学 院：南京大学软件学院

团 队：第5组

成 员：陈书玉 陈俐俐 程芸菲 陈悦

完成日期：2016/12/18

酒店预订系统

单元测试及集成测试的回顾

目录

[更新历史 3](#_Toc10549)

[1.单元测试回顾 4](#_Toc647)

[1.1概述 4](#_Toc10115)

[1.2测试用例的有效性分析 4](#_Toc26088)

[1.3采取的弥补手段 4](#_Toc8733)

[1.4度量数据 4](#_Toc18689)

[1.5小结 4](#_Toc15551)

[2.集成测试回顾 6](#_Toc30583)

[2.1概述 6](#_Toc22504)

[2.2测试用例的有效性分析 6](#_Toc19831)

[2.3采取的弥补手段 6](#_Toc3607)

[2.4度量数据 6](#_Toc30335)

[2.5小结 6](#_Toc21473)

# 更新历史

|  |  |  |  |
| --- | --- | --- | --- |
| **修改人员** | **修改日期** | **修改原因** | **版本号** |
| 陈俐俐 | 2016/12/11 | 创建文档模版 | V0.1 |
| 陈俐俐 | 2016/12/18 | 完成文档 | V1.0 |

# 1.单元测试回顾

## 1.1概述

大体上使用了TestDriver的思想指导开发，在构造阶段，对于有较为复杂的逻辑处理和外部内部接口的类，先编写JUnit测试用例。并且编写了mock object以隔离外界环境对此类的影响，而后进行开发，开发完成之后，先进行JUnit自动化测试，通过之后提交。测试的方法包括随机测试和边界值测试，部分复杂逻辑采用了白盒测试中的语句覆盖测试。

## 1.2测试用例的有效性分析

测试用例大多是两个思路，一个是测试主干正常执行，另一个是边界值的专门测试。测试结果比较有效，能够找到一些比较隐蔽的错误。

但是由于前期所写的单元测试只涉及部分复杂的逻辑结构，且对应要用的mock object 写的也比较少，这就导致了后期测试的困难。并且一开始考虑的不是很周到，导致前期写的一些测试到构造后期无法使用。实际上在真正构思好了属性和方法接口后再去开发 mock object比较合适。另外，所做的单元测试只是做到了代码覆盖，可能逻辑组合是会出问题。

## 1.3采取的弥补手段

对于一些复杂的逻辑结构，在后期进行了代码评审并进行了路径覆盖测试。在构造后期对复杂的逻辑结构重写了能够使用的单元测试。

## 1.4度量数据

**单元测试覆盖度：**

使用 工具进行度量。由于主要测试的部分为bl层，因此所测试的覆盖率为bl层的单元测试覆盖率。度量结果如下图示例

## 1.5小结

1. 在对整个项目有一个成熟的详细设计之后再进行mock object和单元测试JUnit的开发，不然会对后期的测试造成很大的不便。
2. 要尽量构造可测试的代码，为测试取得创造有利条件，由于在构造阶段未考虑测试的因素，因此导致有些代码片段测试起来比较困难。
3. 测试用例的设计中，关于边界值的测试往往能够发现隐藏的问题。在设计测试用例时，应多考虑这方面测试方法的使用。

# 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和部分层开发了Driver。在构造阶段采用自底向上的构造方式，每开发一个层次，就使用上一个层次的driver，进行maven build。逐步将上层集成至已经开发完成的代码当中。

## 2.2测试用例的有效性分析

总体上来说体系结构阶段开发的测试用例并不十分成功，在构造阶段，我们对体系结构的设计进行了部分调整，主要调整的内容包括层与层之间的调用接口，进行了更精细的考量，这就导致开发的stub与driver很多是失效的。所以在真正的体系结构定下来之后，项目小组重新开发了stub和driver来保证持续集成的正常进行。但其实正真的构造阶段也是集中在一个很短的时间段，因此持续集成的效果并不是特别明显，测试用例的有效性并不高。此外，由于一开始对于桩和驱动的理解并不是很透彻，不明白该如何在实际操作中利用桩和驱动进行集成测试，因此在构造初期并没有真正进行持续集成。

## 2.3采取的弥补手段

在对体系结构设计进行更精细的考量与调整后，再重新开发stub和driver。

## 2.4度量数据

**集成测试覆盖度：100%**

## 2.5小结

1. 在体系结构阶段一定要对设计做充分的考虑，形成相对稳定的接口，这样才能使在体系结构阶段写的桩和驱动在后期能真正被用来集成。
2. Stub开发不能过于简单，如果方法接口调用全都返回一样的值，会对单元测试造成较大的困难。