

内容提要

- 讨论

讨论

- 您的组织现在是否在应用自动化验收测试？
- 现在效果如何？
- 遇到哪些问题？

内容提要

- 讨论
- 正确的质量观

- **质量是开发人员的神圣责任，而不仅仅是测试人员的责任**

-The burden of quality is on the shoulders of those writing the code. Quality is never “some tester’ s” problem.

- **只有将开发和测试完全地混合在一起，不分彼此，才能够真正获得好的质量**

-Quality is achieved by putting development and testing into a blender and mixing them until one is indistinguishable from the other.

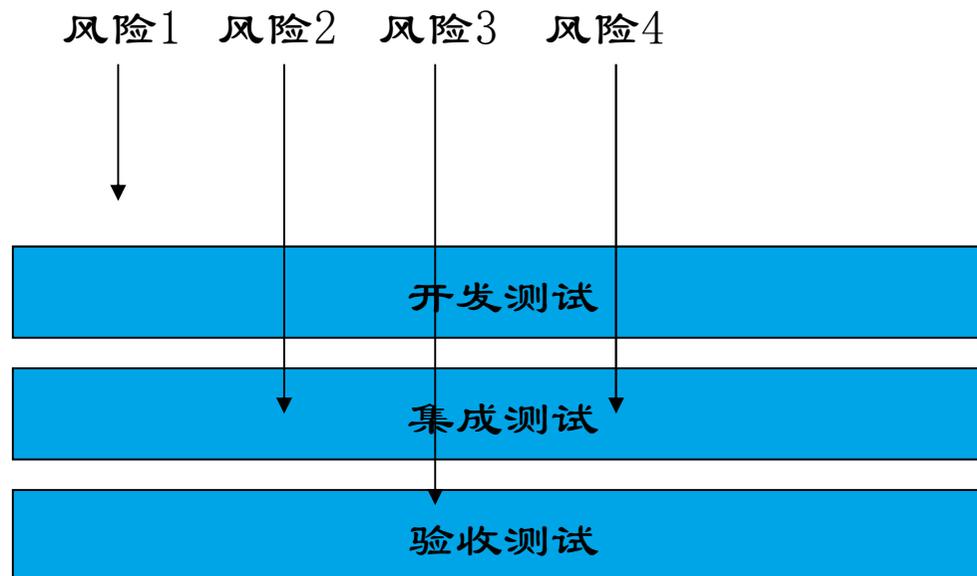
内容提要

- 讨论
- 正确的质量观
- 什么是分层测试

分层测试的基本思想 - 净水器模型

- 为什么要分层测试？

- 分层才能保证快速反馈，而不是都等到最后才反馈
- 恰当的分层测试可以降低总测试成本



内容提要

- 正确的质量观
- 什么是分层测试
- 分层测试案例一 — 谷歌如何分层测试

测试类型

- **小测试**

- Small tests cover a single code unit in a completely faked environment.

- **中测试**

- Medium tests cover multiple and interacting code units in a faked or real environment.

- **大测试**

- Large tests cover any number of code units in the actual production environment with real resources.

Table 2.1. Goals and Limits of Test Execution Time by Test Size

	Small Tests	Medium Tests	Large Tests	Enormous Tests
Time Goals (per method)	Execute in less than 100 ms	Execute in less than 1 sec	Execute as quickly as possible	Execute as quickly as possible
Time Limits Enforced	Kill small test targets after 1 minute	Kill medium test targets after 5 minutes	Kill large test targets after 15 minutes	Kill enormous test targets after 1 hour

Table 2.2. Resource Usage by Test Size

Resource	Large	Medium	Small
Network Services (Opens a Socket)	Yes	localhost only	Mocked
Database	Yes	Yes	Mocked
File System Access	Yes	Yes	Mocked
Access to User-Facing Systems	Yes	Discouraged	Mocked
Invoke Syscalls	Yes	Discouraged	No
Multiple Threads	Yes	Yes	Discouraged
Sleep Statements	Yes	Yes	No

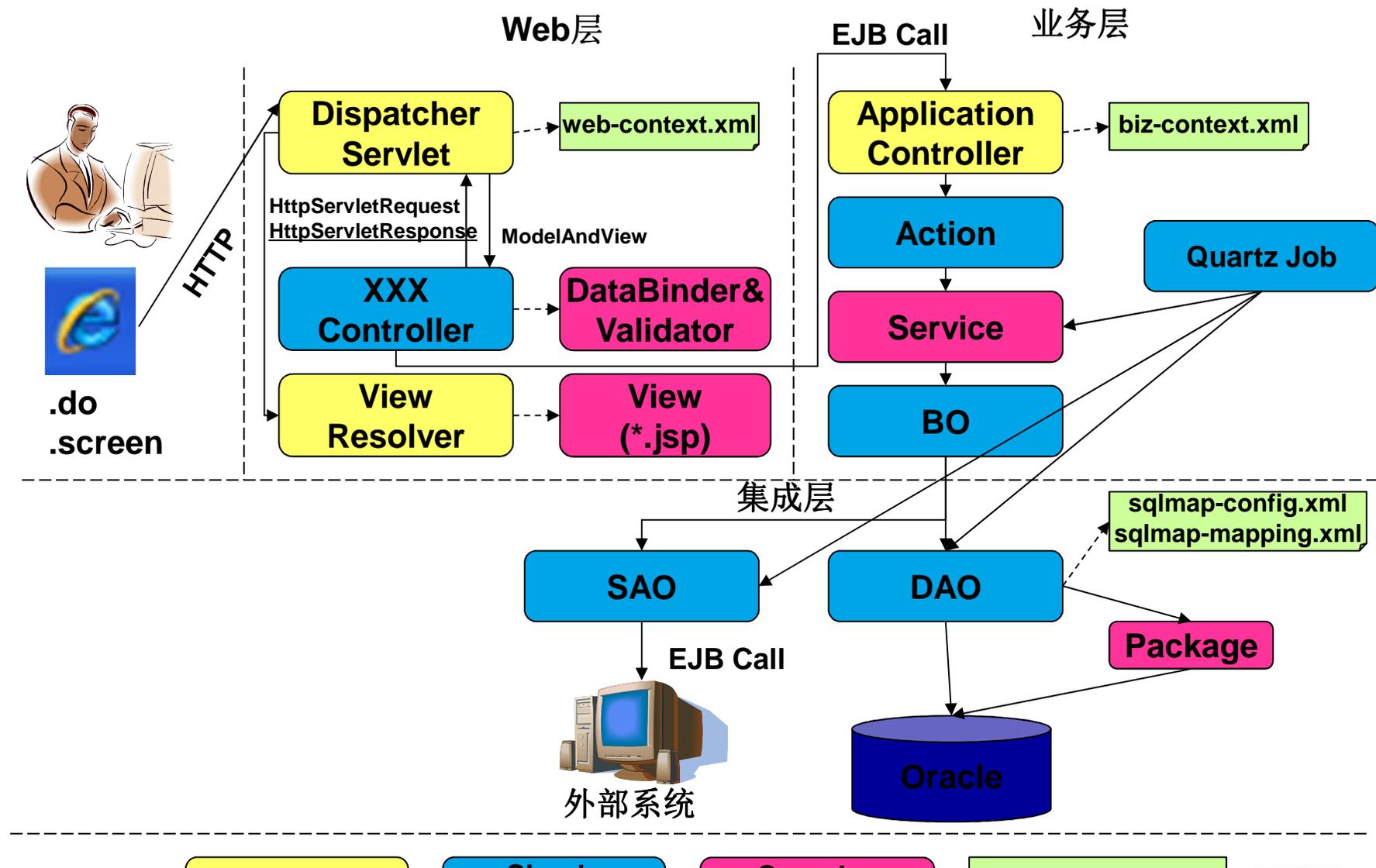
测试类型

- **谷歌采用70/20/10原则：70% 小，20% 中，10% 大**
 - Projects at Google are encouraged to maintain a healthy mixture of test sizes among their various test suites.
 - Overinvesting in end-to-end automation often ties you to a product's specific design

内容提要

- 正确的质量观
- 什么是分层测试
- 分层测试案例一 – 谷歌如何分层测试
- 分层测试案例二 – 企业信息系统如何测试

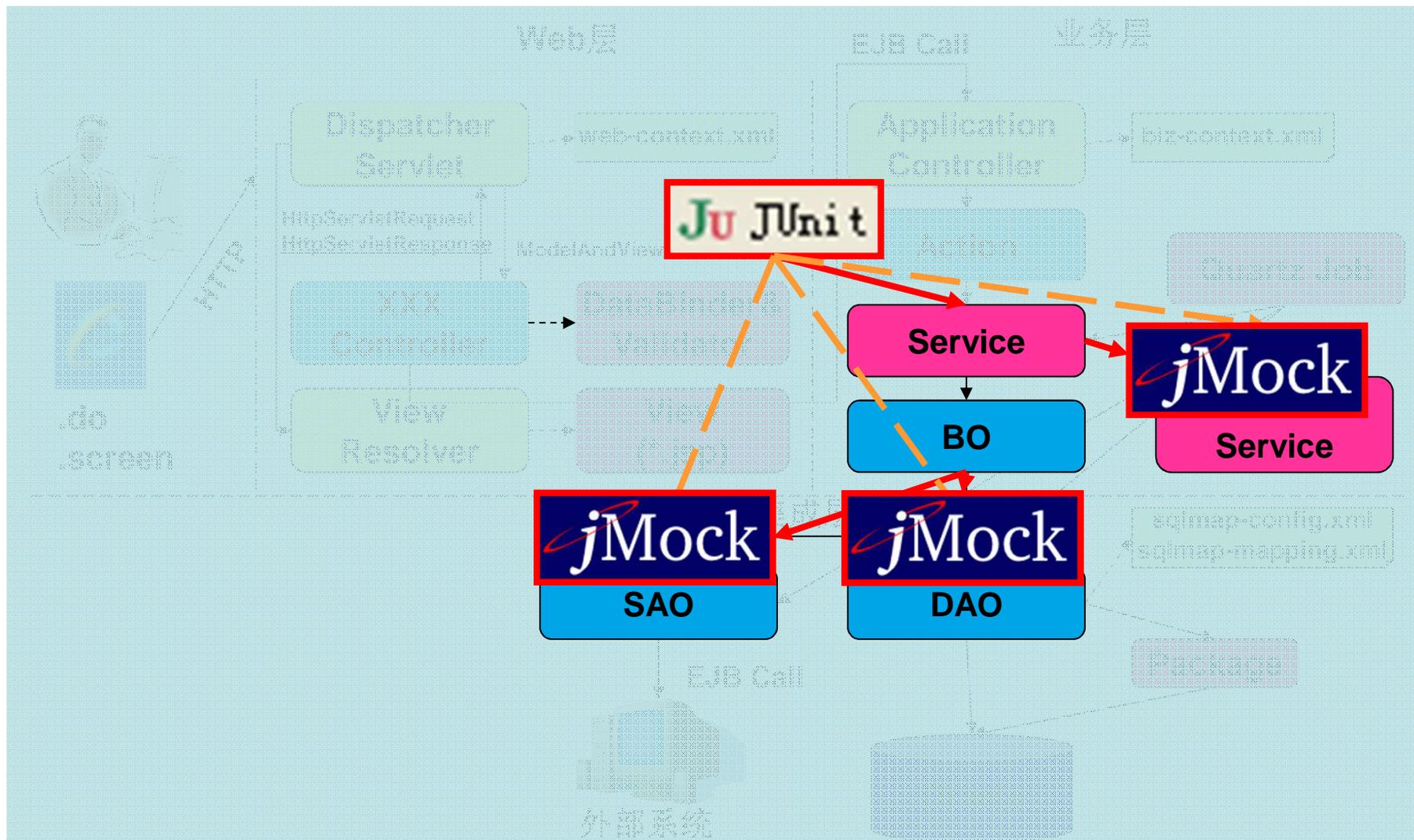
风险分析



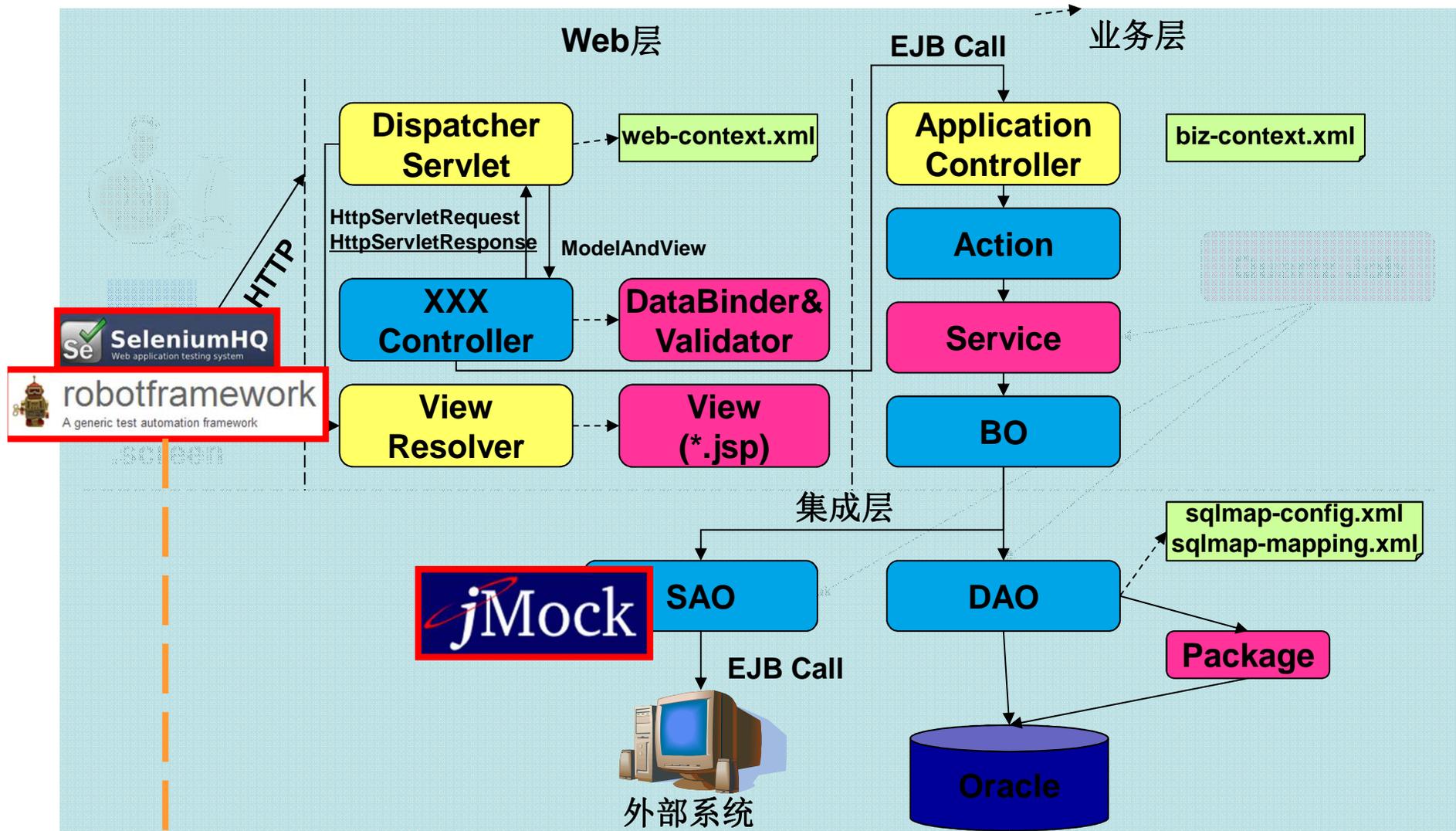
分层测试策略 (例子)

	开发者测试 (DT)	单领域测试 (IDIT)	集成测试 (AT)
测试粒度	Service, Package, Job, ...	单个子系统	多个子系统
关注的风险	一个模块内的逻辑功能错误	单个子系统的功能是否正确	多个子系统的集成问题
执行环境	开发者本机	开发集成测试环境	Staging环境
测试类型	Out-Container	In-Container	In-Container
数据库	N/A	测试数据	类生产数据

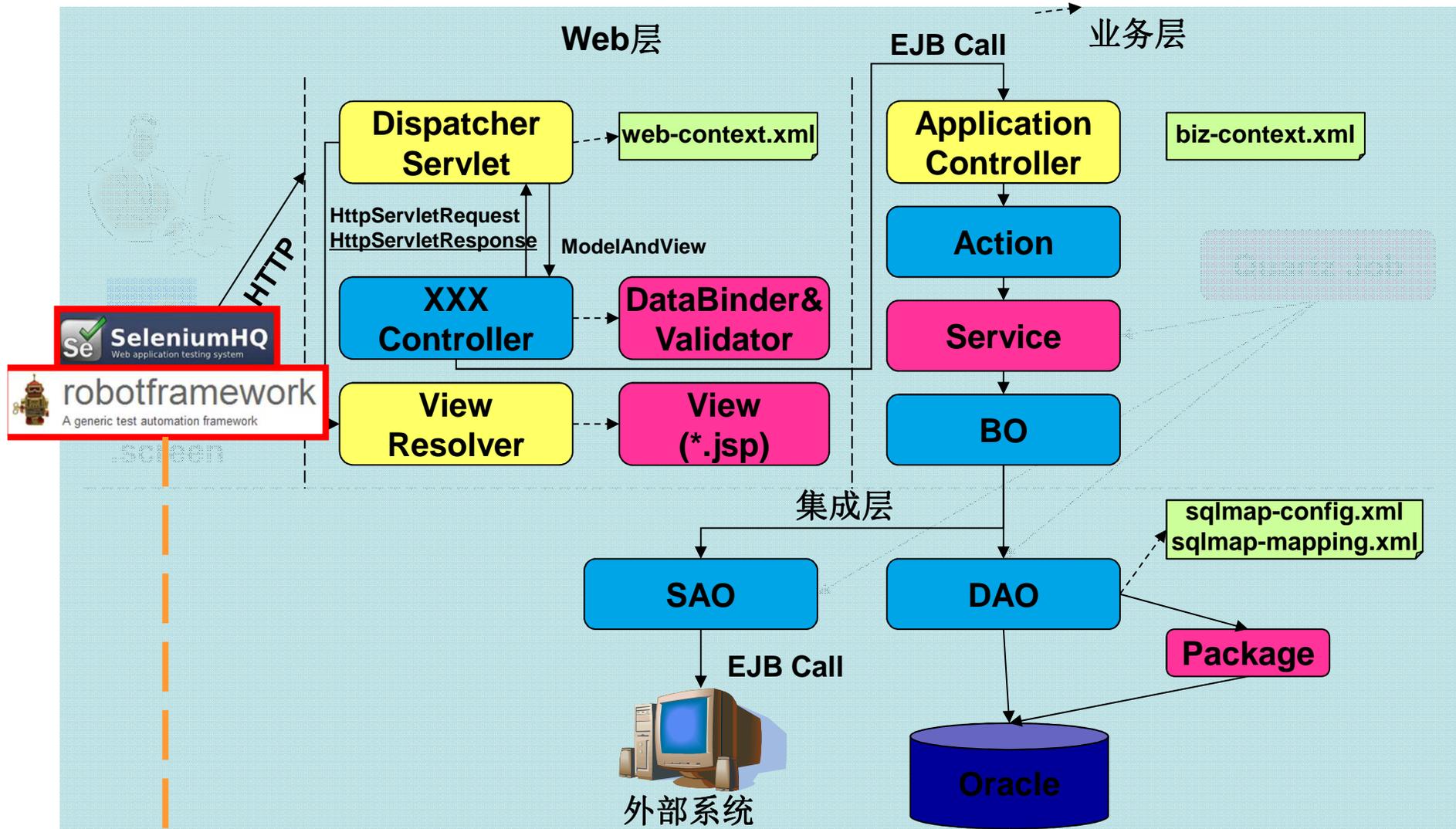
针对Service的开发者测试 (方案一、MockDB方案)



单领域验收测试



跨领域验收测试



内容提要

- **正确的质量观**
- **什么是分层测试**
- **分层测试案例一 - 谷歌如何分层测试**
- **分层测试案例二 - 企业信息系统如何测试**
- **分层测试中的关键技术难点**
 - 开发者测试当中的难点

开发者测试的难点

- **开发者测试的难点在于测什么，其次才是怎么测？**
 - 问题1：什么是合适的被测单元？
 - 问题2：如何找到合适的被测单元？
 - 问题3：如何提高被测单元的可测试性？
 - 问题4：何时停止开发者测试？

单领域验收测试的难点

- **单领域验收测试的难点在于：**
 - 问题1：如何编写出可维护的测试用例？
 - 问题2：如何隔离外部系统？
 - 问题3：如何控制数据库输入？
 - 问题4：何时结束单领域验收测试？