

酒店预订系统单元测试与集成测试回顾

Winter Studio



2016-12-18

南京大学软件学院

朱润之 杨凯 徐天泽 张磊

# 1.单元测试回顾

## 1.1 概述

整体上使用了测试驱动开发的思想指导开发，对于涉及到数据库的data层以及逻辑复杂的bl层，在构造阶段的董事边写了junit测试用例，并编写了mock object隔离外界环境对于此类的影响，而后进行的开发。测试方法采用随机测试和边界值测试，大部分使用黑盒测试，对于部分较为复杂的逻辑使用了白盒测试

## 1.2 测试用例的有效性分析

测试用例在前期编写的时候较为顺利，但是由于前期是基于mock的简单测试，层与层之间的交互只有接口而没有具体的实际逻辑，导致后面的测试越来越艰难，这点值得反思，反应了对于mock的设计不实际，不周到。且测试后期分离了data和bl层，使用rmi进行远程连接，网络异常导致了许多测试的失败，应该提前开发rmi，在rmi的基础上进行开发和测试会更好一些

## 1.3 采取的弥补手段

对于涉及到rmi的测试，首先通过在服务器端的测试，以及使用mock完成在客户端的测试，接着使用简单数据测试通信的稳定性，最后再进行统一的单元测试

## 1.4 度量数据

单元测试的主要测试部分为bl层和data层，对于每个模块的覆盖率应该都达到了70%以上

# 2.集成测试回顾

## 2.1 概述

在体系结构设计阶段结束的时候，为每一层开发了Stub和上下层开发了driver，在构造阶段采取了自底而上的方法，从data层开始开发，没开发一个层次，都是用上一个层次的driver进行maven builder测试，同时通过jenkins进行自动化的集成测试

## 2.2 测试用例的有效性分析

总体上来说，前期开发的stub与driver基本上被舍弃，由于前期对系统的了解不够深入，导致大部分测试都被抛弃，层与层之间接口的大量改变，导致了测试失败，所以前期对于接口的编写是至关重要的。前中期一直使用jenkins进行自动化构建测试，达到了较好的集成测试效果，但在后期由于对fx以及rmi的引入，导致自动化构建难以成功，对于自动化中的引用包的问题要注意

## 2.3 采取的弥补手段

在详细设计之后，重新编写定义了大部分接口，同时更改了集成测试代码，在jenkins失效之后，频繁的使用手动构建进行测试

## 2.4 度量数据

集成测试的覆盖掉应该达到了90%以上。

### 3 小结

无论是集成测试还是单元测试，对于前期体系结构的确立以及接口的编写都有着很强的要求，但我们在这两方面做得不足，导致前期很多工作后期需要重做。能够使用工具进行集成测试是一个优点，但对于工具的使用达不到熟练的程度，导致后期测试无法进行。要正确的选择先进而易用的工具