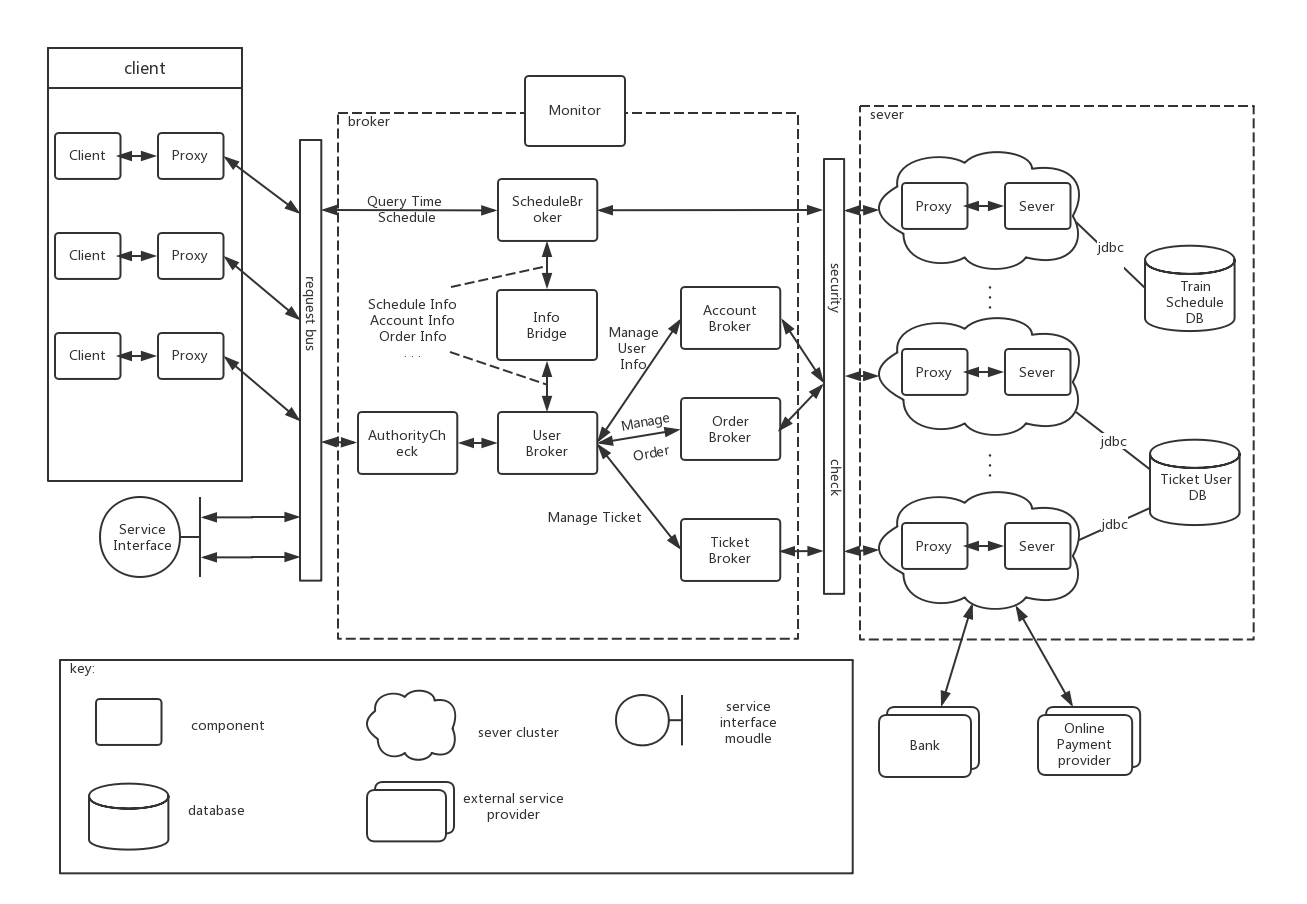
**Broker架构设计和评估ADD过程**

## 第一次迭代

分解系统如图所示。主要系统模块有client模块，broker模块，server模块，监控模块，安全模块。其主要职能如下:

1. broker模块：有定位服务器，向合适的服务器转发请求，向客户端返回结果或异常。UserBroker组件负责有关用户的请求转发处理，该broker向AccountBroker（处理账户信息）、OrderBroker（处理订单信息）、TicketBroker（处理票务请求）三个二级broker分送请求。
2. server模块：后端数据逻辑处理，服务提供等
3. 监控模块：作为broker模块的监控组件，职能是侦测模块中可能出现的错误。错误产生时，Monitor可以采取有效措施来消除错误、规避风险、提交报告。
4. 安全模块：面对来自外部的恶意攻击和刷票行为，安全模块需要过滤信息，拦截无效或者恶意请求，保证其他组件输入输出数据合法。



## 第二次迭代

#### 选择元素

选择的元素是user-broker模块，负责用户账户、订单和购票请求的转发分派。

#### 选择ASR

第二次迭代选择的ASR是性能、可用性。

对于该系统，由于该系统的用户基数较大且流量高峰时间比较集中，就要求系统在正常情况下和高并发模式下能做出正确而快速的响应，另外由于broker在系统中的枢纽地位，一旦出错当机，对系统的影响是难以估量的，因此系统的可用性和性能显得尤为重要。

#### 候选策略表和决策

#### 可用性

|  |  |  |  |
| --- | --- | --- | --- |
| 策略 | 优点 | 缺点 | 决策 |
| 主动冗余 | 通过对重要broker的备份，可以在主broker出现问题时切换到备份的broker。增强了系统的可用性和安全性；另外由于broker架构的灵活性，可以比较方便地切换主从broker，回复速度较快 | 冗余的broker增加了系统的负担，当系统切换broker时不可避免地会增加系统的负载压力和复杂度，耗费资源 | 不采用，耗费资源太多 |
| 被动冗余 | 通过从broker中的某些信息来恢复主broker。增强了系统的可用性，两个broker各自有不同的功能，功能不完全冗余，系统的负担较小 | 主broker和从broker之间的信息同步可能对系统造成一定的压力，而且错误恢复的速度比较慢 | 采用，可以对于较重要的或者有前后顺序逻辑的broker使用被动冗余来加强可用性 |
| 移除可疑broker | 错误恢复后将发生错误的broker移除，可以防范broker再次出错 | 某些broker的功能是无法替代的，移除后可能造成系统瘫痪 | 不采用 |

#### 性能

|  |  |  |  |
| --- | --- | --- | --- |
| 策略 | 优点 | 缺点 | 决策 |
| 模块细化 | User-broker可以将自己的功能分解成几个较独立的功能模块，分派给二级broker。User-broker负载压力减小，能够更好更快地处理高并发问题 | Broker之间的通信更加复杂，增加了系统的复杂度 | 采用 |
| 优先级调度 | User-broker通过识别优先级对优先级更高的请求进行优先转发，可以保证系统对于重要功能的快速响应 | 请求优先级设置问题，难以决定哪些请求更重要。另外优先级调度也会增加系统的复杂度 | 不采用，对于本系统而言用户的请求并没有很明显的优先级 |
| 建立缓存模块 | 通过把请求结果缓存到缓存模块中，对于重复请求可以直接响应而不用向server进行请求，解决一部分并发问题，可以减少等待时间，提高系统性能 | 需要对用户请求结果进行缓存，可能造成一定的负载压力，另外某些实时变化的信息并不适合缓存 | 不采用，因为不会有多个用户对user-broker中的信息同时进行访问（账户信息、订单信息、票务信息），没有缓存的必要 |

#### 第二次迭代结果

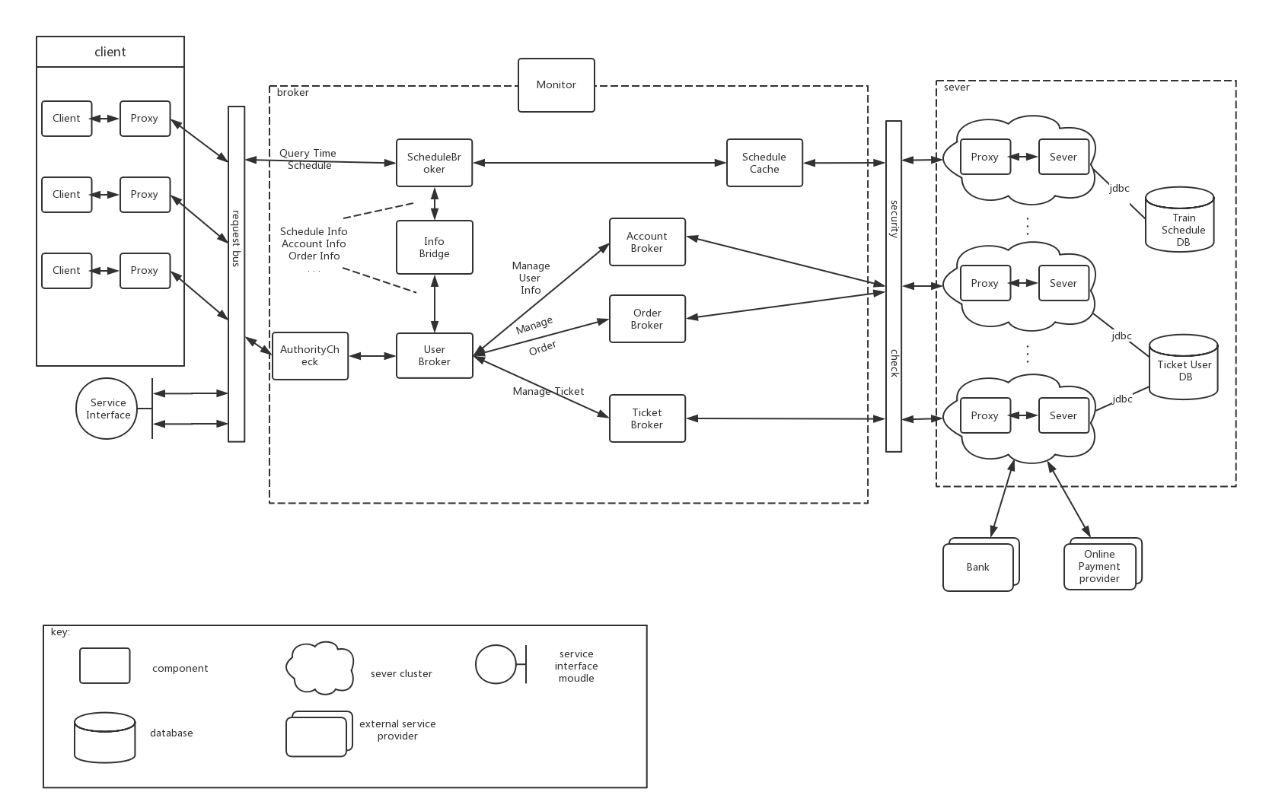
如图所示，新增元素说明如下

account-broker模块：负责user-broker中的账户请求服务的二级转发

order-broker模块：负责user-broker中的订单请求服务的二级转发

ticket-broker模块：负责user-broker中的票务操作请求服务的二级转发

schedule-cache模块：负责对列车调度表进行缓存



## 第三次迭代

#### 选择元素

第三次迭代选择的元素是安全模块，包括broker部分的authorityCheck，位于broker和server之间的security Check。

#### 选择ASR

第二次迭代选择的ASR是安全性。这个模块主要负责系统的安全控制。

#### 候选策略表和决策

#### 安全性

|  |  |  |  |
| --- | --- | --- | --- |
| 策略 | 优点 | 缺点 | 是否采用 |
| 检测请求模式 | 通过检查正常服务请求的模式和机器指令，来控制访问权限。避免抢票插件不正常的操作数据。避免刷票程序占据带宽，增强性能。 | 增加系统复杂度，需要专门的模块来记录请求模式，比较请求模式。可能增加每一条请求的时间，降低性能。 | 采用 |
| 检测服务拒绝情况 | 通过检查同一身份认证多次失败的情况，能够防止恶意的测试用户密码。 | 同上 | 采用 |
| 识别信息一致性 | 通过检验hash值或者checksum来确定数据一致性。能够避免用户数据直接传输或者存储，在数据泄露时造成重大损失。 | 增加信息传输总量，占有带宽，降低性能。 | 采用 |
| 识别角色 | 对外部输入系统的资源加以识别，防止脚本注入攻击等。 | 增加系统复杂度。 | 采用 |
| 用户认证 | 通过识别用户身份，避免不当的数据操作，其他用户修改了某个用户的个人信息。 | 需要与其他组件交互，存储用户数据，增加系统复杂度。 | 采用 |
| 用户授权 | 通过给用户授予合适的权限，限制用户的能力，避免用户对重要数据的不当操作。 | 无 | 采用 |
| 限制对计算机资源的访问 | 通过限制直接访问计算机资源，避免暴露系统内部细节。减少了黑客攻击的可能。 | 无 | 采用 |
| 最小化系统的攻击面 | 通过减少系统访问面，要求所有的请求必须经过安全处理模块，能够集中处理所有的攻击。 | 因为所有的请求都必须经过安全检查部分，所以该组件一旦遭到攻击失效，会影响系统的可用性安全性。 | 采用 |
| 当攻击发生时删除敏感的资源 | 系统已经遭到潜在的攻击，与其使重要数据被窃取，不如直接销毁，减小损失的影响。 | 对于不是特别敏感的资源，删除导致的损失过大 | 不采用，该系统没有敏感资源 |

#### 第三次迭代结果

如图所示，新增元素说明如下。

请求拒绝监测模块：在请求验证失败后，记录失败信息。再次请求时检查验证失败记录，做出判断并且拒绝请求。避免恶意尝试破解用户密码

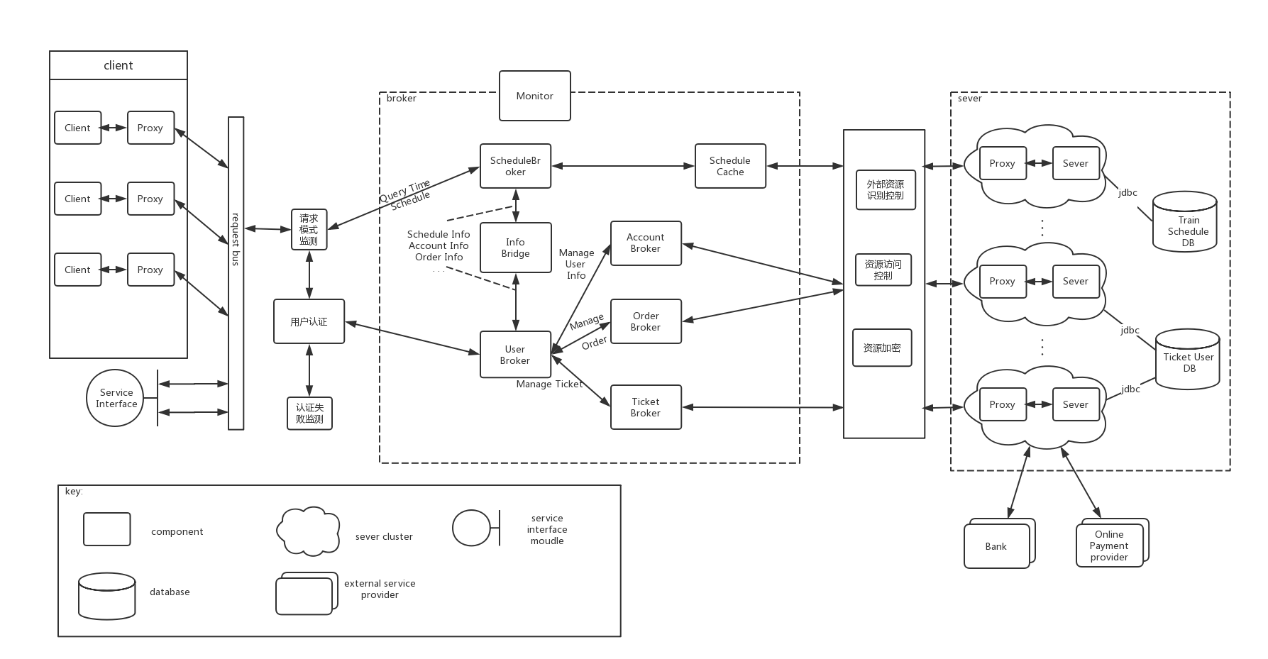
请求模式监测模块：记录请求情况，对比请求模式。在

用户认证模块：确认用户身份，认证权限

外部资源识别控制模块：对输入系统内资源进行识别和转化，包括文件提交，输入数据等，避免注入攻击。

资源访问控制模块：控制对系统资源的访问请求

资源加密模块：对需要加密的输入资源进行加密



## 第四次迭代

#### 选择元素

第四次迭代选择的元素是monitor监测模块。

#### 选择ASR

第二次迭代选择的ASR是可用性、性能。

对于该系统架构， broker的引入带来了系统错误的集中产生点，此处产生的错误必然给项目整体带来较大的影响。Monitor作为监控组件，需要保证核心broker模块正常运行，固然可用性必不可缺。当某错误产生时，该模块同样需要具有应对错误的能力，在最短时限内完成错误处理任务，所以说性能同样重要。

#### 候选策略表和决策

#### 可用性

|  |  |  |  |
| --- | --- | --- | --- |
| 策略 | 优点 | 缺点 | 是否采用 |
| Heartbeat | 持续周期性监测，并且可以进行少量数据传递交换组件状态信息 | 对于监测组件的性能要求高，实时监听的情况下还要有能力应对可能出现错误 | 采用。周期性的通信检查，可以有效保证被监测组件可能出现的错误。 |
| 消极冗余 | 仅有一个组件响应请求，其他冗余组件与之保持状态匹配，随时可以进行替换 | 占用一定资源，当前组件和冗余组件之间存在频繁的信息交流 | 采用。在不占用太多资源的情况下，能够有效处理错误出现的情况。 |
| Ping/echo | 通过时长和通信检查模块之间的连通性，常用于远程模块之间 | 模块之间的主动连通性检查，不定期，模块间联系复杂，压力增大 | 不采用。Ping/echo主要适用于远程模块之间的通信检查，不定期检查。 |
| 投票 | 对于逻辑控制模块有有效的监测作用，通过对比不同投票者的输出侦测错误 | 适用场景有限，投票逻辑定义困难，消耗资源较多 | 不采用。投票逻辑难以定义，因为broker模块内组件职责差异较大，且消耗资源过多。 |
| 积极冗余 | 所有组件均响应请求，只有一个组件的响应被采用，动态调节哪个响应被采用 | 占用资源大 | 不采用。本系统对于性能要求高，本策略增大了请求处理的任务量。 |
| 冷冗余 | 对发生错误组件进行替换，保证错误的出现不影响系统 | 系统宕机时间略长，多用于可以人工操作的组件 | 不采用。操作复杂，本系统对于宕机时长要求高。 |
| 事务 | 对于错误的恢复身份有效，处理操作能被回滚 | 事务的记录占用资源较大 | 不采用。消耗资源过大，并且监控模块的任务不适合事务化。 |

#### 性能

|  |  |  |  |
| --- | --- | --- | --- |
| 策略 | 优点 | 缺点 | 是否采用 |
| 限制操作时间 | 控制一项请求的操作时间，保证处理资源的合理分配 | 对于实时记时组件的性能要求较大 | 采用。对broker处理请求的时长进行上限监控，保证broker资源被合理分配，不会因为处理时间过长而导致请求阻塞。 |
| 增加资源 | 从基本层面增强模块的整体处理能力 | 产生预算 | 采用。增强monitor模块的处理能力、资源空间，保证其能高效完成监测任务。 |
| 引入并行 | 并行处理，减少阻塞等待时间 | 对于并行处理的组件性能要求高，可能引入并发问题 | 采用。引入不同的线程并行处理不同的监控任务。 |

#### 第四次迭代结果

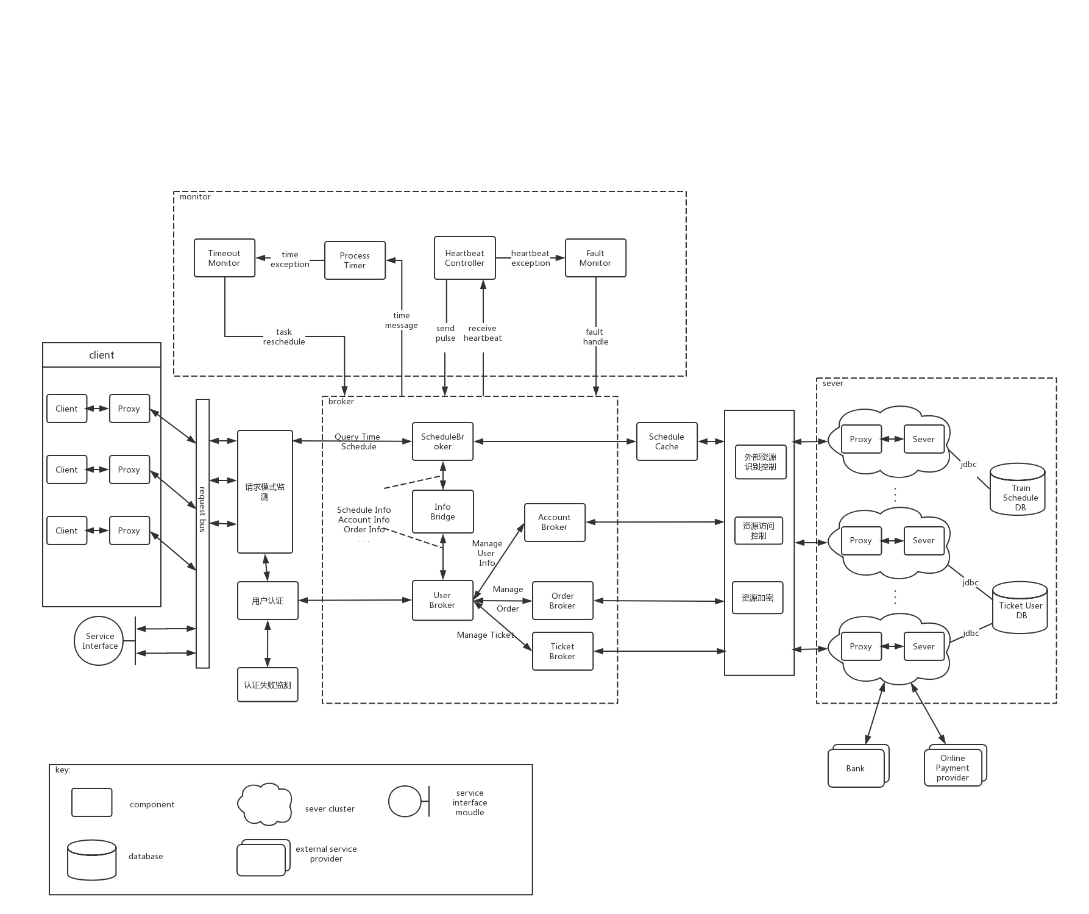
新增元素如下：

心跳控制模块（HeartbeatController）：周期性负责发送脉冲信号并且对broker的响应做 出解析，当解析异常时发送异常报告给错误监控模块

错误监控模块（FaultMonitor）：接受来自心跳控制模块的异常报告，并且处理异常情况

进程记时模块（ProcessTimer）：记录broker传来的时间信息，控制每次请求的响应时 间，如果判断超时，将发送超时异常给超时监控模块

超时监控模块（TimeoutMonitor）：接受来自进程记时模块的超时异常，对broker等待 任务集进行动态调度



## 第五次迭代

#### 选择元素

第五次迭代选择的元素是服务器模块。

#### 选择ASR

第二次迭代选择的ASR是安全性、互操作性、可用性、易修改性。

1. 因为12306的用户群体非常广大，所以安全性尤为重要，如果12306的服务器模块被黑客成功攻击，那么会对整个系统都会是非常大的打击，而且可能引发更大的隐患。
2. 因为12306系统设计财务流通、身份识别，所以需要与银行系统与公安系统进行交互，用于财务转账、身份识别，所以互操作性对于12306系统来说很重要。
3. 因为12306的用户群体非常广大，所以系统要保持高可用性，这就需要服务器端一直要保持高可用的状态。
4. 因为12306业务逻辑可能会发生变化，，所以这使得在服务器端进行修改造成的对其他的模块的影响要尽可能小，所以12306的易修改性要尽可能好。

#### 候选策略表和决策

|  |  |  |  |
| --- | --- | --- | --- |
| 策略 | 优点 | 缺点 | 是否采用 |
| 攻击发生时收回数据访问权限 | 在遭受攻击时，启动自我防御机制。极大增加了系统的安全性。 | 增加了失效时间，降低了系统的可用性。 | 采用。增加攻击侦测模块，当侦测到攻击发生时，服务器模块将会拒绝一切外界访问，直到确认安全。 |
| 数据加密 | 对数据加密，保证了数据的安全性。 | 对数据的加密需要消耗额外的资源。 | 采用。因为与银行系统的交互的信息可能被拦截和监听，所以加密是必须的。 |
| 用户认证 | 能够有效阻止一些外部的攻击，增加了安全性。 | 验证工作消耗额外资源。同时可能会有不短猜测以获取验证身份的行为。 | 不采用，因为此时处理的是服务器模块的安全性事务，用户认证会由客户端进行工作，这边再做就是重复工作。 |
| 限制访问 | 限制访问范围，增加了对敏感数据的保护，提高了安全性。 | 消耗大量额外的资源，同时也需要系统构建者额外的工作量。 | 采用。对于查询的用户，将其权限在只能查看自己想找的信息之内。对于其他操作的用户，同样如此。不透露多余的任何信息。 |
| 保证冗余的服务器和网络连接 | 保证了系统失效的时候能够立即采取措施， | 需要大量的计算资源作为候补。 | 采用。如果当前正在运行的服务器突然失效，则立即使用冗余的服务器和网络连接。 |
| ping/echo | 能随时对系统发起询问请求。 | 通讯量是heartbeat方案的双倍。 | 不采用。因为采用ping/echo的机制的检查消耗通讯量大。 |
| heartbeat | 定期检测系统是否已经失效。消耗资源较少。 | 在系统内部要实现额外的heartbeat相关逻辑。 | 采用。因为采用heartbeat的机制所进行的检查是定期的，可靠，而且消耗通讯量较少。 |
| monitor | 能即时监控整个系统，有利于立即应用系统失效的情况。 | monitor一直监测整个系统，消耗了大量的系统资源。 | 不采用。因为monitor机制占用了太多的系统资源。 |
| 服务定位（locate） | 采用目录命名机制，方便了与外界系统的交互行为。 | 维护服务目录需要大量的资源。 | 采用。在一个已知的目录服务里面，去定位自己想找到的服务，比如银行消费服务、公安人员监测服务。 |
| 裁剪接口 | 通过接口很好地限制地系统与外界的交互范围。 | 对接口的修改影响整个全局的功能。 | 不采用。12306系统的服务器端主要是作为服务的使用者而不是服务的提供者。 |
| 模块分解 | 将职责不同的部分分解出来，使得他们相对于其他部分独立，更加容易独立于其他的部分进行修改。 | 分解要消耗一定的成本，同时可能会增加复杂度。 | 采用。可以将服务器中不同逻辑部分拆分来进行分解。 |
| 延迟绑定 | 到最后才决定执行的动作。 | 越往后就越有多种可能性，会使得复杂度提高。使得成本变高，测试效率变低。 | 不采用。提高了复杂度。 |
| 重构（refactor） | 通过抽象出共同的service，可以降低内聚性，使得代码的效率更高。 | 成本较大。 | 不采用。成本太大。 |

#### 第五次迭代结果

如图所示，新增元素说明如下。

攻击侦测模块：侦测非正常访问，在攻击发生时禁止外界访问服务器模块。

数据加密模块：对服务器与外部系统交互时传送的数据进行加密和解密工作。

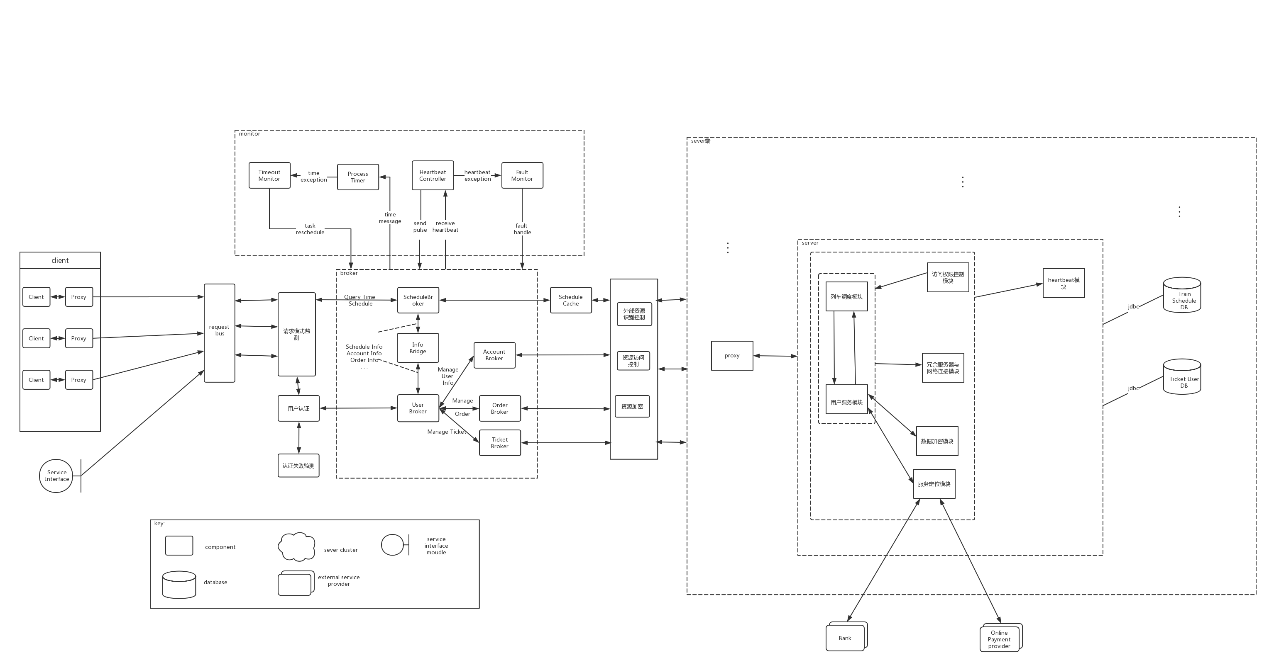
冗余的服务器与网络连接模块：作为候补，等待当前运行的服务器失效或者网络连接失效而发挥作用。

heartbeat模块：监测服务器模块是否失效，以尽快做出反应。

访问权限控制模块：监测用户的访问范围，使用户的访问范围始终被限制在必须需要访问的范围内。

服务定位模块：用于处理与外部系统的交互，将外部系统视为服务的提供者，从而简化系统的工作。

服务器端内部拆分为列车调度模块与用户票务模块。



## 第六次迭代

#### 选择元素

第六次迭代选择的元素是请求总线模块。

#### 选择ASR

第二次迭代选择的ASR是性能、可用性。

1. 因为12306的用户群体非常广大，所以经常会出现高并发的情况，这时候要考查系统的性能，在压力情况下，系统要能够正常运行而不是崩溃，所以请求总线需要对于请求能够合理分派，保持负载均衡。
2. 因为12306购票系统不能出现长时间失效的情况，而如何请求总线一旦失效，那其实整个系统也就因为这一个组件而失效。因此确保请求总线的可用性是非常重要的。

#### 候选策略表和决策

|  |  |  |  |
| --- | --- | --- | --- |
| 策略 | 优点 | 缺点 | 是否采用 |
| 智能分派 | 能够识别出请求对应的具体broker，将其直接分派到对应的broker上，从而实现智能分派，避免造成单个broker的压力过大。 | 需要额外的工作量。 | 采用。 |
| 队列分类 | 对于user broker再构建一个user broker的请求队列，对于schedule broker构建一个schdule broker的请求队列。提高了效率。 | 耗费资源。 | 采用。 |
| 增加资源 | 提高了请求总线的效率，从而提高处理请求效率。 | 耗费资源。 | 采用。 |
| 限制执行时间。 | 给每一个请求的处理确定一个执行时间的上限，避免单个请求耗费时间过多，造成性能下降。 | 可能会错失一些请求造成一些请求没有回应。 | 不采用。错失请求的范围扩大会演变为失效情况。 |
| 请求总线冗余 | 防止正在运行的请求总线的失效情况。如果其失效，则用冗余的总线来代替它。 | 耗费资源。 | 采用。 |
| ping/echo | 能随时对系统发起询问请求。 | 通讯量是heartbeat方案的双倍。 | 不采用。 |
| heartbeat | 定期检测系统是否已经失效。消耗资源较少。 | 在系统内部要实现额外的heartbeat相关逻辑。 | 采用。在server对请求响应时，不发出heartbeat以免占用资源，在server不处理请求的空闲时刻，发出heartbeat来确保其没有失效。 |

#### 第六次迭代结果

如图所示，新增元素说明如下。

智能识别模块。用于识别正在处理的请求它对应的是user broker还是schedule broker，并把请求放入对应的broker的请求队列中。

user broker请求队列。用来给对于user broker的请求进行排队。

schedule broker请求队列。用来给对于schedule broker的请求进行排队。

总线冗余模块。提供冗余，防止当前总线失效。

heartbeat监测模块。用于检测系统是否失效，帮助其在失效情况下采取相应措施。

