购票模块

Reliability，Interopreability

**步骤一：**

该迭代不需要此步骤

**步骤二：**

该迭代针对购票模块进行分解

**步骤三：**

确定框架驱动元素

|  |  |  |  |
| --- | --- | --- | --- |
| # | 架构驱动元素 | 重要性 | 难易度 |
| 1 | 场景1：阻止恶意抢票 | 高 | 中 |
| 2 | 场景4：和第三方交互 | 中 | 高 |
| 3 | 场景6：高并发的客户流量 | 高 | 高 |
| 4 | 场景9：系统不可以崩溃、无响应 | 高 | 高 |
| 5 | 场景10：简单操作 | 高 | 低 |
| 6 | 场景11：改签退票操作 | 高 | 低 |

**步骤四：**

选择满足架构驱动因素的架构模式。

1. **针对可靠性（Reliability）**
2. **确定设计关注点**

|  |  |
| --- | --- |
| 设计关注点 | 子关注点 |
| 检测可能出现的故障 | 状态检测 |
| 防止发生故障 | 数据备份 |
| 业务高并发处理 |

1. **备选方案以及选择方案**
2. **状态检测**

·用于区分的参数：

- 每次消息传输量

|  |  |  |
| --- | --- | --- |
| # | 模式名 | 消息传输量 |
| 1 | 心跳模式 | 1n |
| 2 | Ping/Echo模式 | 2n |

注：n为需要探测的组件数

* 选择方案：心跳模式
* 选择原因：心跳模式可以及时得到结果，且用于传输消息的消耗较小。同时心跳模式的实现也更为简洁，反馈更有规律性。

1. **数据备份**

·用于区分的参数：

- 数据传输负担

- 数据实时性

|  |  |  |  |
| --- | --- | --- | --- |
| # | 模式名 | 数据传输负担 | 数据实时性 |
| 1 | 同步更新备份数据 | 低 | 高 |
| 2 | 定时更新备份数据 | 高 | 低 |

* 选择方案：同步更新备份数据
* 选择原因：同步更新备份数据与定时更新备份数据相比，更能保证备份数据的实时性。当需要使用备份数据时，同步更新的模式能提供最新的数据。同时，定时更新备份数据可能会因为积累了大量未更新数据而造成数据需要传输的量过大的情况，增加传输负担，影响正常数据的传输。

1. **业务高并发处理**

·用于区分的参数：

- 成本

- 高并发处理能力

|  |  |  |  |
| --- | --- | --- | --- |
| # | 模式名 | 成本 | 高并发处理能力 |
| 1 | 对不同进行业务分布式部署 | 低 | 中 |
| 2 | 增加服务器数量 | 高 | 中 |
| 3 | 分布式部署+增加服务器 | 高 | 高 |

* 选择方案：分布式部署+增加服务器
* 选择原因：在分布式部署不同服务的前提下，增加服务器的数量可以进一步提高业务的高并发处理性能。同时，将不同服务部署到不同的服务器也有助于防止一台服务器崩溃而影响别的服务的情况发生。

1. **针对互操作性（Interoperability）**
2. **确定设计关注点**

|  |  |
| --- | --- |
| 设计关注点 | 子关注点 |
| 定位外部服务 | 查询外部服务 |
| 管理接口 | 管理对外接口 |

1. **备选方案以及选择方案**
2. 查**询外部服务**

·用于区分的参数：

- 灵活性

- 查询效率

|  |  |  |  |
| --- | --- | --- | --- |
| # | 模式名 | 灵活性 | 查询效率 |
| 1 | 注册中心服务查询 | 高 | 低 |
| 2 | 服务绑定 | 低 | 高 |

* 选择方案：注册中心服务查询
* 选择原因：通过向服务注册中心查询相关服务，可以灵活地调用需要的服务，符合SOA的基本思想。绑定服务虽然在有较高的定位服务的效率，但是难以扩展，不利于软件的架构。

1. **管理对外接口**

·用于区分的参数：

- 是否便于维护

- 复杂性

|  |  |  |  |
| --- | --- | --- | --- |
| # | 模式名 | 是否便于维护 | 复杂性 |
| 1 | 定制接口 | 否 | 高 |
| 2 | 组织协调接口调用 | 是 | 低 |

* 选择方案：组织协调接口调用
* 选择原因：组织协调接口调用通过对不同接口进行协调排序，使接口之间分工合作完成功能。而定制接口则针对一个接口进行功能的增加或删除。所以定制接口会增加代码的复杂度，导致系统不利于维护。而协调接口调用则有利于系统的修改与维护。