1。可用

2。负载

3。容量

4。实时

5。易用

6。安全

7。互操作

8。可修改可维护

10.1 第三次迭代过程

选取课程注册模块作为系统元素进行分解。在质量属性方面，该模块与负载性能、互操作性、安全性、实时性等关系密切。

10.1.1 识别所选模块的ASR

|  |  |  |  |
| --- | --- | --- | --- |
| 编号 | Achitectural Drivers | Importance | Difficulty |
| 1 | Senario2: 性能负载，有大量学生同时对某一或某些课程进行注册 | High | High |
| 2 | Senario1: 可用性，大量学生访问系统时，系统要保持可用 | High | Medium |
| 3 | Senario7: 互操作性，对于需要付费的课程，学生注册时系统需要与银行系统交互 | High | Low |
| 4 | Senario6: 安全性，学生支付课程费用时，必须保障网络环境的安全 | High | Medium |

10.1.2 每个ASR可选择的设计策略和理由

性能负载

|  |  |
| --- | --- |
| 可选的设计决策和模式 | 理由和分析 |
| 负载均衡器 | 可以保证以相对较优的方式分配请求，但需要着重考虑成本问题 |
| 反向代理 | Nginx，具有高性能，轻量级的优势，处理高并发请求时有相当高的性能 |
| 内容分发网络CND | 可以根据用户的区域信息分配最近的CDN服务器，分解服务器的压力，但需要考虑CDN服务器的数据同步问题 |
| DNS负载均衡 | 由DNS服务器充当负载均衡调度器，将用户请求分散到多台服务器上，但是调试成本高，分配策略也有很强的局限性。 |
| 增加可用资源 | 有助于提高性能，但是需要考虑成本投入 |

可用性

|  |  |
| --- | --- |
| 可选的设计决策和模式 | 理由和分析 |
| Heartbeat检测 | 周期性地检测系统服务是否可用 |
| Ping-echo | 客户端发起请求，检测系统服务是否可用 |
| 主动冗余 | 所有冗余组件以并行方式对事件作出响应，如果发生错误，系统可以快速恢复 |
| 被动冗余 | 主要组件对事件作出响应，并通知其他组件进行必要的更新，如果发生错误，系统可以在一定时间内恢复 |

互操作性

|  |  |
| --- | --- |
| 可选的设计决策和模式 | 理由和分析 |
| 在服务列表中定位所需服务 | 当所需服务较多时可以快速定位所需服务 |
| 选取和剪裁外部系统的接口 | 外部系统提供的接口可能有很多，有些接口返回的数据可能过多，并不是系统所需要的 |
| 提供多种数据格式 | 多种数据格式来保证系统可以使用不同的外部服务，以防出现数据不匹配的问题 |

安全性

|  |  |
| --- | --- |
| 可选的设计决策和模式 | 理由和分析 |
| 用户数据传输加密 | 加密用户数据以防被盗用，保障财产安全 |
| 自动攻击检测 | 针对网络攻击进行检测，但是会增加系统运行负担 |
| 拒绝可疑访问 | 可以较好地保证访问的合法性，但是可能会引起误伤 |

10.1.3 设计决策的选择和分析

性能负载

|  |  |
| --- | --- |
| 可选的设计决策和模式 | 理由和分析 |
| 负载均衡器 | 不采用，第三方软件虽然可以保证以相对较优的方式分配请求，但是成本过高。 |
| 反向代理 | 采用，高性能，轻量级，拥有众多请求分配策略，有助于处理高并发请求且技术成熟。 |
| 内容分发网络CND | 采用，可以很好的分担服务器的压力，且国内各大CND服务商发展成熟。 |
| DNS负载均衡 | 不采用，分配策略具有局限性，无法根据实际服务器的实时负载差异来调整调度策略，记录缓存过于复杂，并不能很好地完成工作量分配。 |
| 增加可用资源 | 不采用，主要是成本过高。 |

可用性

|  |  |
| --- | --- |
| 可选的设计决策和模式 | 理由和分析 |
| Heartbeat检测 | 采用，周期性检查可以及时检测服务宕机以采取措施 |
| Ping-echo | 不采用，系统服务不可用时可能不能及时检测 |
| 主动冗余 | 不采用，多个备件并发对事件作出响应，需要系统提供更多资源，会增加成本 |
| 被动冗余 | 采用，课程注册虽然要求很高的可用性，但由于课程名额无限制，用户更关系自己是否报名成功，在一定时间内恢复系统是可以接受的。 |

互操作性

|  |  |
| --- | --- |
| 可选的设计决策和模式 | 理由和分析 |
| 在服务列表中定位所需服务 | 不采用，这里只用到外部系统的支付服务，并不是应对多种服务需求。 |
| 选取和剪裁外部系统的接口 | 采用，选取需要使用的支付服务接口，对接口重新剪裁有助于获取更加明确的支付信息。 |
| 提供多种数据格式 | 采用，不同的支付服务可能采用不同的数据格式，提供多种数据格式有利于与多个外部服务的适配 |

安全性

|  |  |
| --- | --- |
| 可选的设计决策和模式 | 理由和分析 |
| 用户数据传输加密 | 采用，加密用户数据以防被盗用，保障财产安全 |
| 自动攻击检测 | 不采用，加重了系统的运行负担，且第三方支付服务本身对安全性也有较好地保障 |
| 拒绝可疑访问 | 不采用，可能会误伤正常用户的访问，降低用户体验和系统的可用性 |

10.1.4 第三次迭代结果

