

评估报告

1. 质量属性效用树

质量属性	属性精华 (concrete Attribute)	ASR(Scenarios)
可靠性	宕机修复	不超过百分之二十的任意节点宕机后系统仍能够正常使用，宕机节点能够在 1 个自然日内完全恢复 (H,H)
	业务正确性	保证同一时刻只有一次请求被响应,一个医生只能同时进行一次问诊，服务一个用户(H,M)
性能	用户访问	任何情况下，请求丢失的比例不能超过 0.01%；用户的请求响应时间不能超过 500ms;最少可以支持 100000 用户同时在线登陆
可伸缩性	硬件升级	服务器和数据库等硬件升级，热升级可以在 4 小时内全部完成，期间所有服务请求仍可正常响应(M,M)
可扩展性	模块扩展	增加医生交流的模块，交流模块的增加在 2 小时内完成，增加影响的源代码不超过5% (M,M)
易用性	用户信息缓存	推送预约排队到达消息给患者，并保利用户能在5s内获得消息(M,L)
安全性	检测攻击，防御攻击	阻止恶意预约影响他人的问诊进度，系统应在1s内检测到恶意预约请求，并在 1s内冻结该用户 (H,L)
	数据加密	系统能够检测到攻击，同时采取相应保护措施，阻止攻击行为,阻止用户的数据被盗取(H,L)
互操作性	使用接口包装	能够应对第三方接口发生变化，系统更新可在系统不下线的情况下在 1 天内完成(L,H)
可配置性	使用配置文件	可通过配置文件修改策略，系统在 5 秒内根据配置信息的改变完成推送策略变更(L, H)
可维护性	硬件升级	服务器和数据库等硬件升级，热升级可以在 4 小时内全部完成，期间所有服务请求仍可正常响应(M,M)
	建立日志	测试人员应对出现 Bug 的修订工作，定位到 bug 位置的时间不超过 20 分钟；修订时间不超过 120分钟；修订所影响的代码量不超过 2%(L,M)
可测	建立日志	测试人员应对出现 Bug 的修订工作，定位到 bug 位置的时间不超过 20 分钟；修订时间不超过 120分钟；修订所影响的代码量不超过 2%(L,M)

试性	控制系统复杂度，封装模块	第三方和系统本身可以独立测试，在 3 小时之中能达到 85%的路径覆盖率(L,M)
----	--------------	---

2. 敏感点

编号	决策	质量属性	分析
S1	心跳包	可靠性	可以实现周期性检测组件是否正常工作，但是会占用部分计算资源
S2	并行处理请求	性能	处理相关逻辑时可以减少用户的等待，但是并行的调度计算和资源成本消耗较大
S3	根据优先级将请求排序	性能	可以优先处理预约相关的较为重要的请求，但是低优先级的处理速度较慢
S4	检测入侵，比较网络通信模式	安全性	通过比较网络通信模式，防止恶意预约影响他人的行为，是安全性的敏感点
S5	数据加密	安全性	通过对用户数据进行加密，对数据采取保护措施，阻止用户数据被盗取
S6	增加配置模块，用配置文件改变分发策略	可配置性	通过修改配置文件修改策略，正面影响了可配置性，是可配置性的敏感点
S8	建立日志	可测试性，可维护性	通过建立日志，便于测试工作的进行和bug的发现与重现，正面影响了可测试性和可维护性，是可测试性和可维护性的敏感点
S9	对模块进行封装，控制系统复杂度	可测试性，互操作性	不仅代码易于测试，也易于修改，可以对第三方和系统进行独立测试，实现了高内聚低耦合，正面影响了可测试性和互操作性
S10	硬件热升级	可伸缩性，可维护性	在不中断服务的情况下，对数据库和服务器等升级，正面影响了可伸缩性，是可伸缩性的敏感点
S11	使用Hibernate框架	可伸缩性，可维护性	对JDBC访问数据库代码进行封装，能支持切换各种关系型数据库，正面影响了可伸缩性和可维护性，是可伸缩性和可维护性的敏感点
S12	预约用户信息缓存队列	易用性	通过缓存用户信息，当预约将要到达用户的时候，可以让用户快速问诊

3. 权衡点

编号	决策	质量属性	分析
T1	心跳包	可靠性, 性能	可以实现周期性检测组件是否正常工作, 但是会占用部分计算资源
T2	检查	可靠性, 性能	对输入进行合理性检查, 提高了安全性, 但是过多检查会影响性能
T3	使用缓存	性能, 可靠性	对进行了预约的用户, 通过缓存这些用户的信息, 来提高响应速度, 但不同服务器之间, 数据保持一致性的要求比较高
T4	增加医生交流模块	可修改性, 性能	增加医生交流模块, 拓展了聊天实现的功能, 可以复用咨询模块的功能
T5	使用 Hibernate 框架	可伸缩性, 性能	可伸缩性, 性能 在关系型数据库结构发生 升级改变时, 不需要修改 其他部分。由于封装了 JDBC, 所以没有 JDBC 直接访问数据库效率高, 且配置文件复杂, 降低了 性能
T6	缓存最近得到的用户信息	易用性, 可靠性	缓存可以减少预约模块和用户管理模块的交互, 可以快点得到用户的信息, 但有可能存在数据不一致性, 影响推送结果
T7	检测入侵, 比较网络通信模式	安全性, 性能	虽然提高了安全性, 但是可能增加每一条请求的时间, 降低性能, 是安全性和性能的权衡点
T8	数据加密	安全性, 性能	提高了安全性, 但是需要更多资源来对数据进行加密处理, 降低了性能, 是安全和性能的权衡点
T9	增加配置服务模块	可配置性	虽然提高了可配置性, 但是容易成为系统被攻击的地方, 是安全性和可配置性
T10	建立日志	可维护性, 性能	虽然提高了可维护性, 但是需要更多的硬件资源和记录操作, 降低了性能, 是可维护性和性能的权衡点

4. 风险

编号	决策	分析
R1	心跳包	心跳发送速率过低可能会影响到检测
R2	没有业务服务器的冗余	没有业务服务器的冗余，对可靠性有影响
R3	主动冗余只有一个monitor	存在单点失效问题，影响可靠性
R4	增加配置模块	配置模块可能会成为攻击的主要目标，影响安全性和可靠性
R5	检测入侵，比较网络通信模式	当检测模块遭到大量集中攻击时，可能发生单点失效，影响其他正常访问
R6	建立系统日志	当出现bug时，通过记录的日志有小部分概率可能找不出错误
R7	并行处理请求	虽然能加大并发量，及时响应用户的请求，但是还是存在请求丢失，请求等待过长的风险

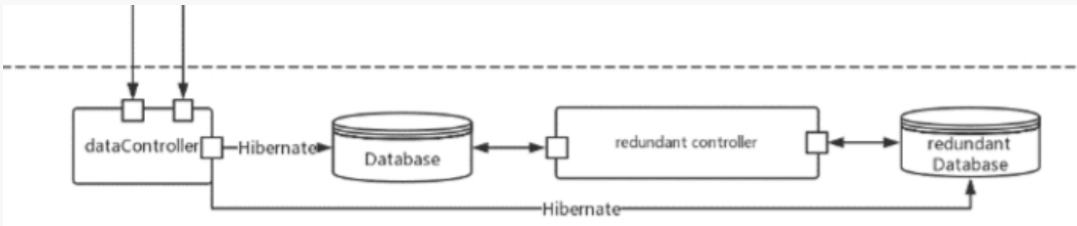
5. 非风险

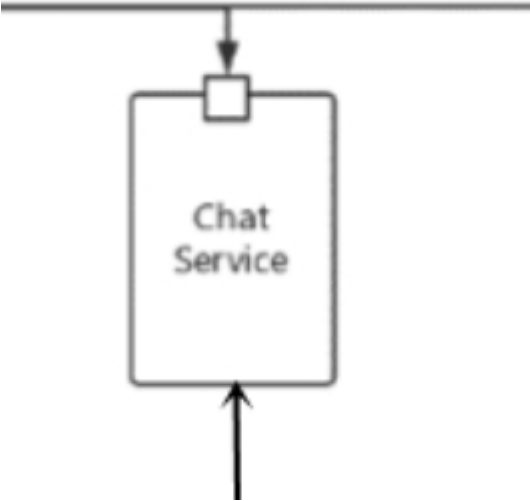
编号	决策	分析
N1	硬件升级	对数据库和服务器的升级已经有成熟的规范和方法，降低了风险
N2	模块扩展	增加医生交流的模块，而不是在医生管理模块中增加功能，降低了模块间的耦合
N3	使用接口包装	使用接口封装，可以应对第三方服务的变化，降低了系统和第三方的耦合
N4	控制系统复杂度，封装模块	封装各个模块，降低了模块之间的耦合，提高了模块间的内聚，易于对各个模块进行独立测试和修改
N5	心跳包	心跳包已经被广泛用于各成熟系统中
N6	用户信息缓存	已经有成熟的解决办法，比如redis等框架
N7	使用Hibernate框架	Hibernate框架已经很成熟，广泛应用于各类系统之中

6. ATAM分析

场景1：数据或业务服务器宕机				
质量属性	可靠性			
环境	系统正在处理某些业务			
刺激	部分节点或节点群停止工作			
响应	其它节点能够快速完整接替宕机节点的工作，且宕机节点能够快速恢复			
决策	敏感点	权衡点	风险	非风险
心跳包	S1	T1	R1	N5
分析	心跳可以快速检测到是否有组件失效			
有关架构图表				

场景3：系统运行，用户访问				
质量属性	性能			
环境	系统正常运行时			
刺激	大量用户在线进行问诊，预约排队和评价			
响应	系统执行用户请求的操作；处理用户数据；返回结果			
决策	敏感点	权衡点	风险	非风险
心跳包	S2		R7	
分析	当大量用户同时对系统进行请求时，需要考虑性能的问题，尽量减少请求等待的时间和请求丢失的几率，在预约咨询等模块需要考虑性能问题，并进行优化			
有关架构图表	<pre>sequenceDiagram participant Bus participant Publisher participant Subscriber Bus->>Publisher: activate Publisher Publisher->>Subscriber: appointment deactivate Publisher activate Subscriber Subscriber->>Bus: deactivate Subscriber</pre> <p>The diagram illustrates a message passing scenario. A horizontal line represents a communication bus. A vertical line on the left represents the publisher, and a vertical line on the right represents the subscriber. The publisher sends a message labeled 'appointment' to the subscriber. The bus is labeled 'Appointment'.</p>			

场景4：服务器和数据库硬件升级				
质量属性	可伸缩性			
环境	系统运行时			
刺激	系统需要升级硬件资源			
响应	系统在不中断当前服务的情况下完成硬件升级			
决策	敏感点	权衡点	风险	非风险
硬件热升级	S10			N1
使用 Hibernate 框架	S11	T5		N7
分析	对数据库和服务器的升级已经有成熟的规范和方法，降低了风险，相关数据库框架有许多面向对象的特性，方便开发人员开发，可移植性好，在数据库结构发生升级改变时，不需要修改其余部分			
有关架构图表				

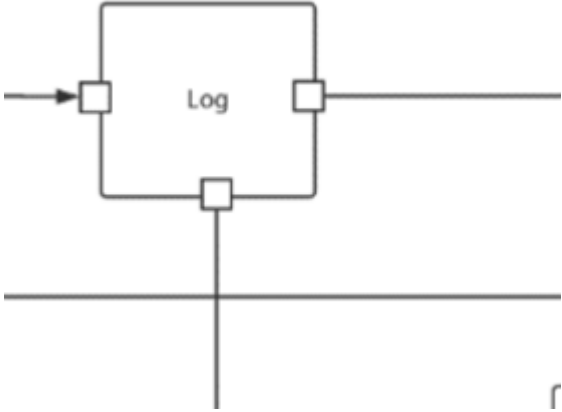
场景5：增加医生交流模块的扩展				
质量属性	可扩展性			
环境	系统在构建中或者已经上线			
刺激	系统需要增加用于医生交流的模块			
响应	系统之中增加对应模块代码，测试增加后的代码			
决策	敏感点	权衡点	风险	非风险
心跳包		T4		N2
分析	可扩展性是系统的重要属性，对医生模块进行扩展时，不应在医生管理模块中增加交流功能，为了降低模块复杂度，应该将医生交流功能隔离出来			
有关架构图表				

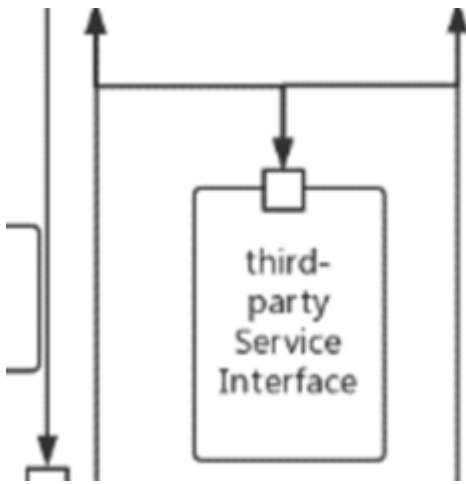
场景6：系统将即用事件循环队列，将预约到达的患者发出的问诊请求进行处理				
质量属性	易用性			
环境	系统运行时			
刺激	患者预约排队到达			
响应	系统将即用事件循环队列，将预约到达的患者发出的问诊请求进行处理，缓存患者信息			
决策	敏感点	权衡点	风险	非风险
使用循环队列，并缓存用户信息	S12	T3		N6
分析	当预约到达患者时，系统需要推送预约排队到达消息给患者，并保利用户能在5s内获得消息，使患者能够及时入诊			
有关架构图表				

场景7：阻止恶意预约影响他人的问诊进度； 场景8：阻止盗取患者个人信息				
质量属性	安全性			
环境	系统运行时			
刺激	预约多个医生却不进行问诊也不取消预约； 攻击者尝试盗取用户信息			
响应	系统冻结此恶意账号； 系统检测到攻击，同时采取相应保护措施，阻止攻击行为			
决策	敏感点	权衡点	风险	非风险
检测入侵， 比较网络通信模式	S4	T7	R5	
数据加密	S5	T8		
分析	安全性是系统重要的质量属性，所有的操作都应该由认证过的用户进行，比较通信模式能够防止恶意用户进行恶意预约； 对用户信息进行加密，阻止对用户信息的攻击，降低了用户信息泄露的危险			
有关架构图表	<pre>graph TD HTTP[HTTP请求] --> RM[Request Monitoring] RM <--> SC[ServiceController] SC --> V[Validator] V --> SC V --> S[Security] S --> SC</pre>			

场景9：能够应对第三方接口发生变化				
质量属性	互操作性			
环境	系统运行时			
刺激	服务接口发生变化			
响应	系统能够适应第三方接口变化，保持正常服务			
决策	敏感点	权衡点	风险	非风险
对第三方接口进行封装	S9			N4
分析	互操作性是系统的重要质量属性，当第三方接口发生变化时，系统更新可在系统不下线的情况下完成			
有关架构图表	<pre>graph BT; TPI[third-party interface] --> TPS[Third-party services];</pre> <p>The diagram is a UML Component Diagram. It features two components: a rectangular component labeled 'third-party interface' and a rounded rectangular component labeled 'Third-party services'. An arrow points from the 'third-party interface' component to the 'Third-party services' component, indicating a dependency. The 'third-party interface' component is positioned below the 'Third-party services' component. To the right of the components, there is a vertical line with a bracket-like symbol at the bottom, likely representing a system boundary or a package.</p>			

场景10：可通过配置文件修改策略				
质量属性	可配置性			
环境	系统已经上线			
刺激	当前患者数量和问诊平均时间发生变化			
响应	系统能够根据配置信息调整每个患者的问诊时限和等待时限等			
决策	敏感点	权衡点	风险	非风险
使用配置文件	S6	T9	R4	
分析	通过配置文件修改系统的推送策略，降低了修改系统的代价，在修改配置文件后，系统应在 5 秒内根据配置信息的改变完成推送策略变更			
有关架构图表	<p>The diagram illustrates the system architecture for configuration-based strategy modification. It features four main components: a Publish Controller, a Configuration Controller, a Pub-Sub Control Center, and a Center. The Publish Controller is connected to the Pub-Sub Control Center via an 'order' message. The Configuration Controller is connected to the Pub-Sub Control Center via a configuration update message. The Pub-Sub Control Center is connected to the Center via a message. The Pub-Sub Control Center also has a feedback loop back to the Configuration Controller. The Center is connected to the Pub-Sub Control Center via a message.</p>			

场景11：测试人员应对出现 Bug 的修订工作				
质量属性	可维护性，可测试性			
环境	系统已经上线			
刺激	系统出现无法预约等问题			
响应	定位到 bug 的位置；修订 bug			
决策	敏感点	权衡点	风险	非风险
建立日志	S8	T10	R6	
分析	日志模块是系统必不可缺的模块，它能记录下系统发生故障时产生错误的源位置，方便开发人员迅速定位bug			
有关架构图表				

场景12：第三方和系统本身可以独立测试				
质量属性	可测试性			
环境	系统开发，测试时			
刺激	完成代码单元测试			
响应	获得测试结果			
决策	敏感点	权衡点	风险	非风险
控制系统复杂度，封装模块	S9			N4
分析	不仅代码易于测试，也易于修改，可以对第三方和系统进行独立测试，实现了高内聚低耦合			
有关架构图表				

7. ATAM评估过程

phase-0 准备

参与者：评估团队组长和项目的关键决策者

形成团队：我们组的组长与对方项目的关键决策者组成团队，由他们将项目的架构文档交给我们

phase-1 进行评估（1）

参与者：评估团队和项目决策者

step 1

step 2

由对方项目的关键决策者从商业角度为我们展示了系统的概要，包括项目的功能，项目的技术、管理、经济或者政策上的约束，商业目标和环境，涉及到的主要受众

step 3

邀请了项目的架构师为我们评估团队做了一次展示，描述了以下方面：

1. 系统的技术约束，例如操作系统、硬件和中间件
2. 与系统交互的其他系统或环境
3. 用来满足质量属性需求的架构方法

step 4

此时，我们评估团队已经研究过了架构文档，已经听过了架构师的展示介绍并咨询过设计系统时采用的模式和策略

接着我们对采用的架构方法进行分类

step 5

这一步中，我们评估团队和项目决策者一起识别项目中最重要质量属性目标，并联系需求，构建出了质量属性效用树

见 1.质量属性效用树

step 6

识别架构设计中的敏感点、权衡点、风险和非风险，然后对架构中最重要的场景进行概括总结

见6. ATAM分析

phase-2 进行评估（2）

参与者：评估团队，项目决策者和架构受众

step 7

这一步中，我们广泛听取受众群体的意见，让他们进行头脑风暴，提取出他们觉得重要并且有意义的情景，提取出来之后对他们排序合并

step 8

这一步中，我们评估团队对新产生的但是很重要的情景进行和step 6 一样的分析，识别架构设计中的敏感点、权衡点、风险和非风险，然后对架构中最重要的场景进行概括总结

step 9

结束评估，展示我们的评估结果

phase-3

对评估结果进行整理，形成评估报告

8. 潜在问题

ASR-5 可扩展性 场景：增加医生交流模块的扩展

场景组成部分	可能的值
源	系统开发人员
刺激	系统需要增加用于医生交流的模块
制品	系统医生交流模块
环境	系统在构建中或者已经上线
响应	系统之中增加对应模块代码，测试增加后的代码
响应度量	交流模块的增加在 2 小时内完成，增加影响的源代码不超过5%

潜在问题：在复杂的系统中，增加一个模块需要经过严谨的研究，从讨论到设计，编码和测试，在两小时内完成比较困难，而且两小时内完成的代码质量较低，容易出错

解决方案：建议将交流模块的增加时间修改为2周以保证代码质量

ASR-3 性能 场景：系统运行，用户访问

场景组成部分	可能的值
源	用户
刺激	大量用户在线进行问诊，预约排队和评价
制品	系统
环境	系统正常运行时
响应	系统执行用户请求的操作；处理用户数据；返回结果
响应度量	任何情况下，请求丢失的比例不能超过 0.01%；用户的请求响应时间不能超过 500ms;最少可以支持 100000 用户同时在线登陆

ASR-7 安全性 场景：阻止恶意预约影响他人的问诊进度

场景组成部分	可能的值
源	外部恶意软件
刺激	以用户身份预约多个医生却不进行问诊也不取消预约
制品	系统
环境	系统运行时
响应	系统冻结此用户账号
响应度量	系统应在1s内检测到恶意预约请求，并在1s内冻结该用户

潜在问题：在高并发的情况，用户的请求响应时间会随之变长，因此要求系统在高并发情况下1s内检测到恶意预约请求较为困难而不实际。高并发下对两者时间度量不匹配，系统在检测恶意预约请求时可能会显示出较大的延迟

解决方案：考虑到高峰预约咨询期间，建议将用户的请求响应时间减少到不超过300ms