Ch14 Quality Attribute Modeling and Analysis 质量属性模型和分析

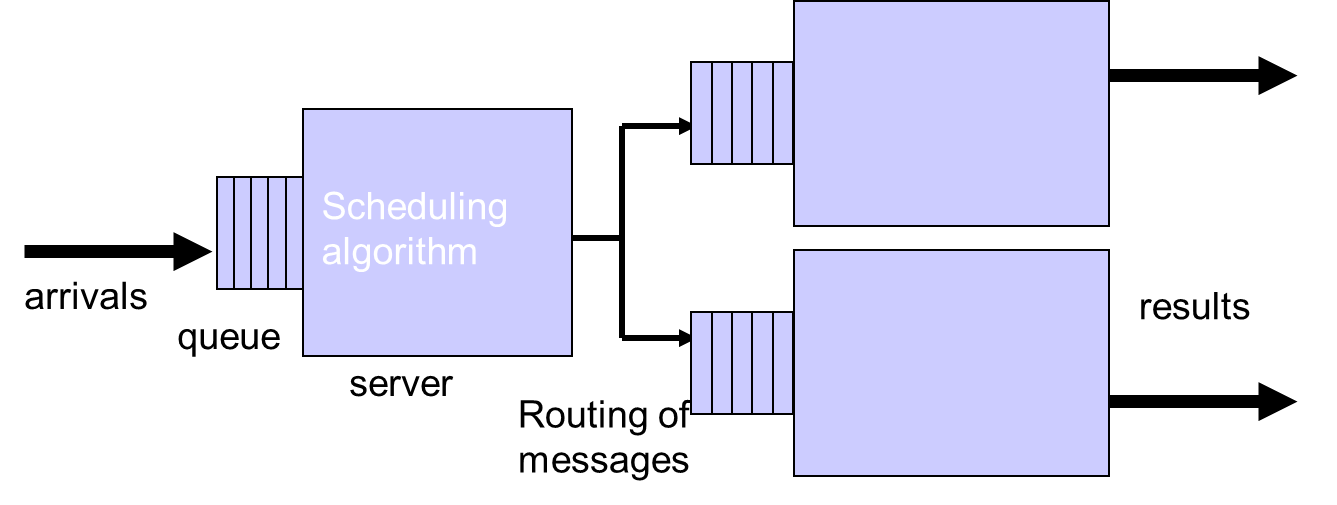
ppt内容很多，主要看红色标题的部分。

建模架构以启用质量属性分析

一些质量属性经过充分理解，经过时间考验的分析模型，可以用于协助分析。

分析模型可以支持属性分析。

Performance Models

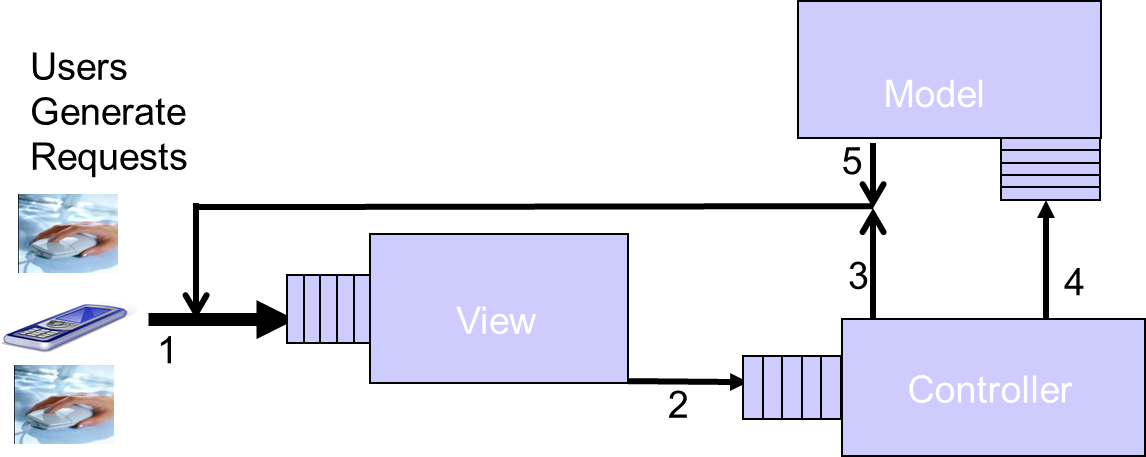


参数：事件到达率、选择的排队规则、选择的调度算法、事件的服务事件、网络拓扑、网络带宽、选择的路由算法

Allocation Model for MVC



Queuing Model for MVC



1. 到达
2. View向Controller发送请求
3. 操作返回到View
4. 操作返回到模型
5. 模型向View发送操作

参数：

从系统外部到达的频率

在视图队列中使用的排队规则

在视图中处理消息的时间

视图发送到控制器的消息的数量和大小

连接视图和控制器的网络带宽

控制器使用的排队规则

处理控制器内消息的时间

控制器发送回视图的消息数量和大小

从控制器到视图的网络带宽

控制器发送给模型的消息数量和大小

模型使用的排队规则

处理模型中的消息的时间

模型发送到视图的消息的数量和大小

连接模型和视图的网络带宽

属性建模的成本/收益

成本：确定前面提到的参数

收益：可以估计延迟

估计参数更准确，延迟的预测越好。

当延迟很重要且值得怀疑时，这是值得的。

当显然有足够的容量来满足需求时，这是不值得的。

可用性建模 Availability Modeling

确定组件的故障率和恢复时间。

假设我们希望通过应用冗余策略来提高使用Broker模式的系统的可用性。

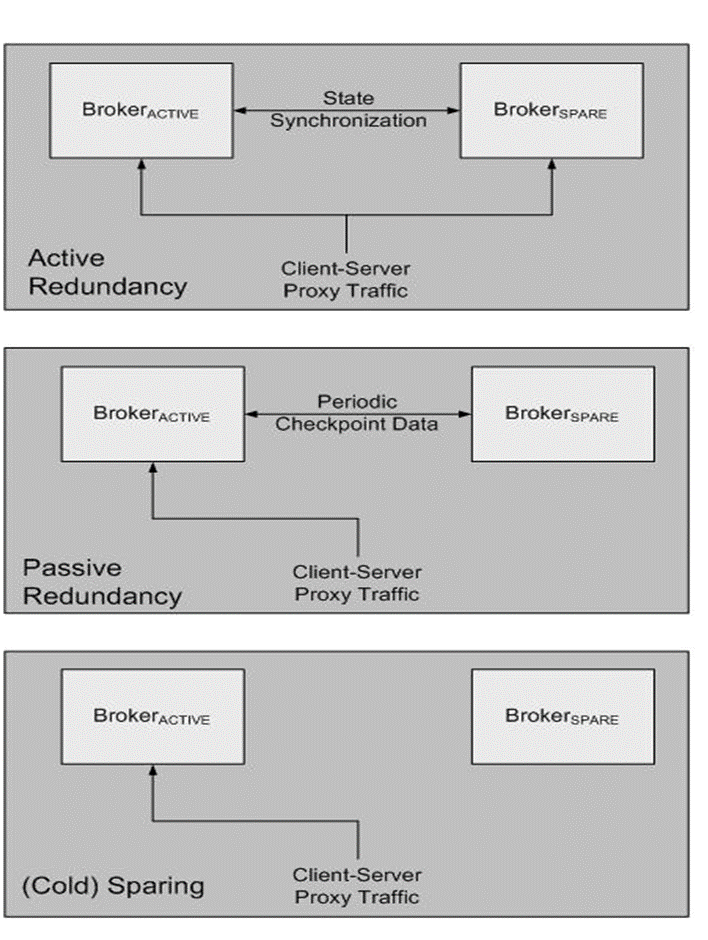
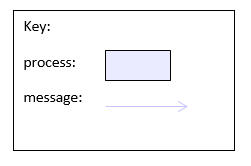
三种增加可用性的broker策略

主动冗余（热备份）(hot spare)

被动冗余（暖备）(warm spare)

备用（冷备用）(cold spare)

让Broker更可用

将概率应用于战术

使用概率模拟不同的策略

* A、B是独立的，P(A or B) = P(A)+ P(B).
* A、B是独立的，P(A and B) = P(A) • P(B).
* A、B是依赖的，P(A or B) = P(A)+ P(B) - P(A and B)
* A、B是依赖的，P(A and B) = P(A) • P(B|A)

可用性的定义

系统的可用性是在需要时它可以运行的概率。 这通常定义为

α=

在计算可用性时通常不考虑计划的停机时间（即，停止服务）。

被动冗余

* + - 假设

组件（主要或备份）的故障与其对应方的故障无关

假设两者的故障概率相同：P（F）

* + - 还可以根据其他策略估计故障的概率。
    - 然后考虑到实施适当策略的成本，我们可以进行成本/收益分析。

【了解】常见质量属性模型的成熟度 Maturity of Quality Attribute Models

|  |  |  |
| --- | --- | --- |
| **质量属性** | **知识基础** | **成熟度/差距** |
| **可用性** | Markov 模型; Statistical(统计) 模型 | 硬件可靠性领域比较成熟，软件领域不太成熟。需要与状态。需要与状态恢复相关的模型以及可将故障百分比归因于软件的模型。 |
| **互操作性** | Conceptual framework  (概念框架) | 低成熟度；模型需要大量的人工解释和输入。 |
| **可修改性** | Coupling and cohesion metrics(耦合类聚指标); 成本(Cost)模型 | 在学术界有大量研究；在现实环境中仍然需要更多的经验支持。 |
| **性能** | Queuing theory(排队理论); Real time scheduling theory(实时调度理论) | 高成熟度；需要大量的教育和训练才能正确使用 |
| **安全性** | 没有架构模型 |  |
| **可测试性** | Component Interaction Metrics(组件交互度量标准) | 低成熟度；很少的经验验证 |
| **易用性** | 没有架构模型 |  |

质量属性清单Checklist

质量属性清淡提供了一种方法：

* 检查需求。这些需求是否不活了特定质量属性的所有细微差别？
* 审计(Auditing)。设计是否满足证明过程所需的所有方面。

安全清单

安全清单很常见。

* 接受信用卡的供应商应符合PCI（个人信用信息）标准
* 电力生产商拥有安全检查清单，以防止对关键基础设施的攻击

清单包括：

产品要求。 例如：PCI检查表指出信用卡背面的安全代码永远不应存储，信用卡#只能以加密形式存储。

流程要求。 例如：补丁应该及时应用，并且应该有一个组织有责任确保它们的人。

【了解】思想实验 Thought Experiments

概念：思想实验在脑海中或口头上通过特定场景进行。

* 通常被架构师中用来在设计中探索替代方案。
* 也会在评估/记录期间使用。

步骤：

1. 列举用例的步骤
2. 在每一步，问自己

* 哪种机制被用来实现哪种特定的质量要求？
* 这种机制是否会阻碍其他质量属性要求的实现？

1. 为以后更深入的分析或原型构建记录问题

【了解】粗略分析 Back-of-the-Envelope Analysis

* 分析不需要精确详细
* 只对有问题的地方或者重要需求进行更深入的分析。

【了解】实验、模拟仿真和原型Experiments, Simulations, and Prototypes

1. 许多工具可以帮助进行试验以确定设计的行为

* 请求生成器可以创建合成负载以测试可伸缩性
* 监视器可以执行非侵入式资源使用检测

1. 依赖于部分或原型实现

* 原型的选择是最重要的选择
* 故障注入工具可以引发故障以确定系统在故障条件下的响应

模拟

存在基于事件的模拟器，其可用于模拟各种负载下的系统行为

* 必须创建模拟仿真
* 必须有各种不同的负载和响应来检查

需求和设计期间的分析

* 在生命周期的不同阶段进行不同类型的分析
* 需求：
* 分析模型/粗略分析可以帮助进行容量规划
* 检查表可以帮助确保捕获正确的需求
* 分析
* 原型可以帮助探索设计选项
* 分析模型或模拟仿真可以帮助理解潜在的瓶颈

实施或部署期间的分析

* 可以在实施过程中或在部署之后使用实验和合成负载(synthetic load)测试
* 可以在部署后使用监视器(Monitors)来确定实际行为并找到瓶颈。

生命周期不同阶段的分析

|  |  |  |  |
| --- | --- | --- | --- |
| **Life-Cycle Stage** | **Form of Analysis** | **Cost** | **Confidence** |
| 需求 | 基于经验的类比 | Low | Low-High |
| 需求 | 粗略分析 | Low | Low-Medium |
| 架构 | 思想实验 | Low | Low-Medium |
| 架构 | 清单 | Low | Medium |
| 架构 | 分析模型 | Low-Medium | Medium |
| 架构 | 仿真 | Medium | Medium |
| 架构 | 原型 | Medium | Medium-High |
| 实现 | 实验 | Medium-High | Medium-High |
| 部署系统 | Instrumentation(仪器) | Medium-High | High |