软件测试 Software Testing
02. 软件测试引论

程适

cheng@snnu.edu.cn

计算机科学学院

2016 年 10 月 13 日

陕西师範大學
SHAANXI NORMAL UNIVERSITY

Outline

- 软件测试的必要性
- 为什么要进行软件测试
- 什么是软件测试
- 软件测试和软件开发的关系
- 测试驱动开发的思想
- 数学基础

软件测试引论

- 软件开发的最基本要求是按时、高质量地发布软件产品，而软件测试是软件质量保证的最重要手段之一。
- 对于软件，不论采用什么技术和什么方法来进行开发，软件产品中仍然会存在或多或少的错误和问题。
- 采用先进的开发方式和较完善的开发流程，可以减少错误的引入，但是不可能杜绝软件的错误，这些引入的错误需要通过测试来发现。

软件测试引论

- 在整个软件生命周期中每个阶段、每个时刻都存在软件测试活动，软件测试伴随着软件开发，以检验每一个阶段性的成果是否符合质量要求和达到预先定义的目标，尽可能早地发现错误并及时改正。

软件测试的必要性

1. 软件问题造成的危害可能会非常严重，有时仅仅因为软件系统在存在一个很小的错误，却带来灾难性的后果。
2. $\longrightarrow$ 软件测试的必要性

为什么要进行软件测试

1. 保证软件质量
2. 软件测试是软件质量保证的关键步骤
3. 越早发现软件中存在的问题，开发费用就越低；在编码后修改软件缺陷的成本是编码前的10 倍，在产品交付后修改软件缺陷的成本是交付前的10 倍；软件质量越高，软件发布后的维护费用越低
4. 软件测试费用占整个软件工程所有研发费用的50%以上

什么是软件测试

1. 软件测试学科的形成
2. 正反两方面的争辩
3. 软件测试的定义
4. 软件测试的其他观点

软件测试学科的形成

1. 1972年，Bill Hetzel 在美国的北卡罗来纳大学（University of North Carolina）组织了历史上第一次正式的关于软件测试的会议。
2. 1973年，Bill Hetzel 正式为软件测试下了一个定义：“软件测试就是为程序能够按预期设想运行而建立足够的信心”
3. 1983年，Bill Hetzel 将软件测试的定义修改为：“软件测试就是一系列活动，这些活动是为了评估一个程序或软件系统的特性或能力，并确定是否达到了预期结果”。

软件测试学科的形成

1. 测试是试图验证软件是“工作的”，也就是验证软件功能执行的正确性；
2. 测试的目的也就是验证软件是否符合事先定义的要求；
3. 测试的活动是以人的“设想”或“预期的结果”为依据。这里的“设想”或“预期的结果”是指需求定义、软件设计的结果。

正反两方面的争辩

1. Glenford J. Myers 认为测试不应该着眼于验证软件是工作的，相反，应用用逆向思维去发现尽可能多的错误。
2. 1979 年，软件测试定义：“软件测试是为了发现错误而执行一个程序或者系统的过程”。
3. Myers的定义是引导人们证明软件是“不工作的”，以反向思维方式，不断思考开发人员理解的误区、不良的习惯、程序代码的边界、无效数据的输入以及系统的弱点，试图破坏系统、摧毁系统，目标就是发现系统中各种各样的问题。

软件测试的定义

- Glenford J. Myers的软件测试定义，虽然受到业界的普遍认同，但也存在一些问题：
  1. 如果只强调测试的目的是寻找错误，就可能使测试人员容易忽视软件产品的某项基本需求或客户的实际需求，测试活动可能会存在一定的随意性和盲目性。
  2. 如果只强调测试的目的是寻找错误，使开发人员产生一个错误的印象，测试人员的工作就是挑毛病的。
  3. 定义强调测试是执行一个程序或者系统的过程，即测试活动是在程序代码完成之后进行，而不是贯穿整个软件开发过程的活动，从而使软件测试的定义具有局限性和片面性。

软件测试的定义

1. IEEE 1983 of IEEE Standard 729 软件测试定义：使用人工或自动手段来运行或测试某个系统的过程，其目的在于验证它是否满足规定的需求或是弄清预期结果与实际结果之间的差别。

软件测试的定义

- 软件测试的正确定义就是：软件测试是由“验证（verification）”和“有效性确认（validation）”活动构成的整体：
  - “验证”是检验软件是否已正确地实现了产品规格书所定义的系统功能和特性。
  - “有效性确认”是确认所开发的软件是否满足用户真正需求的活动。

软件测试的其他观点

1. 风险的观点
2. 经济的观点

软件测试和软件开发的关系

1. 软件测试贯穿着整个软件生命周期，从需求评审、设计评审开始，测试就介入到软件产品的开发活动或软件项目实施中

测试驱动开发的思想

1. 测试驱动开发（Test Driven Development, TDD）
2. 测试第一的开发（Test-first programming）
3. 测试第一的设计（Test-first design）

测试驱动开发的思想

1. 测试在前，编码在后的开发方法：在编程之前，先写测试脚本或设计测试用例。

数学基础

- 离散数学包括：集合论、函数、关系、命题逻辑;
- 概率论。

集合定义

- 集合有三种方式定义：
  1. 简单列出集合的元素
  2. 给出辨别规则
  3. 通过其他集合构建

集合论–空集

- 空集采用符号ϕ表示，在集合中占有特殊位置;
- 空集不包含元素;
- 空集是唯一的，即不会有两个空集。

集合–集合关系

- A是B的子集$A \subseteq B$;
- A是B的真子集$A \subset B$;
- A是B的相等集合;

集合–子集划分

- 定义：子集划分
- 由于划分是一组子集，因此可以把单个子集看做是划分的元素。
- 划分可以保证完备性和无冗余性

函数

- 函数的定义
- 函数的定义域与值域
- 函数的类型
- 函数的合成

关系–集合之间的关系

- 集合之间的关系的定义
- 关系的势的定义
- 关系的参与的定义

命题逻辑

- 命题的定义
- 逻辑操作符
- 逻辑表达式
- 逻辑等价

概率论

- 事件的概率的定义
- 概率论在测试中的应用

图论

- 无向图
- 有向图
- 有/无向图: 如果给图的每条边规定一个方向，那么得到的图称为有向图。在有向图中，与一个节点相关联的边有出边和入边之分。相反，边没有方向的图称为无向图。

图–图的定义

- 图$G = V, E$ 是一个有序二元组，其中V 称为顶集(Vertices Set)，E称为边集(Edges set)，E与V不相交。
- $V = \{n_1, n_2, \ldots, n_m\}$ 和$E = \{e_1, e_2, \ldots, e_p\}$ 其中每条边$e_k = \{n_i, n_j\}$, n_i, $n_j \in V$。

图–基本概念

- 节点的度（Degree）：一个顶点的度是指与该顶点相关联的边的条数，顶点v的度记作$d(v)$。
- 图的关联距阵
- 图的相邻距阵

图–邻接矩阵

- 邻接矩阵（Adjacency Matrix）：是表示顶点之间相邻关系的矩阵。设$G = (V, E)$是一个图，其中$V = \{v_1, v_2, \ldots, v_n\}$。$G$的邻接矩阵是一个具有下列性质的n阶方阵：
  1. 对无向图而言，邻接矩阵一定是对称的，而且主对角线一定为零（仅讨论无向简单图），副对角线不一定为0，有向图则不一定如此。
  2. 在无向图中，任一顶点i的度为第i列（或第i行）所有非零元素的个数，在有向图中顶点i的出度为第i行所有非零元素的个数，而入度为第i列所有非零元素的个数。
  3. 用邻接矩阵法表示图共需要n^2个空间，由于无向图的邻接矩阵一定具有对称关系，所以扣除对角线为零外，仅需要存储上三角形或下三角形的数据即可，因此仅需要$nn - 1/2$个空间。

有向图–基本概念

- 入度（In-degree）和出度（Out-degree）：对于有向图来说，一个顶点的度可细分为入度和出度。一个顶点的入度是指与其关联的各边之中，以其为终点的边数；出度则是相对的概念，指以该顶点为起点的边数
- 节点的类型
- 有向图的相邻距阵

用于测试的图–程序图

- 给定一个采用命令式程序设计语言编写的程序，其程序图是一种有向图，其中：
- 节点是程序语句，边表示控制流(从节点i 到节点j 有一条边，当且仅当对应节点j 的语句可以立即在节点i 对应的语句之后执行)。

用于测试的图–有限状态机

- 有限状态机是一种有向图，其中状态是节点，转移是边；
- 源状态和吸收状态是初始节点和终止节点，路径被建模为通路。
- 大多数有限状态机表示方法都要为边（转移）增加信息，用以指示转移的原因和作为转移的结果要发生的行动。

用于测试的图–状态图

- 状态图是将为维恩图(Venn Diagram) 描述层次结构的能力以及有向图描述有向连接性的能力结合在一起，开发出一种可视化表示法。

小结

- 软件测试的必要性
- 为什么要进行软件测试
- 什么是软件测试
- 软件测试和软件开发的关系
- 测试驱动开发的思想
- 数学基础

小结

1. 为什么要开展软件测试活动？
2. 什么是软件测试？
3. 如何理解软件测试？
4. 软件测试和软件开发的关系是什么？

致谢

谢谢，欢迎提问!