第5章 软件体系结构设计与评估

内容

- 5.1 架构为中心的软件开发过程
- 5.2 属性驱动的设计方法
- 5.3 基于模式的设计方法
- 5.4 模块设计与评估方法
- 5.5 软件体系结构评估

5.5 软件体系结构评估

5.5.1 评审体系结构的原因

5.5.2 质量属性回顾

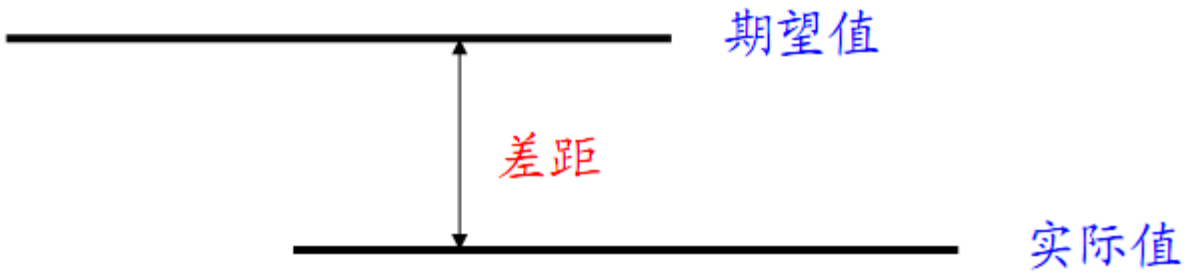
5.5.3 SA评审方法分类

5.5.4 ATAM评审方法

5.5.1 评审体系结构的原因

什么叫“评审”(Evaluation)

- Evaluation is the systematic determination of merit, worth, significance and cost, deficiency of something or someone. (评审：以某种系统化的方法对事物进行评价，以发现其优点和缺点，并判断其是否满足期望的要求)



评审体系结构的原因

- Goals of software development (软件开发的目标):
  - Function soundness and completeness (功能的正确性和完整性);
  - Performance optimization or tradeoff (性能的优化与折中);
- The achievement of a software system's qualities attributes depends much more on the software architecture than on code-related issues. (SA的设计是整个软件开发过程中的关键步骤，系统的质量属性主要由SA所决定)
- How do you know if a software architecture for a system is suitable without having to build the system first? (如何在没有开发系统之前就能够得知，一个系统的SA设计是恰当的？)

评审体系结构的原因

- All design involves tradeoff in system qualities (寻求“折中”)
  - System qualities are largely dependent on architectural decisions (体系结构极大地影响系统质量)
  - Promoting one quality often comes at the expense of another quality (提高一个质量，经常会降低另一个质量)

- A software architecture is the earliest life-cycle artifact that embodies significant design decisions: choices and tradeoffs. (“选择与折中”是设计中首要考虑的问题，软件体系结构是软件生命周期中最早一个遇到此问题的阶段)
  - Choices are easy to make, but hard to change once the system is implemented (选择很容易做，但是一旦系统已经实现，就很难更改)

评审体系结构时需要回答的问题

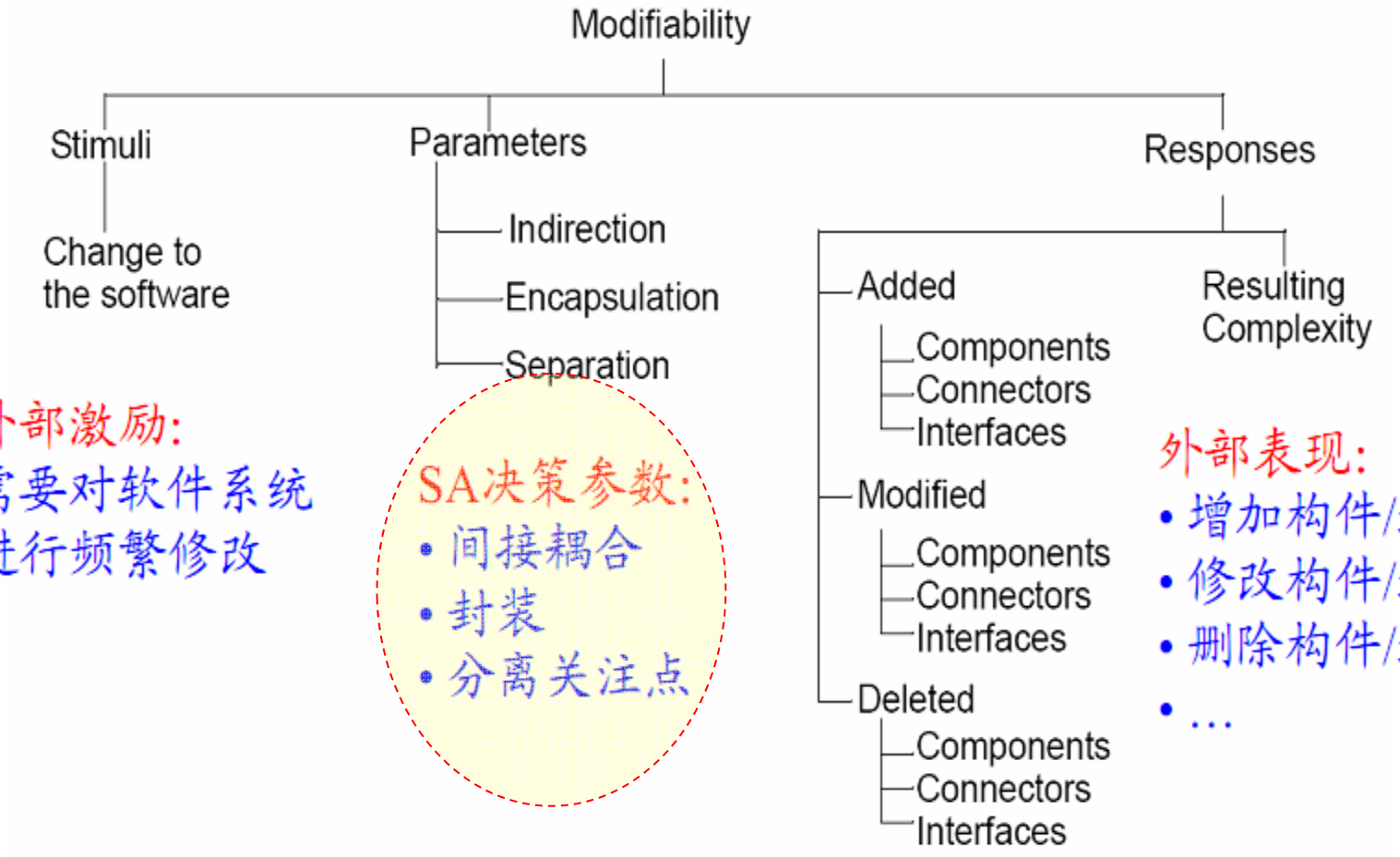
- For any particular software architecture design:
  - What precisely do these quality attributes - modifiability, security, performance, reliability - mean? (都有哪些质量属性？各自都是什么含义？)
  - Can a system be analyzed to determine these desired qualities? (为了确定期望的质量，SA如何被分析？)
  - How soon can such an analysis occur? (这种分析何时发生？)
  - What happens these quality attributes are in conflict with each other? (当各质量属性相互冲突时如何处理？)
  - How can the tradeoffs be examined, analyzed, and captured? (如何进行具体的折中分析？)

5.5.2 质量属性回顾

与质量属性相关的要素

- External stimuli: What are the stimuli to which the architecture must respond? (体系结构必须响应的外部激励是什么？)
- Responses: What is the measurable or observable manifestation of the quality attribute by which its achievement is judged? (质量属性的可观测/可度量外部表现是什么？)
- Architectural decisions (Parameters): What are the key architectural decisions that impact achieving the attribute requirement? (影响质量属性的关键体系结构决策参数是什么？)

举例：可修改性的相关要素



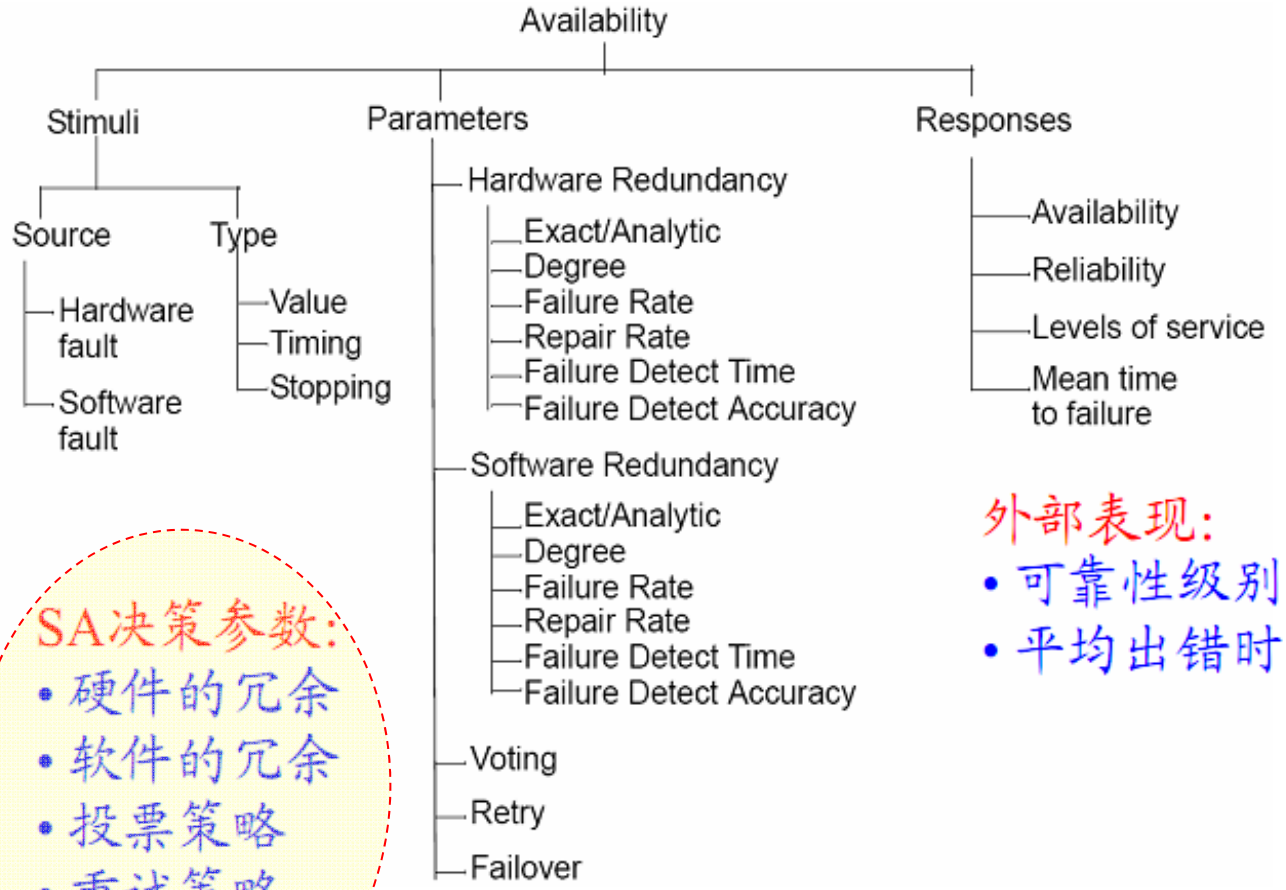
外部激励：
需要对软件系统进行频繁修改

SA决策参数：
- 间接耦合
- 封装
- 分离关注点

外部表现：
- 增加构件/连接件/接口
- 修改构件/连接件/接口
- 删除构件/连接件/接口
- …

举例：可用性的相关要素

外部激励：
- 硬件出错
- 软件出错
- 值出错
- 计时出错
- 系统停止运行



SA决策参数：
- 硬件的冗余
- 软件的冗余
- 投票策略
- 重试策略
- …

外部表现：
- 可靠性级别
- 平均出错时间

5.5.3 SA评审方法分类

三类主要的评审方式

- 基于调查问卷的评审方式
- **基于场景的评审方式**
- 基于度量的评审方式

（1）基于调查问卷的评审方式

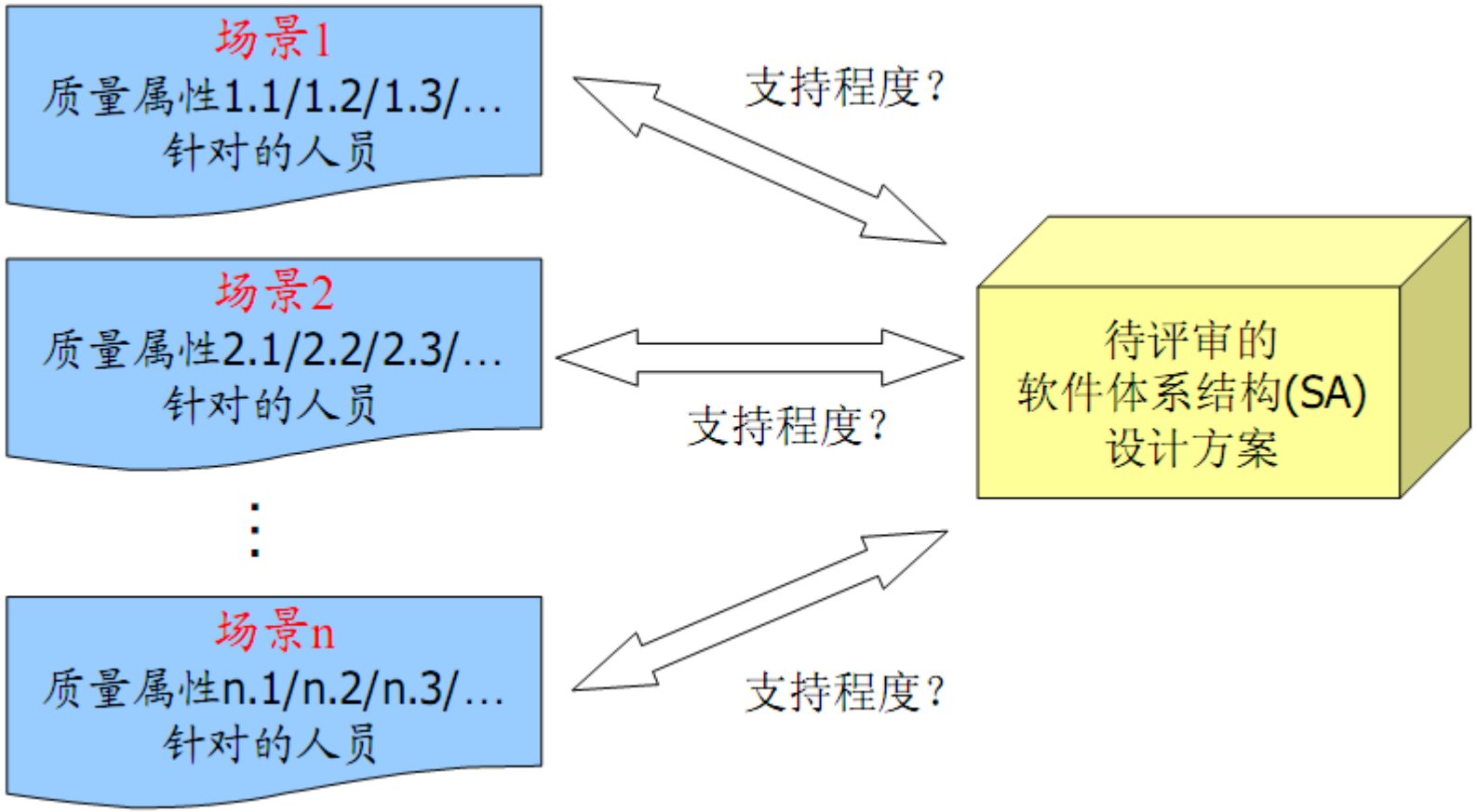
- 调查问卷：一系列有关SA评审的相关问题。
- 评估过程：
  - 多个评估专家考察系统，然后回答问卷中的问题；
  - 对多个评估结果进行综合，得到最终的结果。
- 优点：
  - 自由、灵活，可评估多种质量属性，也可在SA设计的多个阶段进行。
- 缺点：
  - 由于评估的结果很大程度上来自评估人员的主观推断，因此不同的评估人员可能会产生不同甚至截然相反的结果；
  - 评估人员对领域的熟悉程度、是否具有丰富的相关经验也成为评估结果是否正确的重要因素。

（2）基于场景的评审方式

- 基于场景的评审方式：
  - 分析SA对场景的支持程度，从而判断该SA对这一场景所代表的质量需求的满足程度。
  - 场景(Scenario)：系统应用的典型场合。

- 基于场景的方式由SEI首先提出：
  - 体系结构权衡分析方法(Architecture Tradeoff Analysis Method , ATAM)
  - 软件体系结构分析方法(SAAM)

（2）基于场景的评审方式（续1）



（2）基于场景的评审方式（续2）

- 该类方式考虑到了包括开发人员、维护人员、最终用户、管理人员、测试人员等在内的所有相关的人员对质量的要求。
- 基本活动：
  - 设计用于体现待评估质量属性的场景；
  - 分析软件体系结构对场景的支持程度。
- 实施该方法的要求较高：
  - 需要有丰富的领域知识以便对质量属性设计出合理的场景；
  - 必须SA有充分了解以判断它是否支持场景描述的一系列活动。

（3）基于度量的评审方式

- 设计质量指标的具体度量方式(数学公式)。
- 在度量时，可采用自动化的工具，对质量属性进行计算，得到结果。

$$Coupling(MS) = \sum_{i=1}^{n} I(S_i) - I(S)$$

- 三个基本活动：
  - 从软件体系结构文档中获取度量信息并计算得到结果；
  - 建立质量属性和度量结果之间的映射原则，即确定怎样从度量结果推出系统具有什么样的质量属性；
  - 根据映射原则分析推导出系统的某些质量属性。

（3）基于度量的评审方式

- 优点：
  - 结果比较客观、精确。
- 缺点：
  - 很多质量属性无法给出具体的计算公式；
  - 能给出计算公式的质量属性，往往是针对源代码级别的质量属性(如代码行数、方法调用的次数、构件个数等)，而这些属性对SA的评价往往缺乏足够的意义；
  - 需要在SA设计基本完成以后才能进行；
  - 需要评估人员对待评估的体系结构十分了解。

三种评审方式的比较

评估方式	调查问卷或检查表		场景	度量
	调查问卷	检查表		
通用性	通用	特定领域	特定系统	通用或特定领域
评估者对体系结构的了解程度	粗略了解	无限制	中等了解	精确了解
实施阶段	早	中	中	中
客观性	主观	主观	较主观	较客观

5.5.4 ATAM评审方法

ATAM体系结构评审方法

Rick Kazman, Mark Klein, Paul Clements. ATAM: Method for Architecture Evaluation. Technical Report, CMU/SEI-2000-TR-004, 2000.

ATAM

- ATAM: Architecture Tradeoff Analysis Method (体系结构权衡分析方法)
- The purpose of the ATAM is to assess the consequences of architectural decisions in light of quality attribute requirements. (ATAM的目标：按照质量需求，评价体系结构设计)

- ATAM是一种基于场景的SA评估方法。

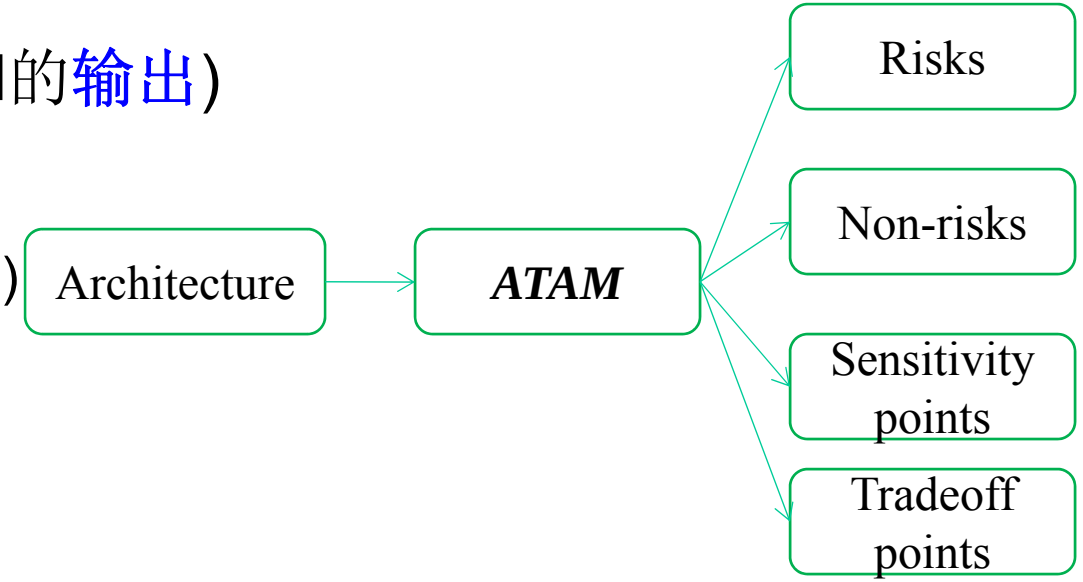
ATAM的目标

- Discover risks - alternatives that might create future problems in some quality attribute (发现风险：可能在未来产生质量问题的方案)
- Discover non-risks - decisions that promote qualities that help realize business/mission goals (发现非风险：可以提高质量的决策)
- Discover sensitivity points - alternatives for which a slight change makes a significant difference in some quality attribute.(发现关键点：方案中一个小小的变化，就可能让质量完全大变样)
- Discover tradeoffs - decisions affecting more than one quality attribute (发现折中：影响一个以上质量的决策)
  - Tradeoff between security and performance (安全性与性能的折中)

ATAM的核心概念

ATAM的核心概念

- Input of ATAM (ATAM的输入)
  - Scenario (场景)
  - Quality attribute specific Questions (需要关注的质量属性及对其的提问)
  - Architectural design (待评价的**SA的设计方案**)
- Intermediate of ATAM (ATAM的中间结果)
  - **Utility tree** (效用树)
- Output of ATAM (ATAM的输出)
  - Risk (风险)
  - Non-risk (非风险)
  - Sensitivity point (关键点)
  - Tradeoff point (折中点)



输入1：场景(Scenario)

- A scenario is a short statement describing an interaction of one of the stakeholders with the system. (场景是一种简短的陈述，用来描述系统的相关人员之间使用该软件系统的一个交互场合)

- Stakeholder (相关人员):
  - End user (终端用户)
  - Software Architect (架构师)
  - Developer (开发人员)
  - Maintainer (维护人员)
  - Tester (测试人员)
  - Market persons (市场人员)

场景的三种类型

- Use case scenarios (用例场景)
  - these involve typical uses of the existing system and are used for information elicitation; (软件系统的常规应用场合)
- Growth scenarios (变化性场景)
  - these cover anticipated changes to the system; (软件系统可预料到的变化)
- Exploratory scenarios (探测性场景)
  - these cover extreme changes that are expected to “stress” the system; (极端的变化，对系统进行压力测试)

1.1 用例场景的例子

- Use case scenarios (用例场景)
  - Remote user requests a database report via the Web during peak period and receives it within five seconds. (performance) (远程用户通过Web请求数据报表，高峰时刻可以在5秒之内得到结果)
  - The caching system will be switched to another processor when its processor fails, and will do so within one second. (reliability) (如果缓存系统处理失败，那么将在1秒之内被切换到另一个处理器)
  - User changes graph layout from horizontal to vertical and graph is redrawn in one second. (performance) (图形从水平布局切换到垂直布局，重画要在1秒之内完成)

- Growth scenarios (变化性场景)
- Exploratory scenarios (探测性场景)

1.2 变化性场景的例子

- Use case scenarios (用例场景)
- Growth scenarios (变化性场景)
  - Add a new message type to the system's repertoire in less that a person-week of work. (使用少于1人周的工作量即可在系统中加入一种新的消息类型)
  - Migrate to a new operating system, or a new release of the existing operating system in less than a person-year of work. (使用少于1人周的工作量即可将系统迁移到其他OS)
  - Double the size of existing database tables while maintaining 1 second average retrieval time. (可以将数据表的尺寸增加一倍，但其平均存取时间却仍然维持1秒)
- Exploratory scenarios (探测性场景)

1.3 探测性场景的例子

- Use case scenarios (用例场景)
- Growth scenarios (变化性场景)
- Exploratory scenarios (探测性场景)
  - Half of the servers go down during normal operation without affecting overall system availability. (即使50%的服务器停止工作也不会影响整个系统的可用性)
  - Improve the system's availability from 98% to 99.999%. (将系统的可用性从98%提高到99.999%)
  - Tenfold increase in the number of bids processed hourly while keeping worst-case response time below 10 seconds. (每小时交易数据成10倍量的增加，但保持最差的响应时间在10秒之内)

输入2：对质量属性的提问

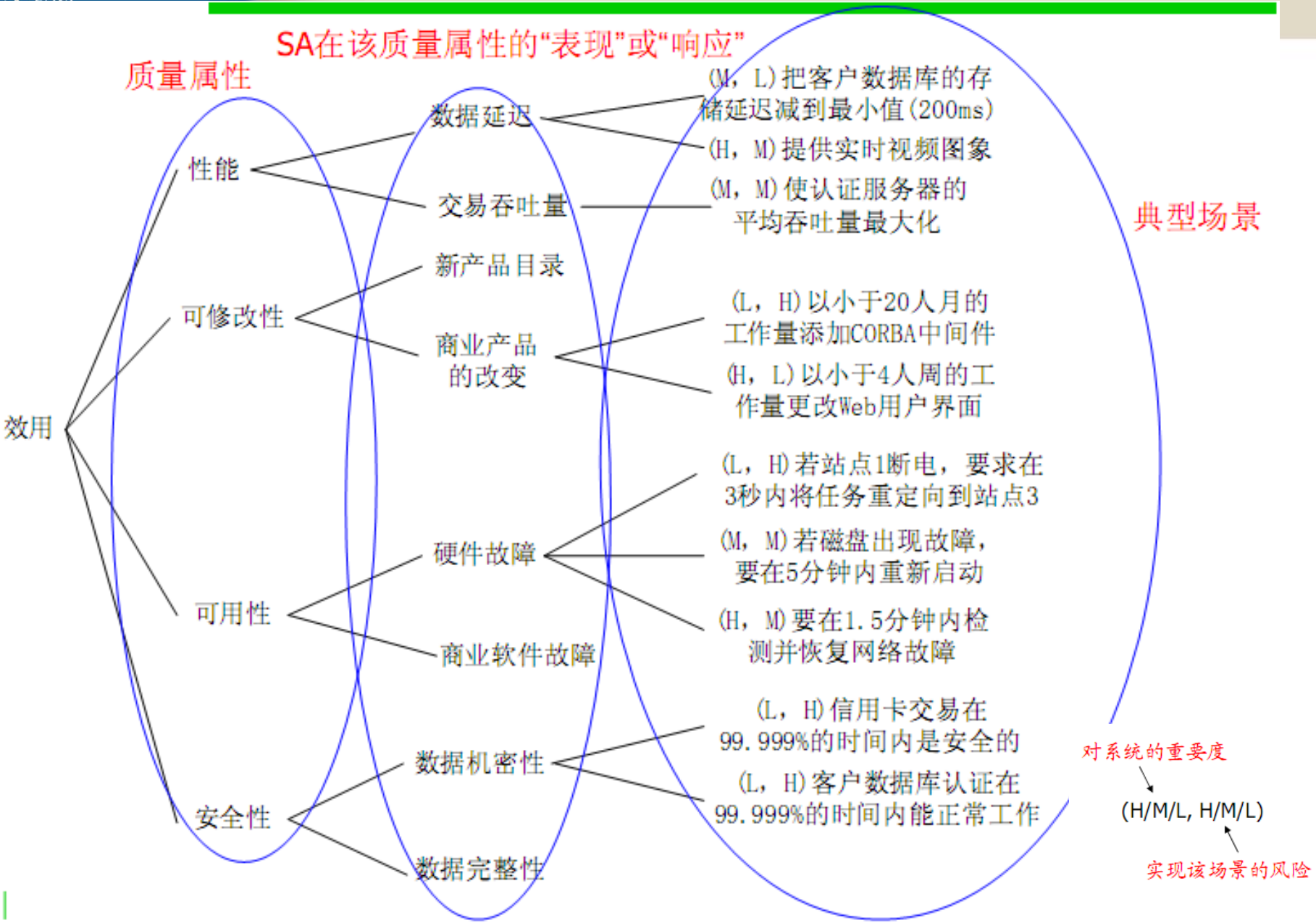
- Quality attribute questions probe styles to elicit architectural decisions which bear on quality attribute requirements. (质询体系结构在特定的质量需求上如何作为)
- Performance (性能)
  - How are priorities assigned to processes? (怎样决定进程的优先级？)
  - What are the message arrival rates? (消息到来的频率是多少？)
- Modifiability (可修改性)
  - Are there any places where layers/facades are circumvented? (是否按层封装功能？)
  - What components rely on detailed knowledge of message formats? (哪些构件依赖消息格式的细节？)

通过“头脑风暴”，尽可能汇总与质量属性相关的各类问题

中间结果：Utility Tree (效用树)

- Utility trees provide a top-down mechanism for directly and efficiently translating the business drivers of a system into concrete quality attribute scenarios. (效用树：提供了一种自顶向下的机制来将系统的业务驱动要素翻译为具体的质量属性及其场景)

- Purpose: provides a prioritized list of scenarios (目的：对场景的优先级进行排序)
  - 可使用各种各样的排序方法，例如：投票


SA在该质量属性的“表现”或“响应”
质量属性
效用
性能
可修改性
可用性
安全性
数据延迟
交易吞吐量
新产品目录
商业产品的改变
硬件故障
商业软件故障
数据机密性
数据完整性
(M，L)把客户数据库的存储延迟减到最小值(200ms)
(H，M)提供实时视频图象
(M，M)使认证服务器的平均吞吐量最大化
(L，H)以小于20人月的工作量添加CORBA中间件
(H，L)以小于4人周的工作量更改Web用户界面
(L，H)若站点1断电，要求在3秒内将任务重定向到站点3
(M，M)若磁盘出现故障，要在5分钟内重新启动
(H，M)要在1.5分钟内检测并恢复网络故障
(L，H)信用卡交易在99.999%的时间内是安全的
(L，H)客户数据库认证在99.999%的时间内能正常工作
典型场景
对系统的重要度
(H/M/L, H/M/L)
实现该场景的风险

输出1：风险和非风险 (Risk and Non-risk)

- Risks are potentially problematic architectural decisions (风险：SA设计中存在潜在问题的决策)
- Non-risks are good decisions relying on implicit assumptions. (非风险：在一个可信的假设之下被认为是“好”的决策)

- Risk and non-risk constituents (风险和非风险需要描述的要素)
  - an architectural decision (已做出或未做出的SA设计决策)
  - a specific quality attribute response (该决策对质量属性造成的后果和影响)
  - a rationale (造成该影响的论据)

“风险”的例子

- SA决策：在三层C/S风格中，第二层业务逻辑的模块划分规则没有明确的给出；
- 可能造成的后果和影响： 这会导致功能的混乱，并可能导致第三层的可修改性较差；
- 判断的依据：如果不将业务逻辑模块划分规则清楚的表述出来，就可能会间接导致构件之间的耦合度增大。

“非风险”的例子

- SA假设：消息的到达速率为1条/秒，消息处理时间<30ms，并存在一个较高优先级的消息处理进程；
- 可能造成的影响：“消息的最长响应时间最长1秒钟”是可行的；
- 判断的依据：消息的到达速率是限定的，高优先级处理进程的“抢占”效果是可知的。

输出2：关键点(Sensitivity point)

- 关键点：SA中对实现特定的质量属性起到关键作用的构件或构件间连接关系；
- 例如：
  - 加密位的多少将对网络的机密性等级起到关键作用；
  - 通讯协议和文件格式的封装程度将决定着系统维护所需的工作量的大小；
  - 把定时方法从一个精确的框架移植到一个不精确的框架，会极大地影响其他模块的正常工作。

输出3：折中点(Tradeoff Point)

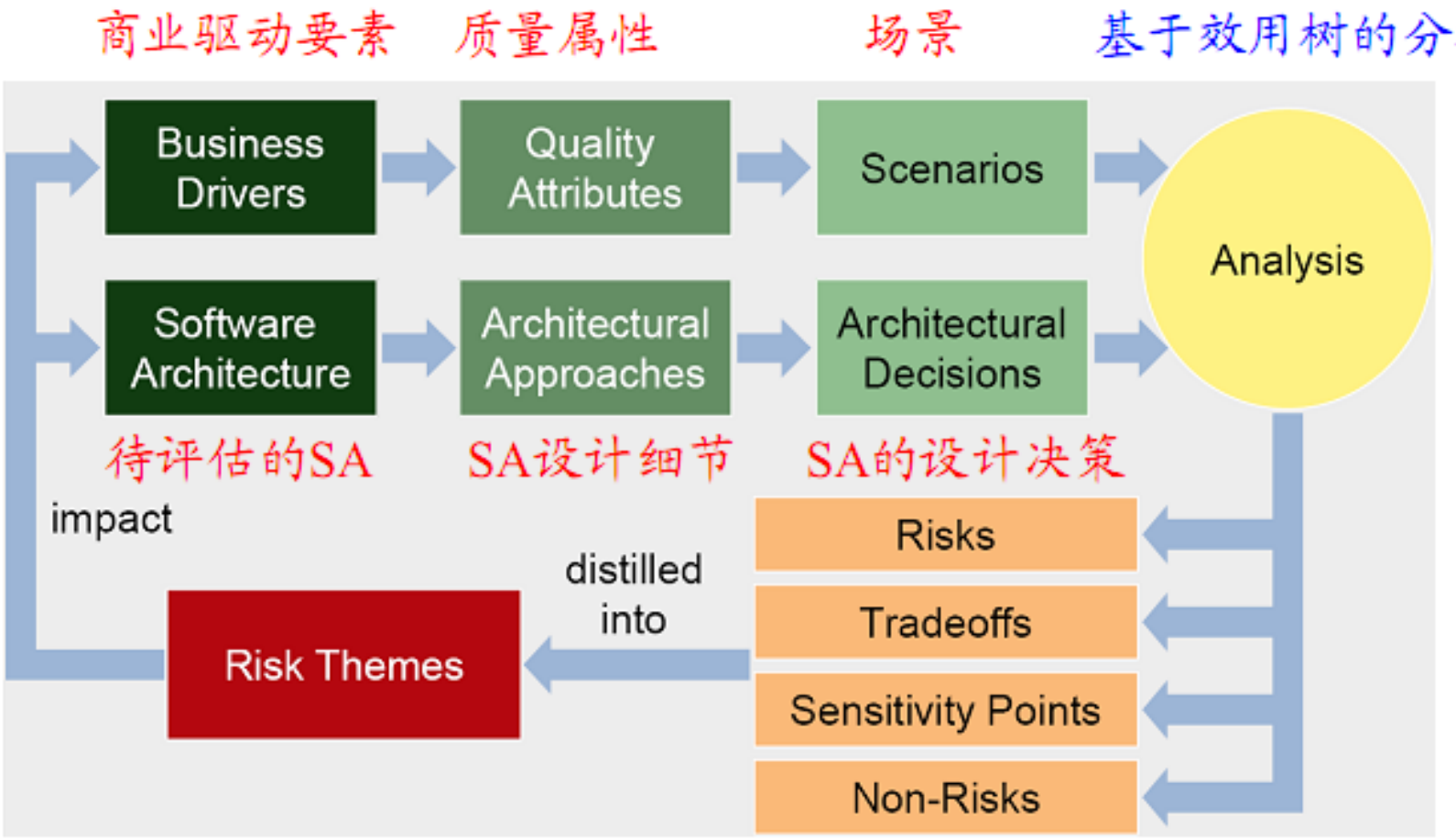
- 折中点：影响到多个质量属性，是多个质量属性的关键点，而且是矛盾的。
- 例如：
  - 改变加密级别会影响到安全性和性能；
  - 提高加密级别会改善安全性，但需要更多的处理时间(性能下降)；
  - 而消息的处理时间不能超过某一范围
  - 因此，“加密级别”就是一个折中点。

折中点的另一个例子

- In order to achieve the required level of performance in the discrete event generation component, assembly language had to be used thereby reducing the portability of this component. (为了达到性能要求，不得不在构件中使用汇编语言，但此构件不再有移植性)

ATAM的执行过程

ATAM的执行过程


商业驱动要素
质量属性
场景
基于效用树的分析
Business Drivers
Quality Attributes
Scenarios
Analysis
Software Architecture
Architectural Approaches
Architectural Decisions
待评估的SA
SA设计细节
SA的设计决策
impact
distilled into
Risk Themes
Risks
Tradeoffs
Sensitivity Points
Non-Risks
输出：风险/非风险/关键点/折中点

ATAM的九大步骤

演示
1. Present the ATAM (介绍ATAM的具体过程)
2. Present business drivers (讲解商业驱动要素)
3. Present architecture (讲解体系结构设计方案)

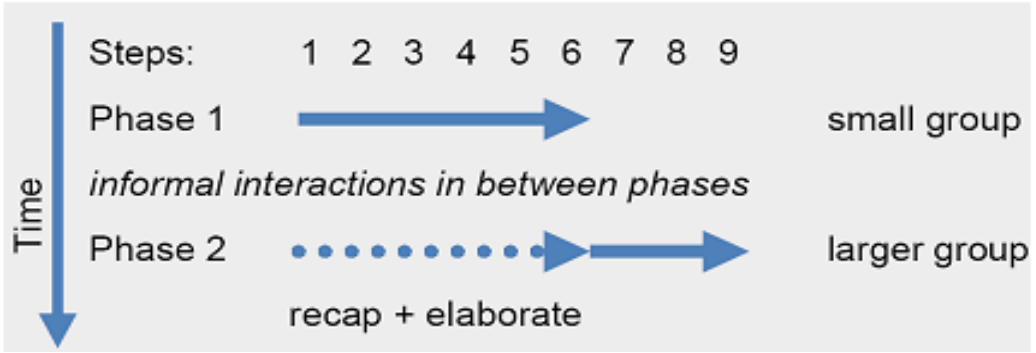
讨论分析
4. Identify architectural approaches (明确体系结构方法)
5. Generate quality attribute utility tree (生成效用树)
6. Analyze architectural approaches (分析体系结构方法)

测试
7. Brainstorm and prioritize scenarios (自由讨论和为场景排序)
8. Analyze architectural approaches (分析体系结构设计)

报告
9. Present results (讲解结论)

ATAM的九大步骤

- ATAM evaluations are often conducted in two stages or phases (ATAM通常被分为两个阶段)



- During phase 1 the architect describes the quality attribute goals and how the architecture meets these goals (在阶段1，架构师描述质量目标，和体系结构如何达到目标)
- During phase 2 we determine if a larger group of stakeholders agrees with the goals and the results (在阶段2，进行具体分析，并确认是否各个角色都同意这些目标和结果)

ATAM的九大步骤

1. Present the ATAM

- Evaluation Team presents an overview of the ATAM including (介绍ATAM的相关步骤、技术和输出):
  - ATAM steps in brief
  - Techniques
  - Outputs

2. Present Business Drivers

- ATAM customer representative describes the system's business drivers including (客户代表描述系统的业务驱动要素):
- Business context for the system (系统的业务环境)
- High-level functional requirements (高层次的功能性需求)
- High-level quality attribute requirements (高层次的质量需求)

3. Present Architecture

- Architect presents an overview of the architecture including (架构师对体系结构的简介):
  - Technical constraints such as an OS, hardware, or middle-ware prescribed for use (技术限制，比如必须要采用的OS、硬件和中间件)
  - Other systems with which the system must interact (其他必须与之交互的系统)
  - Architectural approaches/styles used to address quality attribute requirements (用来满足质量需求的体系结构风格)

3. Present Architecture(Cont.)

- The architect, project manager, and marketing representative need to describe how the system will create value for the organization (架构师、项目经理和市场代表一起来描述此系统如何为客户带来价值)
  - The marketing representative must detail how system responses (functional and quality attribute requirements) map to value. (市场代表必须详细阐述系统的功能和质量需求对市场价值的影响)
  - The project manager must detail how architectural approaches map to cost. (项目经理必须详细阐述体系结构需要的成本)

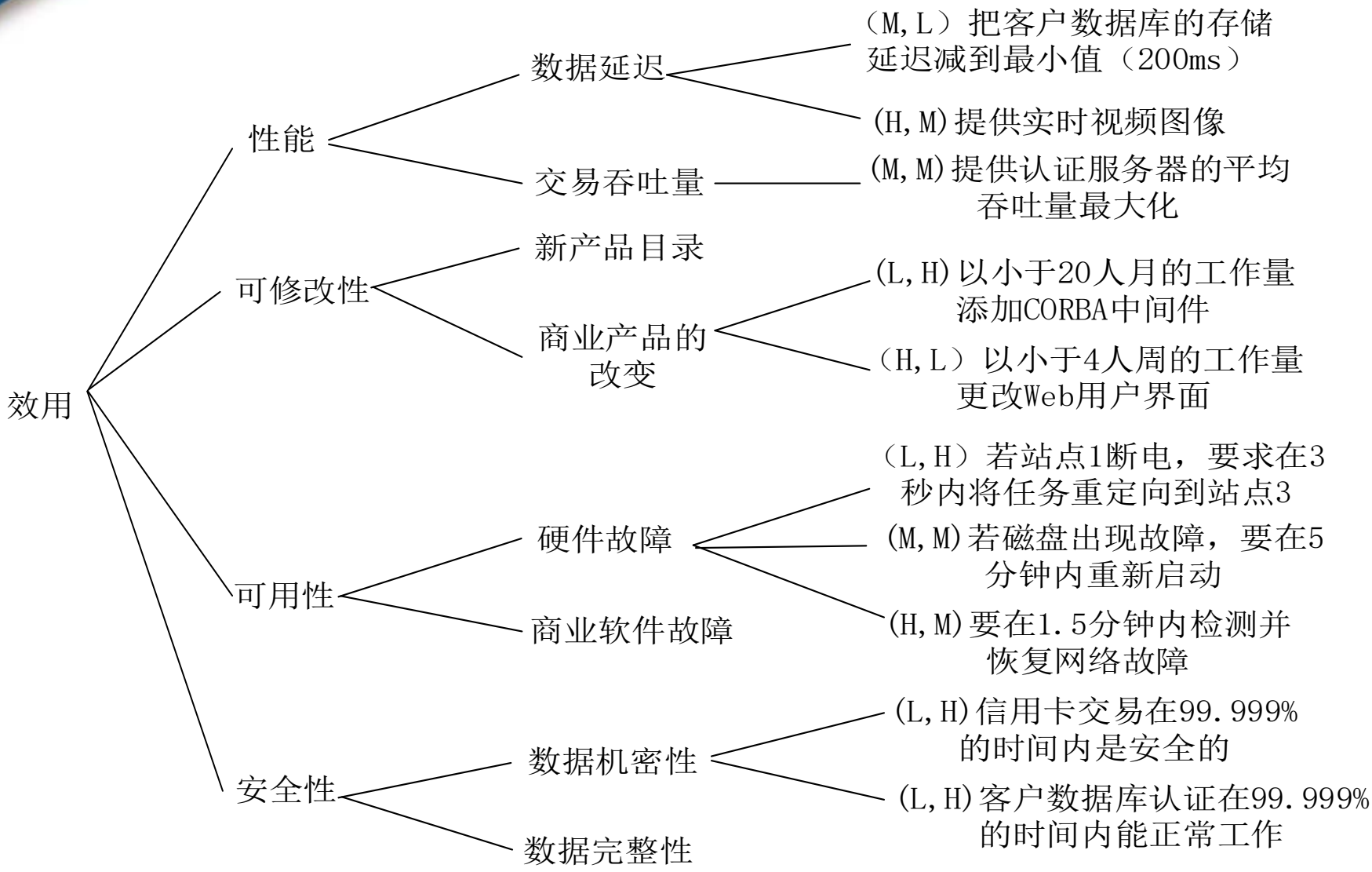
4. Identify Architectural Approaches

- Start to identify places in the architecture that are key for realizing quality attribute goals. (开始确认体系结构中对实现质量需求产生决定作用的部分)
- Identify any predominant architectural approaches. (明确主要的体系结构方法)
- Examples:
  - client-server
  - 3-tier
  - publish-subscribe
  - redundant hardware

5. Generate Quality Attribute Utility Tree

- Identify, prioritize, and refine the most important quality attribute goals by building a utility tree. (通过建立一个效用树，来明确、排序和精炼大部分的质量目标)
- Output: a characterization and a prioritization of specific quality attribute requirements. (输出：质量需求的描述和优先级)

Utility Tree Construction & Prioritization



6. Elicit/Analyze Architecture Approaches

- Motivated by the high priority leaves of the utility tree, the Evaluation Team probes the architecture approaches. (效用树中高优先级的叶子促进评审组探查体系结构)
  - Identify the approaches which pertain to the highest priority quality attribute requirements (确认可以满足高优先级的质量需求的方法)
  - Generate quality-attribute specific questions for highest priority quality attribute requirement (为高优先级的质量需求制定关于质量的问题)
  - Ask quality-attribute specific questions (询问这些问题)
  - Identify and record risks and non-risks (确认和记录风险和非风险，关键点和折中)

Example Approach Elicitation

- Scenario: Detect and recover from HW failure of main switch (场景：检测主交换机的硬件故障，并恢复)
- Stimulus: CPU failure
- Response: 0.999999 availability of switch

Architectural Approaches:	R	S	T

R 有风险决策
S 敏感点
T 权衡点

Example Approach Elicitation

- Scenario: Detect and recover from HW failure of main switch (场景：检测主交换机的硬件故障，并恢复)
- Stimulus: CPU failure
- Response: 0.999999 availability of switch

Architectural Approaches:	R	S	T

- Failover Rerouting
- Heartbeat
- Watchdog
- Backup Data Channel
- Backup CPU(s)

Example Approach Elicitation

- Scenario: Detect and recover from HW failure of main switch (场景：检测主交换机的硬件故障，并恢复)
- Stimulus: CPU failure
- Response: 0.999999 availability of switch

Architectural Approaches:	R	S	T
• Failover Rerouting	×	×	
• Heartbeat	×	×	×
• Watchdog		×	
• Backup Data Channel		×	
• Backup CPU(s)	×	×	

7. Brainstorm and Prioritize Scenarios

- Stakeholders have brainstormed a large set of scenarios. (各个涉众已经讨论出很多很多场景)
- Each stakeholder is allocated a number of votes roughly equal to 0.3 x #scenarios (每个涉众分给一个投票数，其值大约为(0.3 x 场景数))
- Prioritized scenarios are compared with the utility tree and differences are reconciled. (排好次序的场景和效用树进行比较。如果有不同的，则要再次考量，达成一致)

7b. Prioritize Scenarios

场景编号	场景描述	得票数量
4	在 10 分钟内动态地对某次任务得重新安排	28
27	把对一组车辆得管理分配给多个控制站点	26
10	在不重新启动系统的情况下，改变已开始任务的分析工具	23
12	在发出指令后 10 秒内，完成对不同车辆的重新分配，以处理紧急情况	13
14	在 6 人月内将数据分配机制从 CORBA 改变为新兴的标准	12

场景编号	得票数量	质量属性
4	28	性能
27	26	性能、可修改性、可用性
10	23	可修改性
12	13	性能
14	12	可修改性

8. Analyze Architectural Approaches

- Identify the architectural approaches impacted by the scenarios generated in the previous step. (确定被上一步产生的场景影响的体系结构设计)
- This step continues the analysis started in step 6 using the new scenarios. (用第6步同样的方法来分析新的场景)
- Continue identifying risks and non-risks. (继续确认风险和非风险)
- Continue annotating architectural information. (继续标注体系结构信息)

9. Present Results

- Recapitulate steps of the ATAM (总结所有ATAM的步骤)
- Present ATAM outputs (输出结果)
  - architectural approaches
  - utility tree
  - scenarios
  - risks and “non-risks”
  - sensitivity points and tradeoffs
- Offer recommendations (推荐方案)

各步骤中相关的风险承担者

步骤编号	所做的工作	风险承担者群体
1	描述 ATAM 方法	评估小组/客户代表/体系结构设计小组
2	描述商业动机	评估小组/客户代表/体系结构设计小组
3	描述体系结构	评估小组/客户代表/体系结构设计小组
4	确定体系结构方法	评估小组/客户代表/体系结构设计小组
5	生成质量属性效用树	评估小组/客户代表/体系结构设计小组
6	分析体系结构方法	评估小组/客户代表/体系结构设计小组
7	讨论和对场景进行分级	所有风险承担者
8	分析体系结构方法	评估小组/客户代表/体系结构设计小组
9	描述评估结果	所有风险承担者