Chapter 08 Performance

思路：什么是性能？它的通用场景是什么？我们如何对其进行建模分析？它有什么策略？

性能

定义：软件系统满足与时间相关需求的能力，更通俗地讲即系统准时响应用户从其他系统发出的信息/中断/请求等的能力

通用场景

- Source：系统内部或外部
- Stimulus：周期性、间歇性或者随机事件的到来
- Artifact：系统中的一个或者多个组件
- Environment：正常、紧急、峰值负荷、超负荷
- Response：处理事件、更改服务的水平
- Response Measure：延迟、截止日期、吞吐量、抖动、漏率

性能建模

- 思路：性能是与时间相关的，我们可以通过测量时间因素来分析性能的好坏
- 响应时间：系统响应用户发出的信息/中断/请求所需要的时间
  - 组成：处理时间+阻塞时间
    - 处理时间：系统为响应请求而执行操作的时间
    - 阻塞时间：系统无法响应的时间
      - 原因：资源竞争，资源不可用，依赖于其他计算
      - 资源又可以分为硬件资源+软件资源

策略

- 目标：在时间约束内生成对事件的响应，注意这里的时间可以是单个或者流式的，且是触发器
- 类别：
  - 控制资源需求：一般降低需求，在需求侧操作
  - 管理资源：提高资源的利用效率，在响应侧操作

策略树

- 输入：事件到达
- 输出：在时间限制内生成回复
- 控制资源需求
  - 管理采样率
    - 降低数据流采样的频率，以减少不必要的数据处理和系统负载
  - 优先级事件
    - 制定优先级方案，根据事件的重要性对其进行排序。当系统资源不足时，忽略低优先级事件。
  - 减少开销
    - 在**可修改性**（系统灵活性）和**性能**（事件处理效率）之间存在权衡
  - 限制执行时间
    - 限制响应事件所用的执行时间，防止长期运行的任务阻塞系统
  - 提高资源的利用率
    - 改进关键领域中使用的算法将减少延迟，并提高系统的整体资源效率
- 分类
  - 管理资源
    - 增加资源
    - 引入并发
      - 通过并行处理任务来减少阻塞时间，从而提高系统整体性能
    - 保持多个计算副本
      - 减少单个服务器上的资源争用
    - 保持多个数据副本
      - 数据缓存
        - 内存访问比磁盘访问快得多，因此频繁使用的数据可以缓存到内存中，以减少访问时间
        - 本地访问（来自同一台机器）比远程通过网络访问数据更快，因此将频繁使用的数据缓存到本地可以减少网络延迟。
      - 数据复制
        - 减少多次并发访问时的竞争，从而提高可用性和性能。
    - 限制队列大小
    - 调度资源