本周小结（第四周）

进入 11 月份，往往也意味着进入学期的期中考试阶段，同学们恐怕已经完成或正在经受多门课程扎堆考试的考验吧。我们的课程也接近过半，在此要感谢并祝贺所有已完成《软件测试与质量》课程第四周学习的同学们。开始一件事情很难，而坚持一件事情更难。欢迎大家坚持下去。

本周我们讨论了课程的第二部分 技术篇，并主要围绕第 3 章 白盒测试技术中的路径测试展开讨论。

本周我们主要回答了如下的问题。

1 独立路径测试的基本思想

路径测试就是对程序路径的执行进行测试，保证路径执行是按照设计的预期进行的。独立路径测试要解决如下三个核心问题：

- 找到一张用于记录程序路径的地图；
- 确定测试所需最少线性无关路径数；
- 找到所有可能迅速找到缺陷的最佳独立路径。

独立路径测试的原理图如图 1.1 所示：

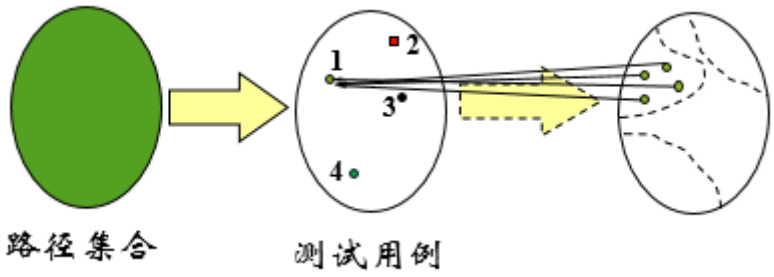

图 1.1 独立路径测试的原理图

2 什么是程序图？

程序图是反映程序结构复杂程序的一种图示，可以看做是一种简化、压缩的流程图或控制流图。

程序图的特点如下：

（1）对节点形状进行简化，采用统一的形状表示；

（2）忽略数据声明。

（3）忽略注释语句。

（4）压缩串行语句。

（5）压缩循环结构。

3 什么是环复杂度？

环复杂度是一种定量描述程序结构复杂度的度量模型。可采用三种方式确定程序的环复杂度：

（1）直观观察法。观察程序图将二维平面分割为封闭区域和开放区域的个数。

（2）公式法。V(G) = e – n +1。

（3）判定节点法。V(G) = P + 1。

请注意这些方法使用的前提条件。

4 如何设计测试用例？

4.1 基本步骤

基于独立路径设计测试用例的步骤如下：

- 将源代码转为程序图，作为路径测试的地图；
- 计算程序图的环复杂度，确定独立路径测试集合的规模；
- 找到所有独立路径；
- 剔除不可行路径；
- 补充路径测试；
- 根据要覆盖的路径，设计测试用例。

4.2 独立路径的确定

独立路径的确定包括两个步骤：确定主路径，以主路径为基础确定其他路径。

（1）确定主路径

主路径就是程序图中风险最高的那条执行路径。从结构来看，就是包含判定节点数目最多的一条路径。

（2）确定其他路径

以主路径为基础，针对主路径中的每个判定节点，依次覆盖已有独立路径中尚未覆盖的判定分支，生成新的独立路径。如果主路径未覆盖程序图中所有的判定节点，则需要针对主路径未覆盖到的那些判定节点，确定其余路径，直至找到所有独立路径。

4.3 不可行路径问题

不可行路径就是在程序执行中不可能执行到的路径。不可行路径破坏了独立路径测试的完备性和无冗余性，给测试带来额外的困难。

受到需求或程序设计的影响，当程序中存在相互关联的判定或循环结构时，可能引入不可行路径。因此，在确定独立路径的过程中，应结合源代码抽取路径，而非仅从程序图抽取独立路径，以确保每一条用于测试的独立路径都是可行路径。最好能结合需求进行程序结构设计，尽量避免在程序中引入不可行路径。

5 如何解决场景爆炸问题？

黑盒测试中，当场景中包含的事件流越多，可构成的场景越多，可能无法穷尽，造成场景爆炸问题。场景爆炸问题的解决可考虑三种方案。

（1）基于独立路径构建典型场景

将场景原理图看做程序图，借鉴独立路径测试方法，确定典型场景。优势是可以保证场景相互独立，无漏洞，但不可避免不可行场景。

（2）基于事件流的个数构建典型场景

根据事件流的个数，本着面向缺陷隔离的独立测试原则，对每个事件流单独设计测试。优势是易于缺陷定位，但仍不可避免不可行场景，且不保证场景独立和完备。

（3）基于需求构建典型场景

根据原始需求设计场景。优势是保证每个场景可行，但场景之间多存在冗余，且场景设计严重依赖于测试人员对需求的熟悉程度。

建议的使用策略如下：

- 当事件流之间相互独立时，建议基于独立路径设计场景。
- 否则，建议先基于独立路径设计场景，并保证场景可行，然后基于需求补充场景。

6 下周预告

下周，我们将进入第四章 测试管理与应用的讨论，在第四章，我们将讨论测试管理的相关工作内容，展示相关的测试管理工具，以及相关的测试工具，如功能测试工具、性能测试工具、代码检查工具等。第四周，我们主要来看看在实际的测试工作中，如何对测试用例和软件缺陷进行管理，并依据测试管理的流程来介绍相关的测试管理工具。