-性能

场景组成部分	可能的值
源	用户，第三方系统
刺激	非周期性的数据请求，操作请求
制品	Broker 模块，服务器模块，request bus
环境	系统正常运行时，系统高负载时
响应	系统执行用户请求的操作；处理用户数据；返回数据
响应度量	系统在满负载运行时，请求响应延迟时间不超过 5s 任何情况下，请求丢失的比例不超过 0.001%

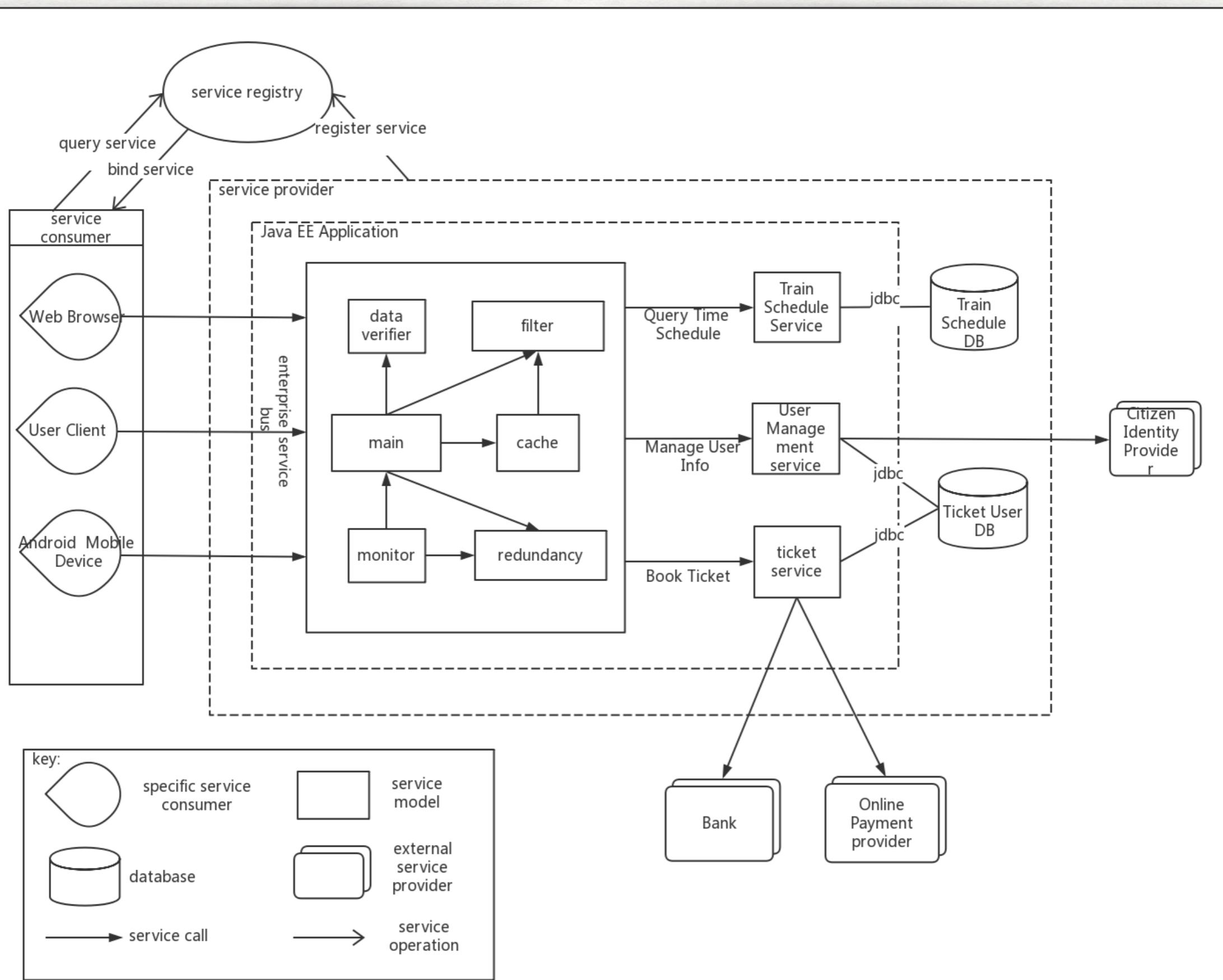
ASR-安全性

场景组成部分	可能的值
源	对系统的攻击行为
刺激	黑客对系统的数据攻击、DOS 攻击等
制品	系统
环境	系统运行时
响应	系统检测到攻击，同时采取相应保护措施， 避免攻击范围扩大，消除攻击带来的影响
响应度量	在 1s 之内检测到正在进行的攻击。在 10s 内采取安全措施。 在 60s 内消除攻击带来的影响。

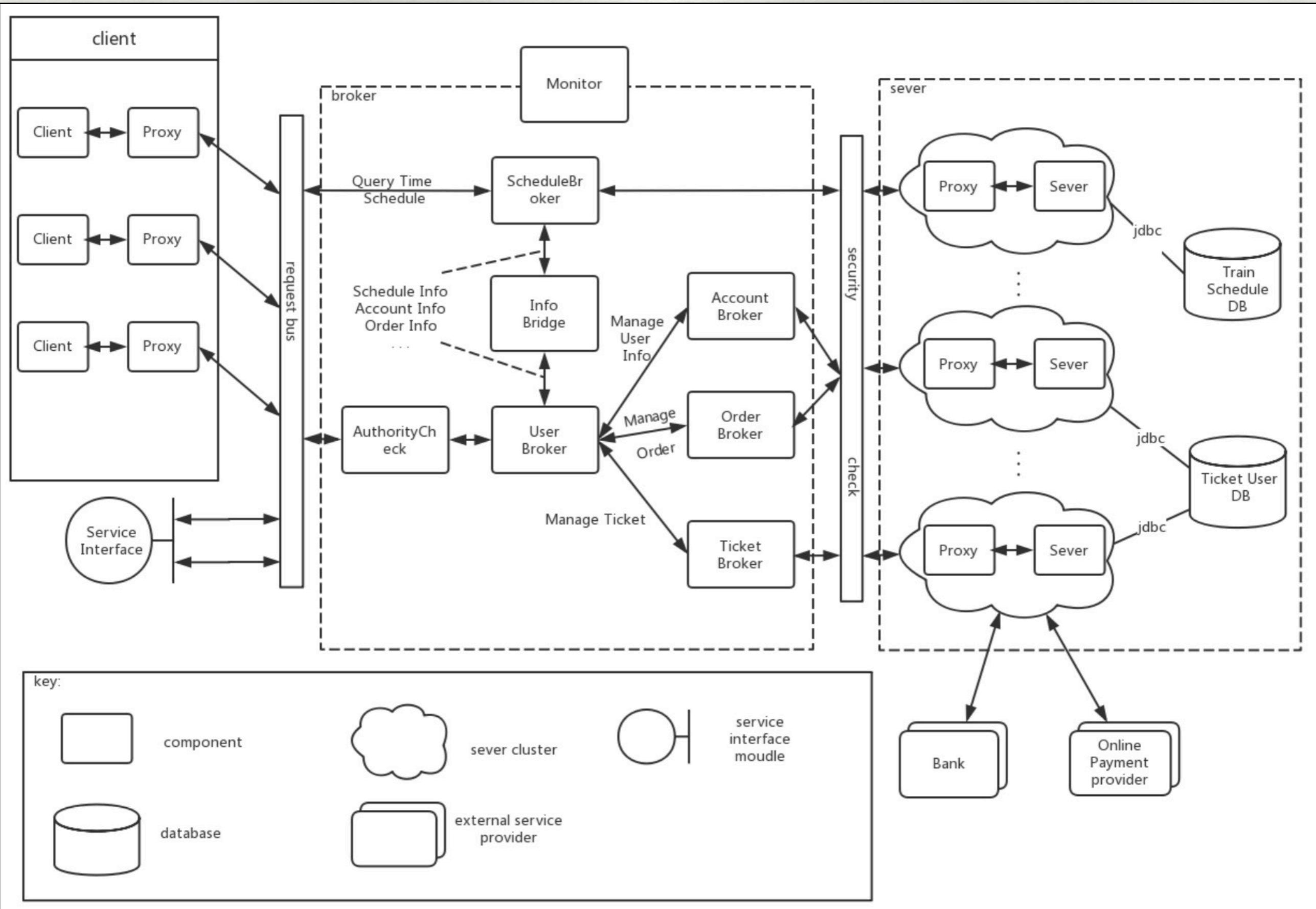
ASR-可用性

场景组成部分	可能的值
源	系统内部的运行或者是用户的操作
刺激	大量用户的操作导致系统压力增大或者是系统自身运行时出现错误
制品	系统
环境	系统运行时
响应	如果压力过大则将压力转移到一些设备上。 如果是出错，那么快速修复错误。
响应度量	诊断异常状况的时间不超过 2s；将请求压力转移的时间不超过 3s； 快速修复错误的时间不超过 2s；错误修复成功率不低于 98%

候选架构1——SOA



候选架构2——BROKER



架构对比

- 相同
 - 都有消息中间件(BROKER, ESB)，提供安全等多方面服务
 - 都具有较高的互操作性(BROKER的代理, SOA的注册)
 - 都适合异步和分布式应用，可扩展性较好
- 不同
 - 架构侧重点不同：
 - 性能方面，SOA无法保障，BROKER有灵活的解决方案
 - 可用性方面，SOA各个服务模块更加分散，没有易失效的单点，经过设计BROKER也可以达到较高的可用性

架构选择决策——BROKER

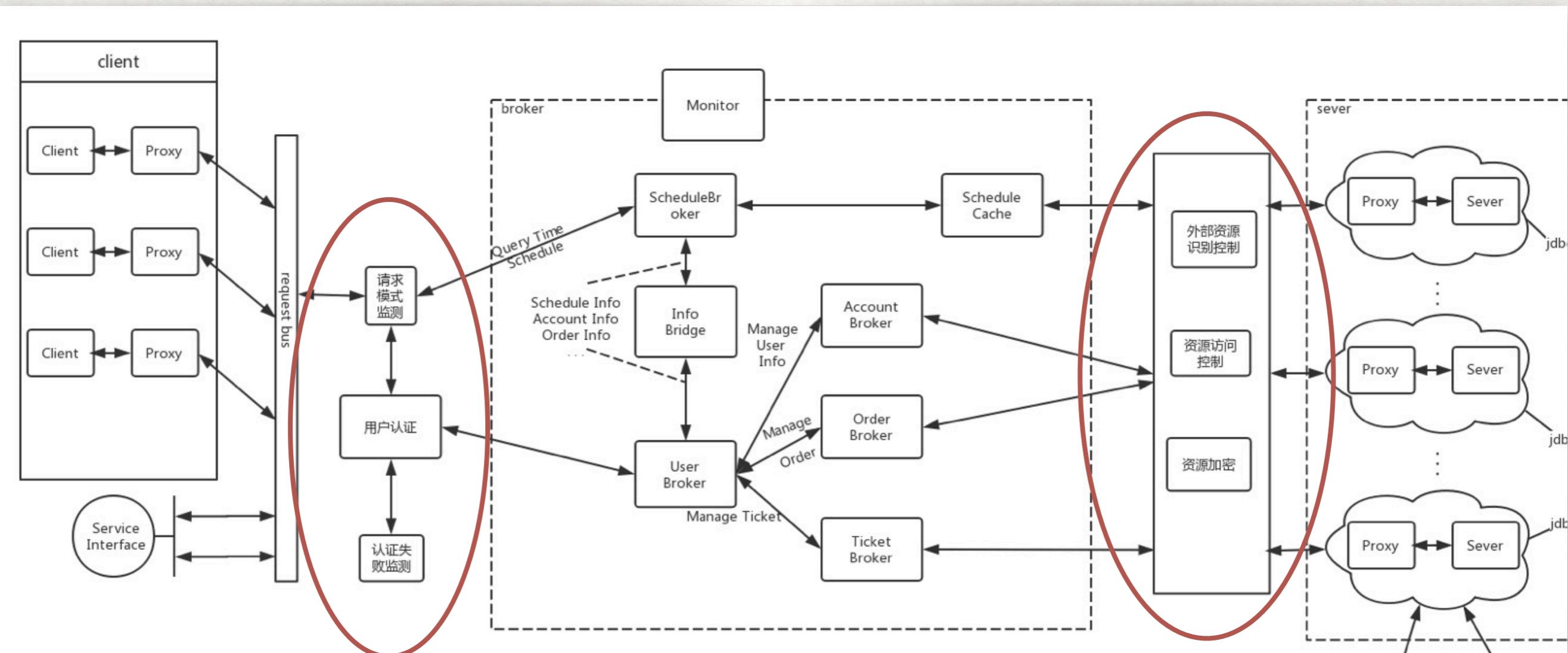
- 舍弃SOA架构原因
- 系统业务相对稳定， SOA强调服务之间的低耦合，对扩展性、灵活性要求高的系统更有价值
- SOA无法保证高性能， 服务的实现依赖其他服务和中间件

- 选择BROKER架构原因
- BROKER使用事件驱动架构， 可以异步处理购票操作， 适合应对大流量的数据流，在这个车票业务上性能更好
- 虽然BROKER存在单点问题， 但是经过设计， 可以通过集群方式来缓解BROKER的压力， 提升可用性

ADD迭代3——系统安全模块

- 选择ASR: 安全性
- 使用策略: 检测请求模式、检测服务拒绝、用户认证与授权

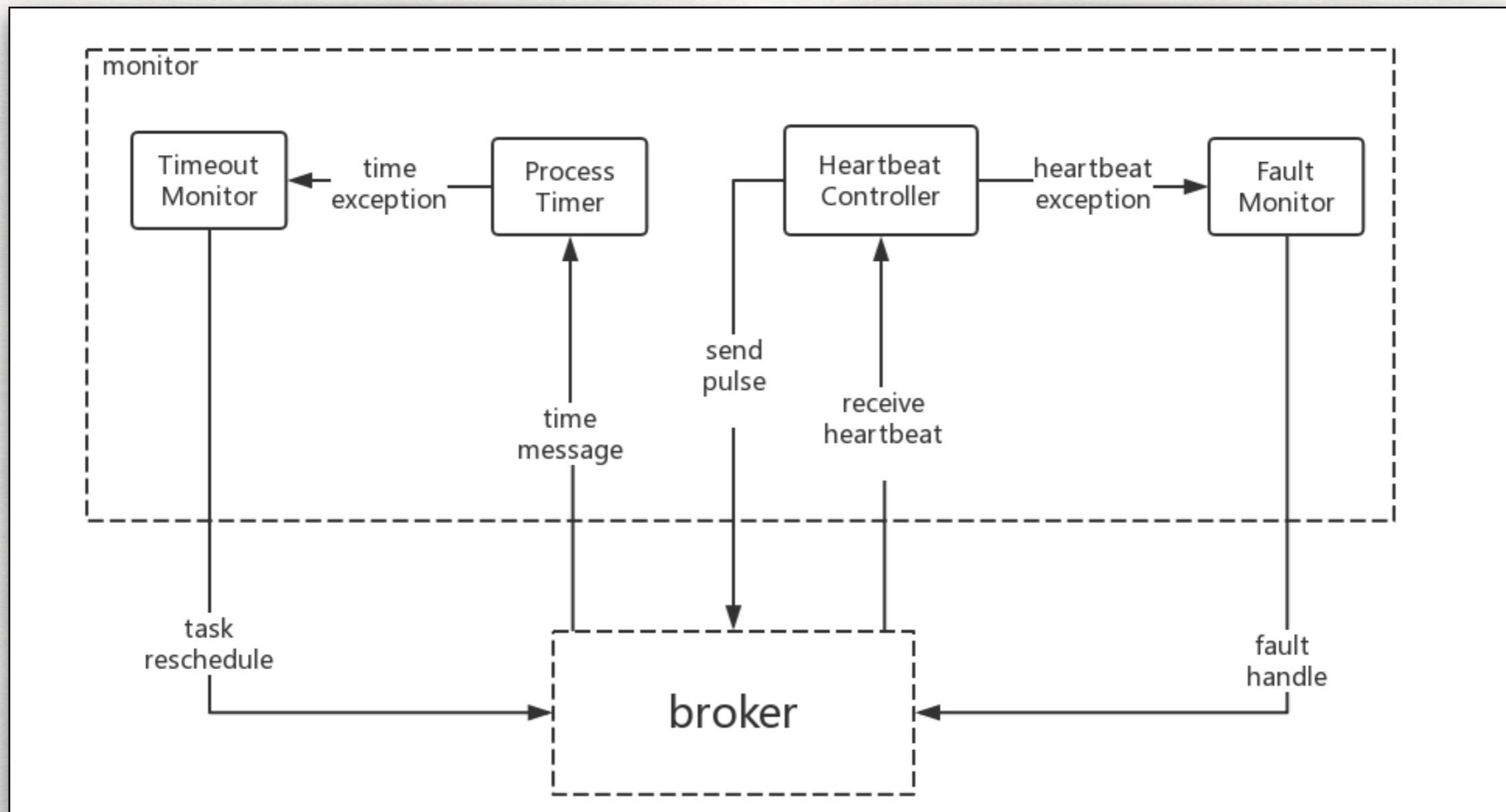
资源访问限制、信息加密



ADD迭代4——BROKER检测模块

- 选择ASR：可用性、性能
- 使用策略：Heartbeat、消极冗余（图里面没有体现）

限制操作时间、引入并行（图里面也没有体现）



ADD迭代5——服务器模块

- 选择ASR:

- 1 安全性 2 可用性

- 3 互操作性 4 易修改性

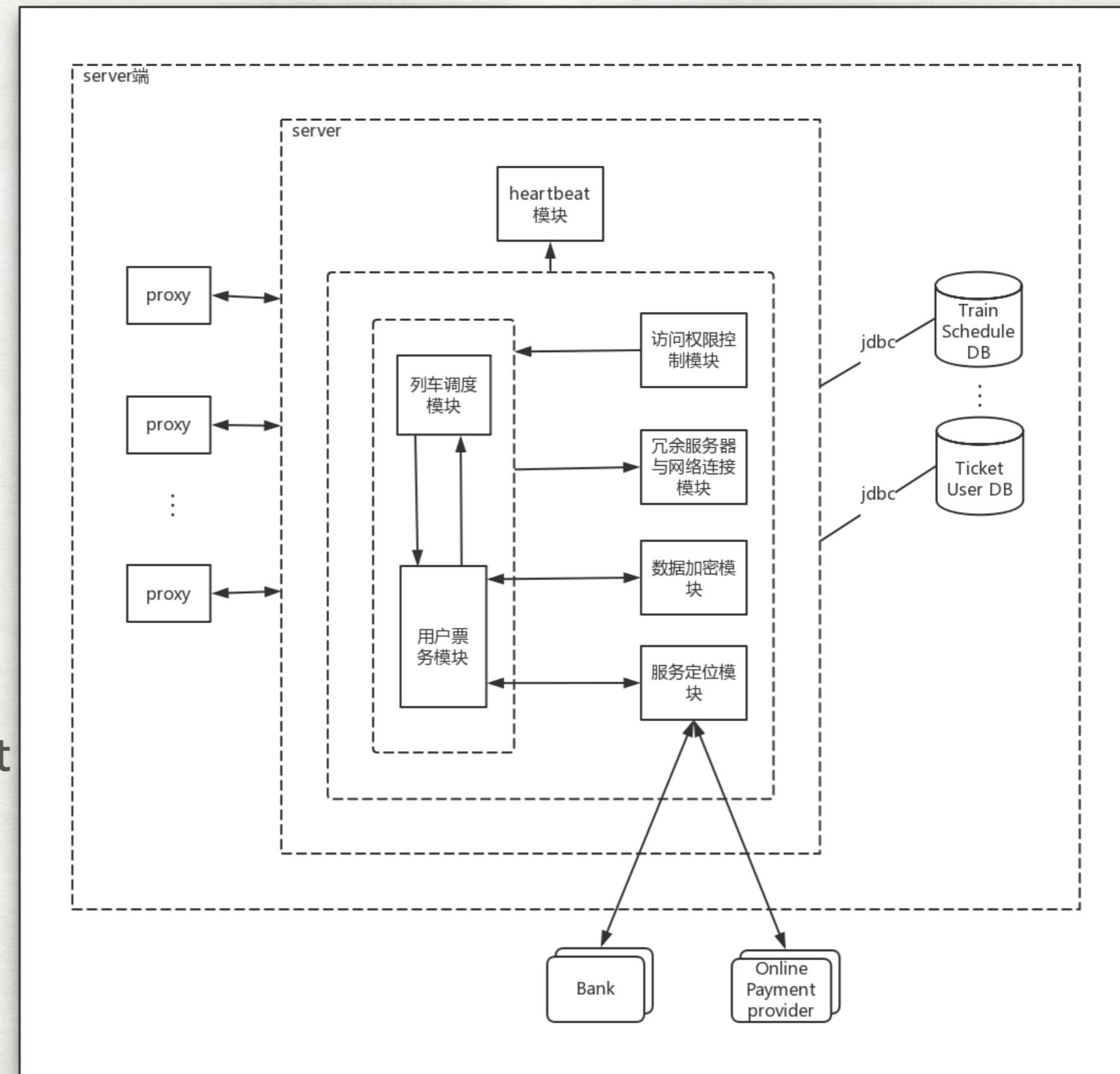
- 使用策略:

- 1 攻击侦测、数据加密、限制访问

- 2 主动冗余、heartbeat

- 3 服务定位

- 4 模块分解



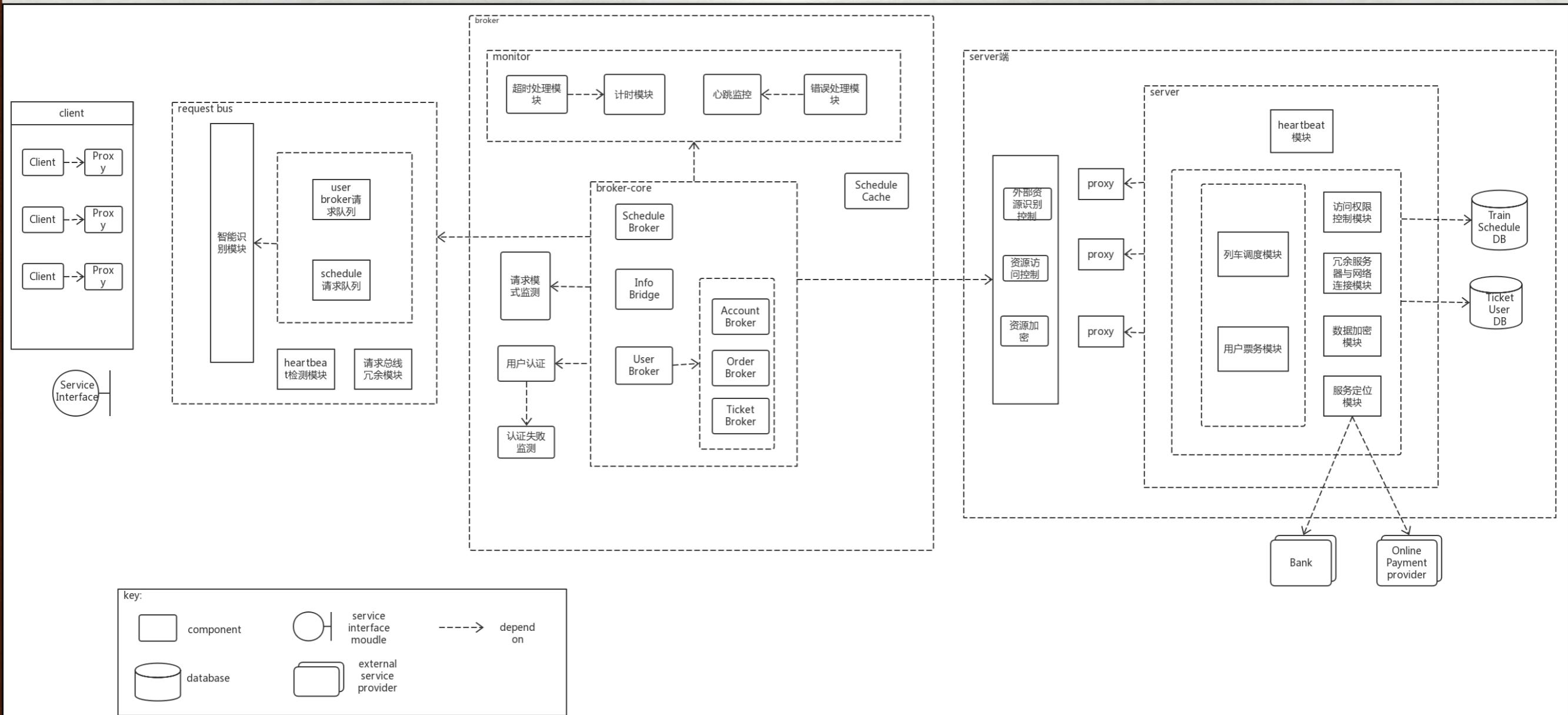
ADD迭代6——BROKER请求总线

- 选择ASR：性能、可用性
- 使用策略：消息队列、异步处理、智能分派；冗余、heartbeat

ADD迭代7——数据库模块

- 选择ASR：性能
- 使用策略：读写分离：买票消耗小，查票更加占用资源。
- 查票（读互联网内存库）和买票（写铁路内网）分开

BROKER—MODULE视图OR CC 反正一个图



THE END