二、 黑盒测试技术

测试需求

测试计划

测试设计

测试执行

测试管理

黑盒测试重点讨论的是测试设计问题：

如何根据软件需求产生合适的、适量的测试用例？

主要内容

1. 黑盒测试的概念
2. 测试需求的描述
3. 测试规格的描述
4. 从需求规格构造测试用例
5. 其它常用测试技术
6. 测试方法的选择

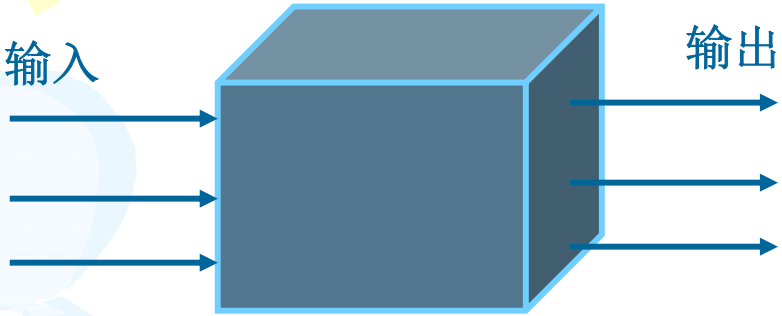
1 黑盒测试的概念

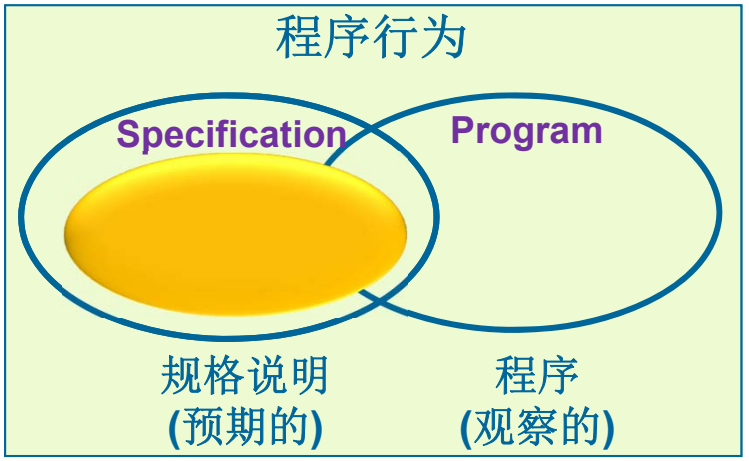
- 称谓
  - 黑盒测试又称功能性测试、数据驱动测试、基于规格说明的测试
- 定义
  - 在已知软件所应具有的功能的基础上，检查程序功能能否按需求规格说明书的规定正常使用，功能是否有遗漏，性能等特性要求是否满足。
- 特点

  参照规格说明检查软件，不要求考察代码，以用户视角进行测试

黑盒测试

- 将软件看做从输入到输出的黑盒，从规格说明出发





- 优点
  - 与软件如何编码实现无关，可适应实现变化
  - 测试工作可与实现并行，没有编码前即可开始准备测试
- 缺点
  - 测试用例集局限于已描述行为（安全漏洞等无法发现）

黑盒测试的常见内容

- 每项功能符合实际要求
- 程序安装、启动正常，有相应的提示框、错误提示等
- 系统的界面清晰、美观，功能清楚，符合使用者习惯
- 输入灵活，能接受正确的输入，对异常输入有容错处理
- 数据的输出结果准确，格式清晰
- 支持各种应用的环境，能配合多种软硬件
- 与外部应用系统的接口有效
- ……

黑盒测试的关键步骤

(1) 分析需求规格;

(2) 构造测试用例;

(3) 执行测试;

(4) 分析测试结果

准备阶段，可以和开发并行

步骤解读

- 分析需求规格
  - 从软件的需求规格出发，形成**测试的需求**
    - 具体要测什么功能（需求）？

      Q：校园一卡通有哪些功能？
    - 什么是一个功能正确的实现（规格）？

      Q：一卡通消费功能的正确实现应该做到怎样？
  - 软件的需求规格未必文档化，需要测试人员来分析
  - 以表格等方式清晰列出测试需求，指导规范化的测试
  - 如有可能，用严格的数学语言描述软件的需求规格

    严格的描述不仅更清晰，也便于自动生成测试用例

步骤解读

- 构造测试用例
  - 根据需求规格来构造原始测试用例集
  - 不考虑程序的内部结构
  - 测试用例不仅包括测试输入，还包括预期输出

    不同软件的输出千差万别，预期输出常无自动化办法获得，依赖测试者对软件的理解

步骤解读

- 执行测试
  - 选择合适、适量的测试用例进行测试
- 分析测试结果
  - 测试的充分性怎样？
  - 发现了哪些缺陷？是否严重？
  - 对软件质量的总体评价如何？

主要内容

1. 黑盒测试的概念
2. **测试需求的描述**
3. 测试规格的描述
4. 从需求规格生成测试用例
5. 其它常用测试技术
6. 测试方法的选择
7. 黑盒测试一些基本问题的探讨

2 测试需求的描述
Requirements

- 主要是关于功能成分的描述
  - 需求跟踪矩阵
  - 用例场景
- 每个测试需求对应可独立测试的基本功能块

2.1 需求跟踪矩阵（RTM）

- 用一个表格来表达测试过程关注的软件需求规格，关联相关的测试用例，作为开展测试工作的基础

锁和钥匙系统需求规格说明			
序号	需求标识	需求描述	优先级
1	BR-01	插入号码为123的钥匙并顺时针转动，应能上锁	高
2	BR-02	插入号码为123的钥匙并逆时针转动，应能开锁	高
3	BR-03	只有号码为123的钥匙可以用来上锁和开锁	高
4	BR-04	其它东西都不能用来上锁	中
5	BR-05	其它东西都不能用来开锁	中
6	BR-06	锁受重物撞击也不能被打开	中
7	BR-07	锁和钥匙重量必须为150g左右	低
8	BR-08	开锁和上锁转动方向可变，便于左手人士使用	低

2.1 需求跟踪矩阵

- 目的：
  - 对软件开发过程所有需求进行跟踪
  - 建立需求和测试用例的映射

需求跟踪矩阵样本

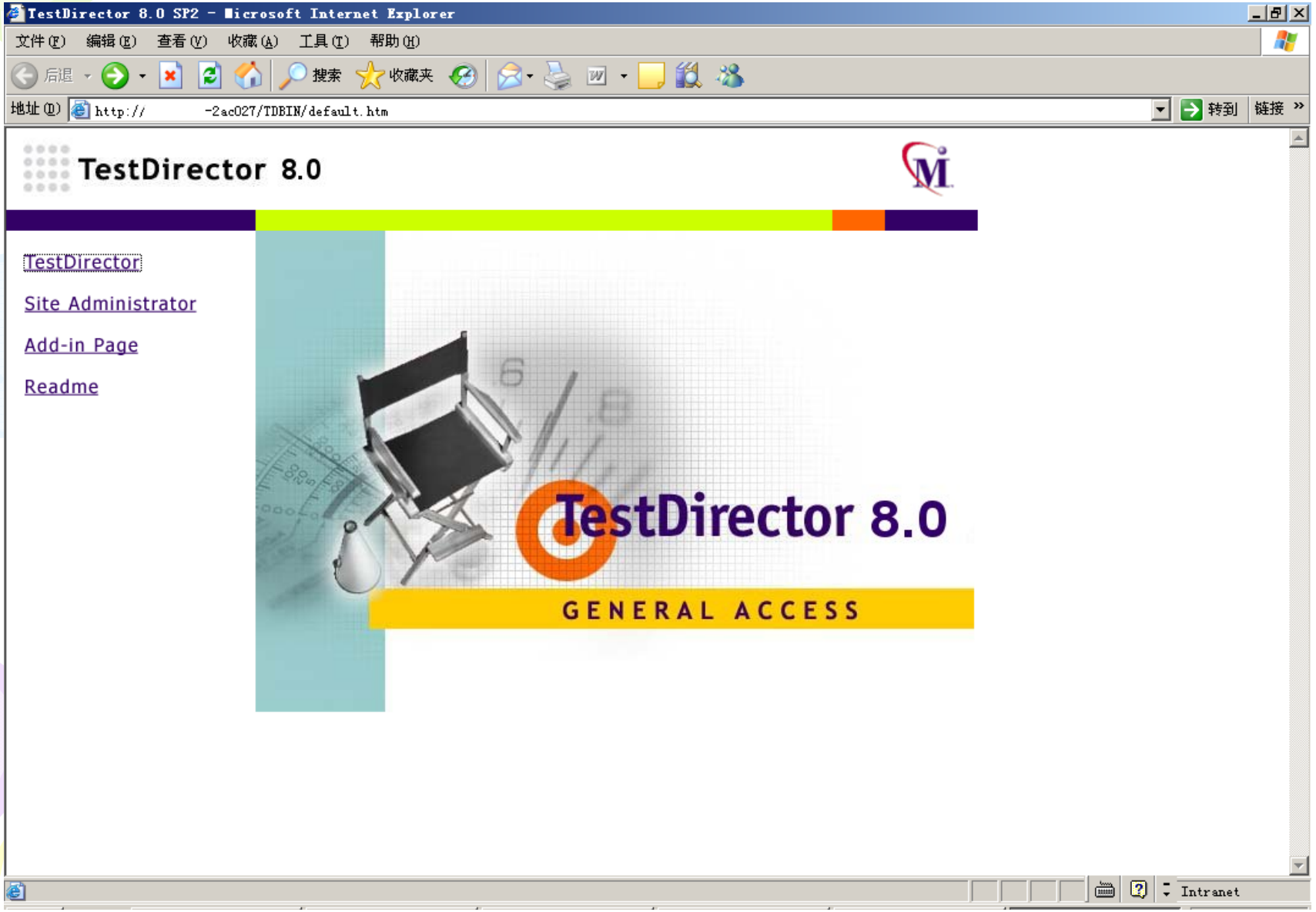
需求标识	需求描述	优先级	测试条件	用例标识	测试阶段
BR-01	插入号码为123的钥匙并顺时针转动，应能上锁	高	使用号码为123的钥匙	**LOCK_001**	单元 组件
BR-02	插入号码为123的钥匙并逆时针转动，应能开锁	高	使用号码为123的钥匙	**LOCK_002**	单元 组件
BR-04	其它东西都不能用来上锁	中	使用IC卡 使用发卡	**LOCK_005** **LOCK_006**	集成
BR-06	锁受重物撞击也不能被打开	中	使用石头砸锁	**LOCK_011**	系统
BR-08	开锁和上锁转动方向可变，便于左手人士使用	低			未实现

2.1 需求跟踪矩阵

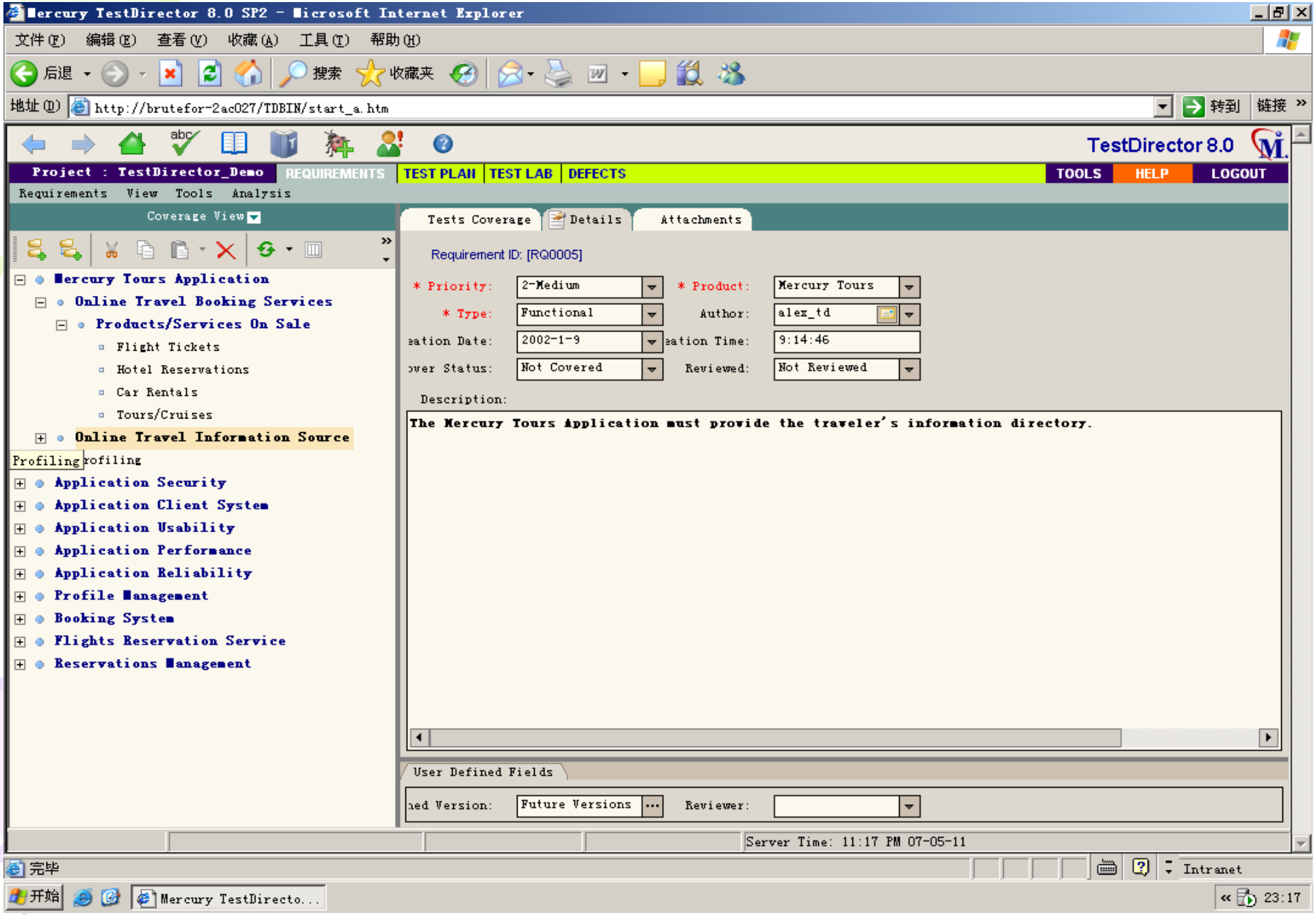
- 作用：
  1. 跟踪每个需求的测试状态，不会遗漏
  2. 可优先执行优先级高的测试用例，尽早发现重要缺陷
  3. 可导出特定需求对应的测试用例清单
  4. 作为评估测试工作量和测试进度的依据
     - 按优先级分类的需求数
     - 按需求分类的测试用例数
     - 已设计的测试用例数

矩阵形态多种多样，不拘泥于特定形式

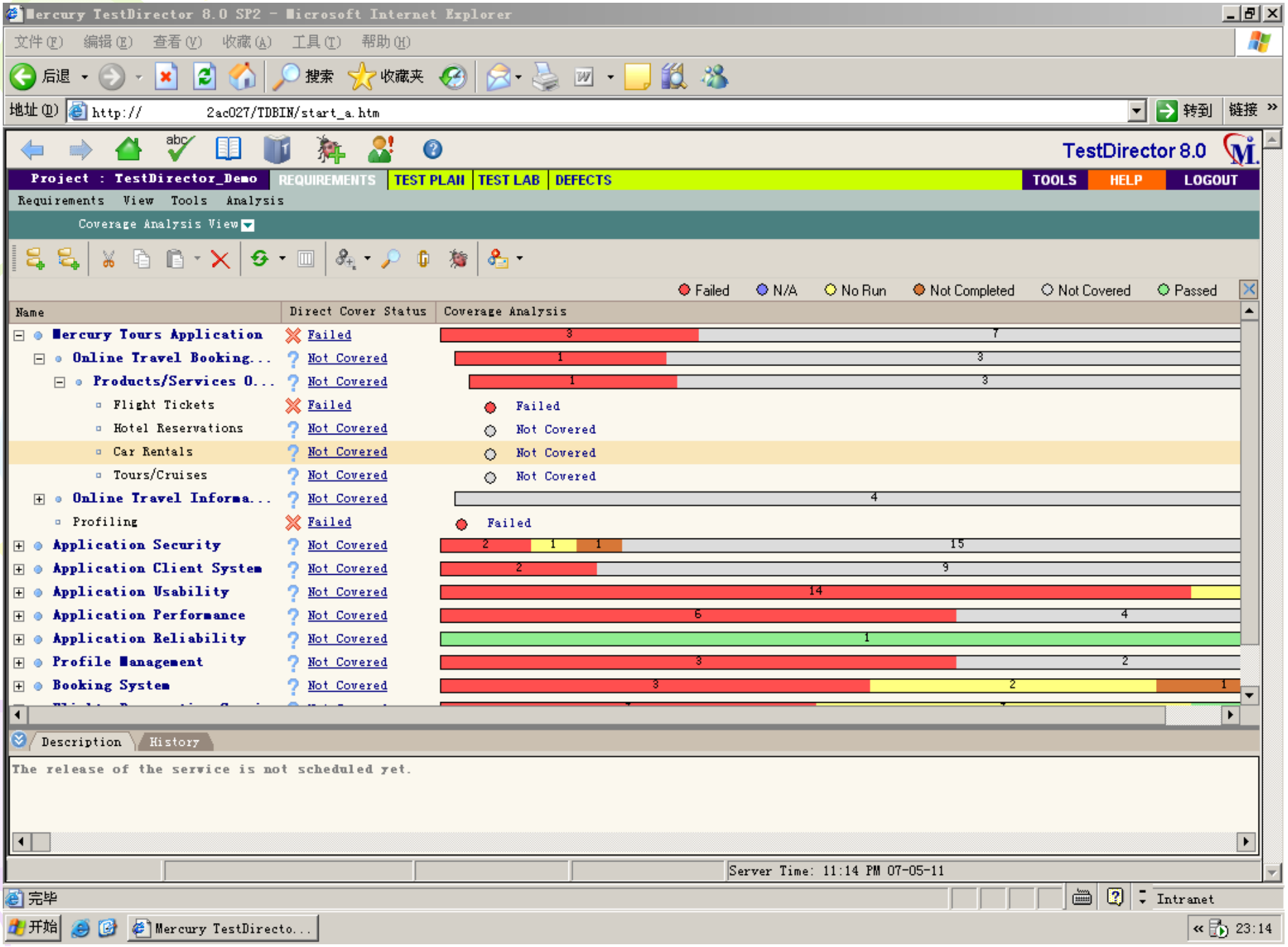
TestDirector的需求描述

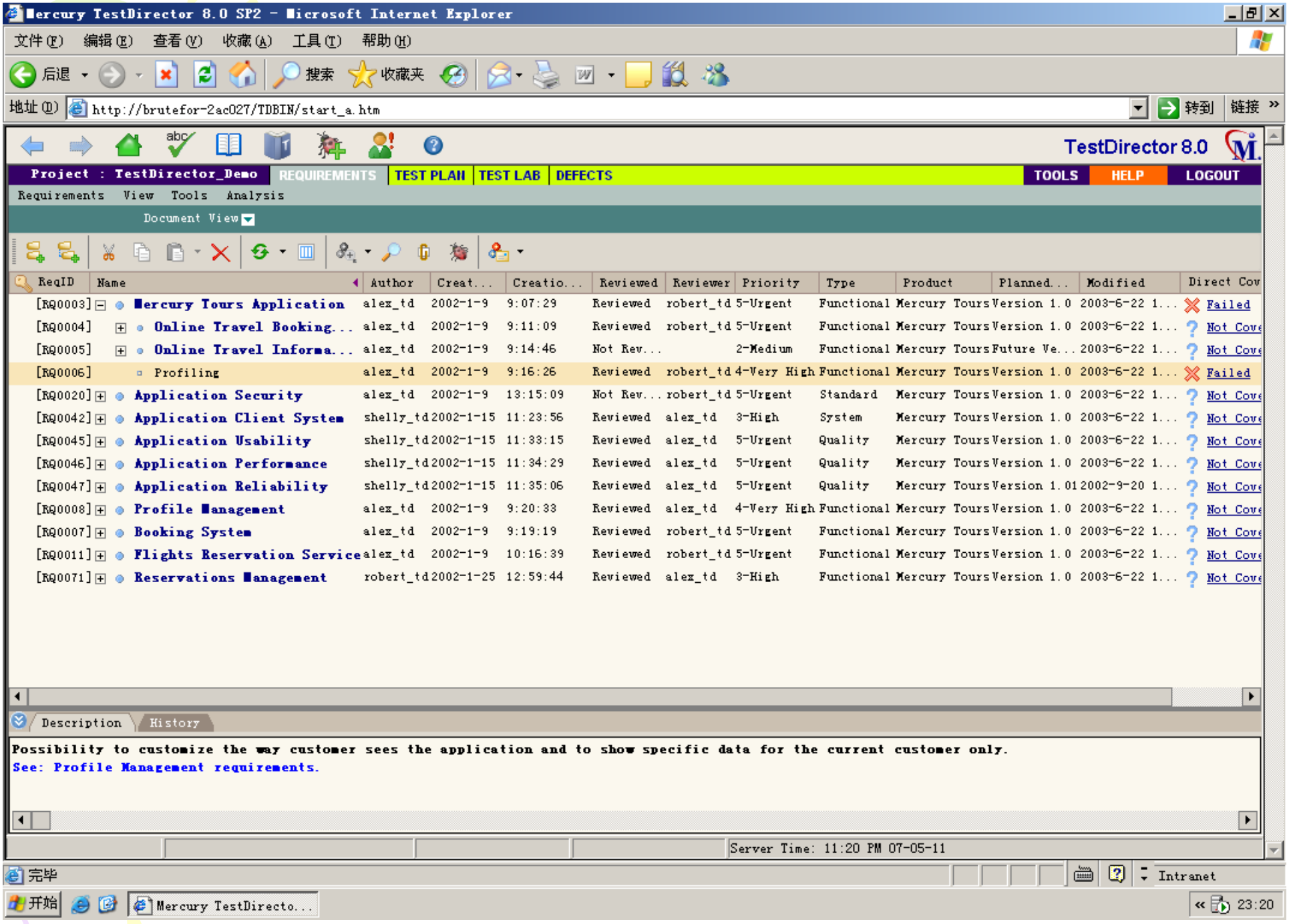


管理需求及其详细信息

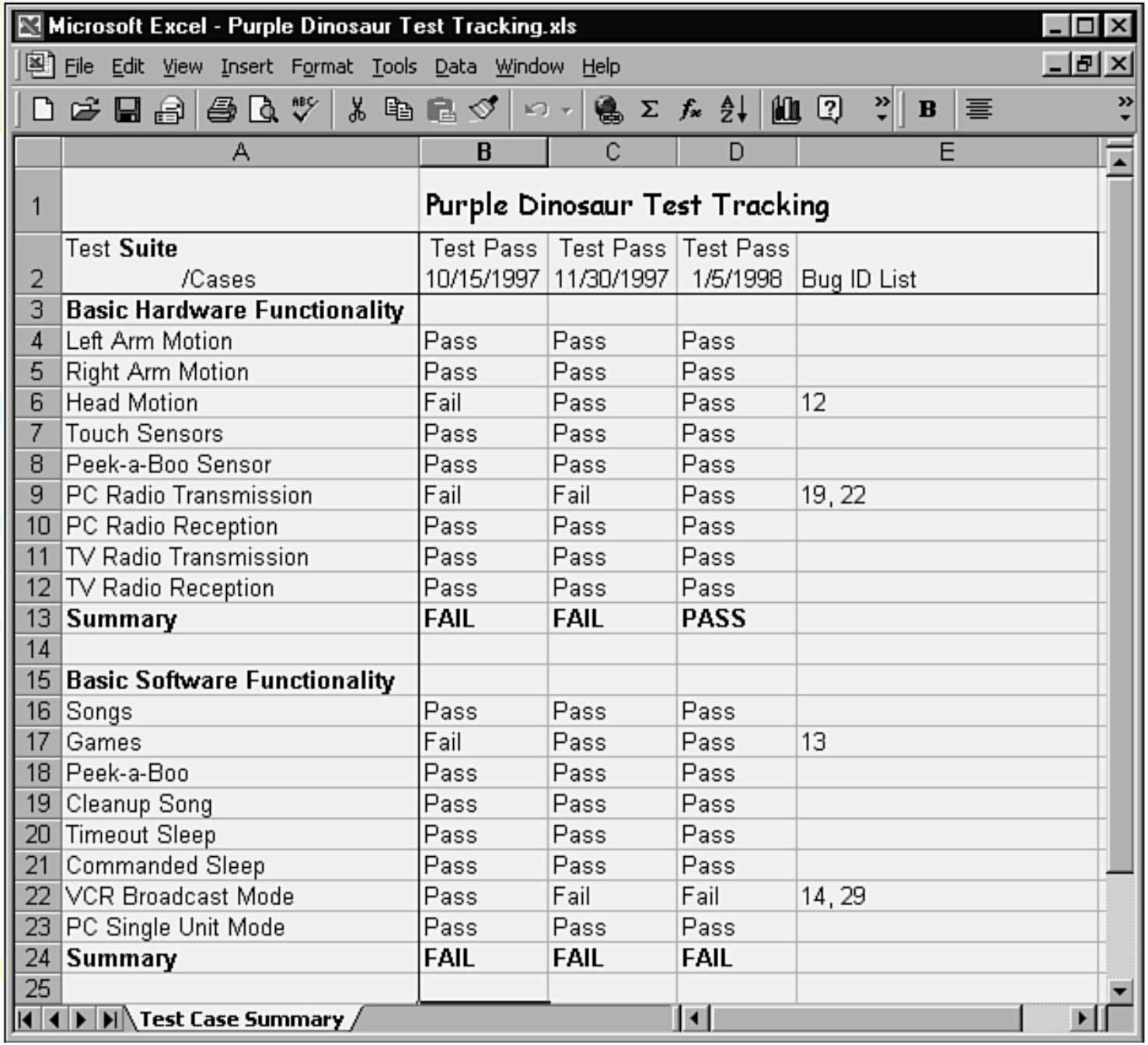

Mercury TestDirector 8.0 SP2 - Microsoft Internet Explorer
文件(F) 编辑(E) 查看(V) 收藏(A) 工具(T) 帮助(H)
后退 搜索 收藏夹
地址(D) http://brutefor-2ac027/TDBIN/start_a.htm
转到 链接
TestDirector 8.0
Project : TestDirector_Demo
REQUIREMENTS
TEST PLAN
TEST LAB
DEFECTS
TOOLS
HELP
LOGOUT
Requirements View Tools Analysis
Coverage View
Mercury Tours Application
Online Travel Booking Services
Products/Services On Sale
Flight Tickets
Hotel Reservations
Car Rentals
Tours/Cruises
Online Travel Information Source
Application Security
Application Client System
Application Usability
Application Performance
Application Reliability
Profile Management
Booking System
Flights Reservation Service
Reservations Management
Tests Coverage
Details
Attachments
Requirement ID: [RQ0005]
* Priority: 2-Medium
* Product: Mercury Tours
* Type: Functional
Author: alex_td
2002-1-9
9:14:46
Not Covered
Reviewed: Not Reviewed
Description:
The Mercury Tours Application must provide the traveler's information directory.
User Defined Fields
Future Versions
Reviewer:
Server Time: 11:17 PM 07-05-11
完毕
Intranet
开始
Mercury TestDirecto...
23:17

跟踪需求的测试进展


Mercury TestDirector 8.0 SP2 - Microsoft Internet Explorer
TestDirector 8.0
Project : TestDirector_Demo
REQUIREMENTS
TEST PLAN
TEST LAB
DEFECTS
TOOLS
HELP
LOGOUT
Requirements View Tools Analysis
Coverage Analysis View
Failed
N/A
No Run
Not Completed
Not Covered
Passed
Name
Direct Cover Status
Coverage Analysis
Mercury Tours Application
Online Travel Booking...
Products/Services O...
Flight Tickets
Hotel Reservations
Car Rentals
Tours/Cruises
Online Travel Informa...
Profiling
Application Security
Application Client System
Application Usability
Application Performance
Application Reliability
Profile Management
Booking System
Description
History
The release of the service is not scheduled yet.
Server Time: 11:14 PM 07-05-11
完毕
Intranet
开始
Mercury TestDirecto...
23:14


Mercury TestDirector 8.0 SP2 - Microsoft Internet Explorer
文件(F) 编辑(E) 查看(V) 收藏(A) 工具(T) 帮助(H)
后退 搜索 收藏夹
地址(D) http://brutefor-2ac027/TDBIN/start_a.htm
转到 链接
TestDirector 8.0
Project : TestDirector_Demo
REQUIREMENTS
TEST PLAN
TEST LAB
DEFECTS
TOOLS
HELP
LOGOUT
Requirements View Tools Analysis
Document View
ReqID Name Author Creat... Creatio... Reviewed Reviewer Priority Type Product Planned... Modified Direct Cov
[RQ0003] Mercury Tours Application alex_td 2002-1-9 9:07:29 Reviewed robert_td 5-Urgent Functional Mercury Tours Version 1.0 2003-6-22 1... Failed
[RQ0004] Online Travel Booking... alex_td 2002-1-9 9:11:09 Reviewed robert_td 5-Urgent Functional Mercury Tours Version 1.0 2003-6-22 1... Not Cove
[RQ0005] Online Travel Informa... alex_td 2002-1-9 9:14:46 Not Rev... 2-Medium Functional Mercury Tours Future Ve... 2003-6-22 1... Not Cove
[RQ0006] Profiling alex_td 2002-1-9 9:16:26 Reviewed robert_td 4-Very High Functional Mercury Tours Version 1.0 2003-6-22 1... Failed
[RQ0020] Application Security alex_td 2002-1-9 13:15:09 Not Rev... robert_td 5-Urgent Standard Mercury Tours Version 1.0 2003-6-22 1... Not Cove
[RQ0042] Application Client System shelly_td 2002-1-15 11:23:56 Reviewed alex_td 3-High System Mercury Tours Version 1.0 2003-6-22 1... Not Cove
[RQ0045] Application Usability shelly_td 2002-1-15 11:33:15 Reviewed alex_td 5-Urgent Quality Mercury Tours Version 1.0 2003-6-22 1... Not Cove
[RQ0046] Application Performance shelly_td 2002-1-15 11:34:29 Reviewed alex_td 5-Urgent Quality Mercury Tours Version 1.0 2003-6-22 1... Not Cove
[RQ0047] Application Reliability shelly_td 2002-1-15 11:35:06 Reviewed alex_td 5-Urgent Quality Mercury Tours Version 1.01 2002-9-20 1... Not Cove
[RQ0008] Profile Management alex_td 2002-1-9 9:20:33 Reviewed alex_td 4-Very High Functional Mercury Tours Version 1.0 2003-6-22 1... Not Cove
[RQ0007] Booking System alex_td 2002-1-9 9:19:19 Reviewed robert_td 5-Urgent Functional Mercury Tours Version 1.0 2003-6-22 1... Not Cove
[RQ0011] Flights Reservation Service alex_td 2002-1-9 10:16:39 Reviewed robert_td 5-Urgent Functional Mercury Tours Version 1.0 2003-6-22 1... Not Cove
[RQ0071] Reservations Management robert_td 2002-1-25 12:59:44 Reviewed alex_td 3-High Functional Mercury Tours Version 1.0 2003-6-22 1... Not Cove
Description History
Possibility to customize the way customer sees the application and to show specific data for the current customer only.
See: Profile Management requirements.
Server Time: 11:20 PM 07-05-11
完毕
Intranet
开始
Mercury TestDirecto...
23:20

一个简化版本的需求跟踪矩阵



Microsoft Excel - Purple Dinosaur Test Tracking.xls

File Edit View Insert Format Tools Data Window Help

	A	B	C	D	E
1		Purple Dinosaur Test Tracking			
2	Test **Suite** /Cases	Test Pass 10/15/1997	Test Pass 11/30/1997	Test Pass 1/5/1998	Bug ID List
3	**Basic Hardware Functionality**				
4	Left Arm Motion	Pass	Pass	Pass	
5	Right Arm Motion	Pass	Pass	Pass	
6	Head Motion	Fail	Pass	Pass	12
7	Touch Sensors	Pass	Pass	Pass	
8	Peek-a-Boo Sensor	Pass	Pass	Pass	
9	PC Radio Transmission	Fail	Fail	Pass	19, 22
10	PC Radio Reception	Pass	Pass	Pass	
11	TV Radio Transmission	Pass	Pass	Pass	
12	TV Radio Reception	Pass	Pass	Pass	
13	**Summary**	**FAIL**	**FAIL**	**PASS**	
14					
15	**Basic Software Functionality**				
16	Songs	Pass	Pass	Pass	
17	Games	Fail	Pass	Pass	13
18	Peek-a-Boo	Pass	Pass	Pass	
19	Cleanup Song	Pass	Pass	Pass	
20	Timeout Sleep	Pass	Pass	Pass	
21	Commanded Sleep	Pass	Pass	Pass	
22	VCR Broadcast Mode	Pass	Fail	Fail	14, 29
23	PC Single Unit Mode	Pass	Pass	Pass	
24	**Summary**	**FAIL**	**FAIL**	**FAIL**	
25					

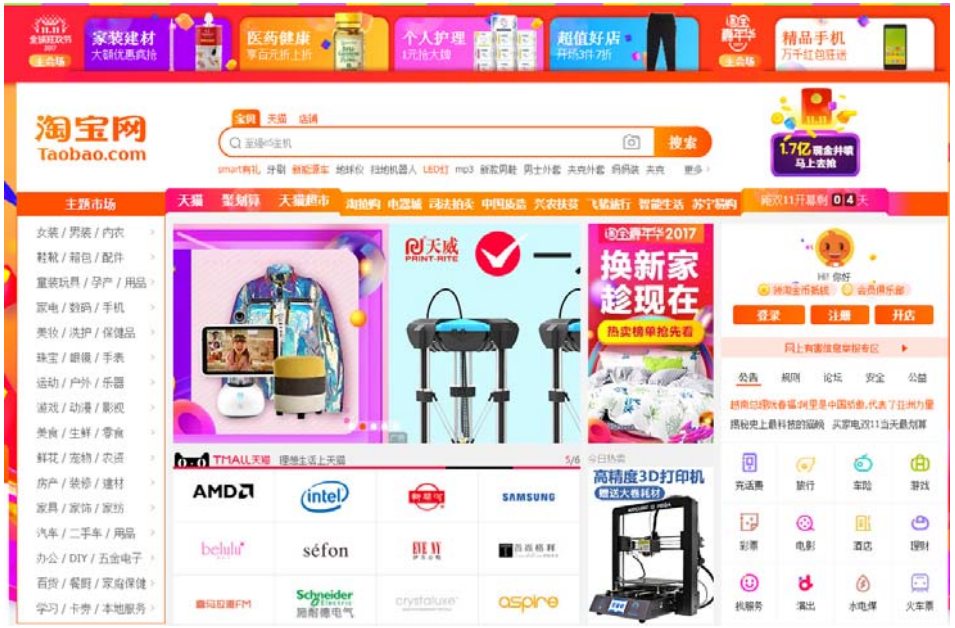
Test Case Summary

问题

- 需求跟踪矩阵难以直接应用于复杂软件

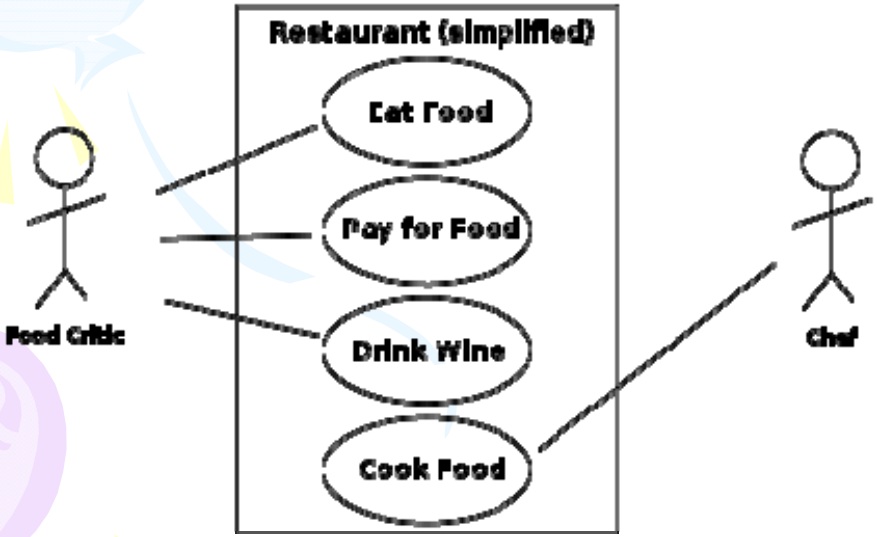
- 考虑以下问题：

  Q:淘宝有哪些功能点需要测试？



2.2 用例场景

- 用例：
  - 用例用于表示系统所提供的服务，它定义了系统是如何被参与者所使用的，它描述的是参与者为了使用系统所提供的某一完整功能而与系统之间发生的一段对话。



找出利益关联者**(stakeholders)**

找出利益关联者相关的主要活动类型

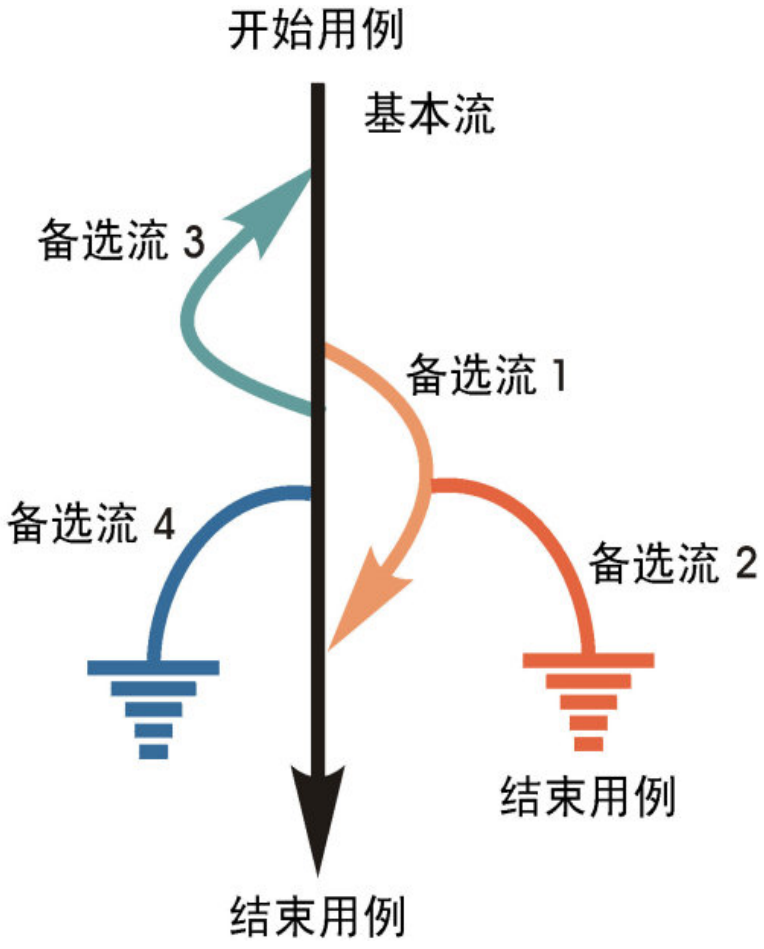
找出每类活动可能出现的各种情况

场景的获得

- 分析和识别用例
- 为每个用例列出基本流及各项备选流
- 根据基本流和各项备选流生成不同的场景

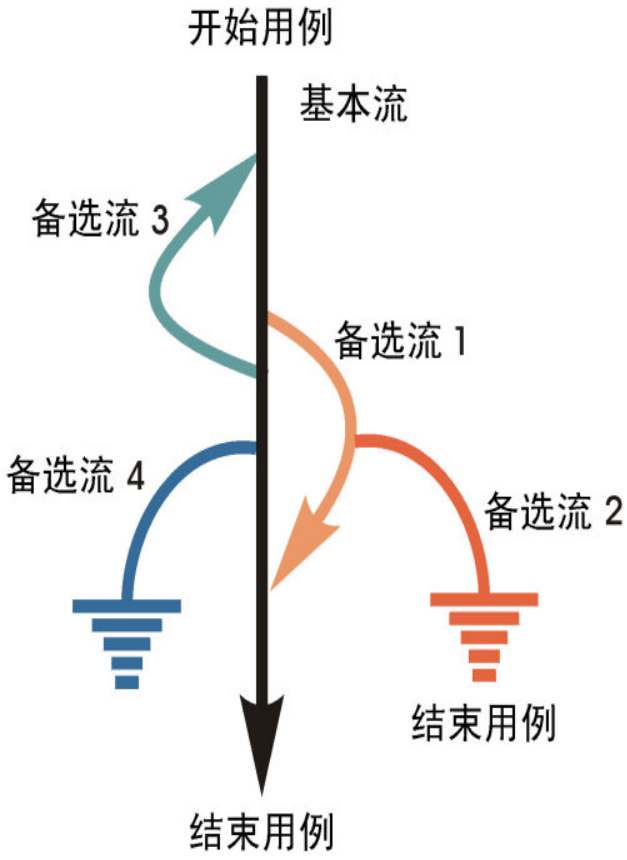
2.2 场景

- 场景用来描述流经用例的路径，从用例开始到结束，遍历这条路径上所有基本流和备选流。
- 基本流：正常流程
- 备选流：自基本流开始，在某特定条件下开始执行
  - 可能重新加入基本流（流1, 流3）
  - 可能起源于另一备选流（流2 ）
  - 终止用例不再重新加入某流（流2，流4）



场景

- 场景 1 基本流
- 场景 2 基本流 备选流 1
- 场景 3 基本流 备选流 1 备选流 2
- 场景 4 基本流 备选流 3
- 场景 5 基本流 备选流 3 备选流 1
- 场景 6 基本流 备选流 3
  备选流 1 备选流 2
- 场景 7 基本流 备选流 4
- 场景 8 基本流 备选流 3 备选流 4
- ……



例子：在线购书系统

- 订购过程：用户登录到网站后，选择书籍，选好书籍后进行订购，这时把所需图书放进购物车，选择完成后生成订单，审核订单并提交，付款，整个购物过程结束。

- 通过以上的描述，可确定基本流和备选流：

基本流	登录 选书 加入购物车 审核订单 提交 付款
备选流1	登录 帐号不存在
备选流2	登录 密码错误
备选流3	登录 无选购书籍
备选流4	登录 退出系统

用例场景

- 使用场景的优点
  - 用例驱动的开发，将需求直接表达成了场景
  - 场景的代表性易保证所选测试用例能够尽可能地反映问题
  - 场景易全面罗列，由此容易产生全面的测试用例
  - 场景可以较生动地描绘软件使用情景，有利于据此设计测试用例，同时使测试用例更容易理解和执行

作业二

主要内容

1. 黑盒测试的概念
2. 测试需求的描述
3. **测试规格的描述**
4. 从需求规格构造测试用例
5. 其它常用测试技术
6. 测试方法的选择

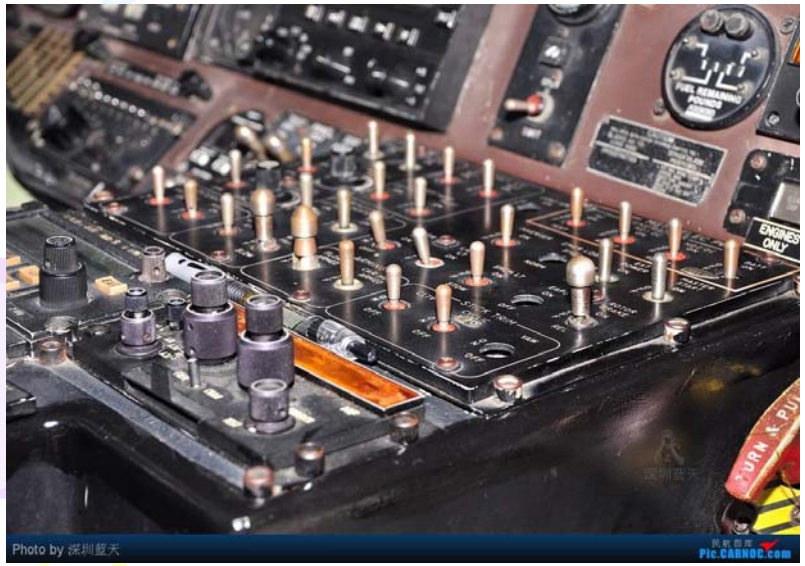
2 测试规格的描述 Specification

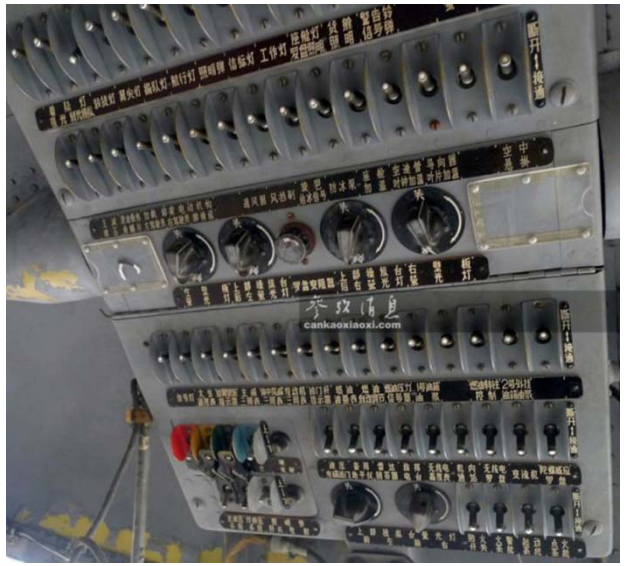
- 主要是关于功能正确性的描述
  - 因果图
  - 决策表
  - 契约（前置、后置条件）
  - 自动机（状态转换图）
  - 代数规约
  - 模态逻辑
  - Petri网
  - .....

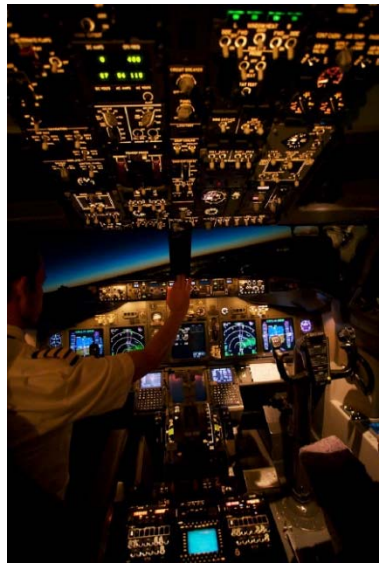
3.1 因果图

- 利用图解法分析输入的各种组合情况，分析输入和输出之间的关系。

- 适用于输入条件间具有各种组合关系，相互制约的情况，是一种高效、系统化的方法。

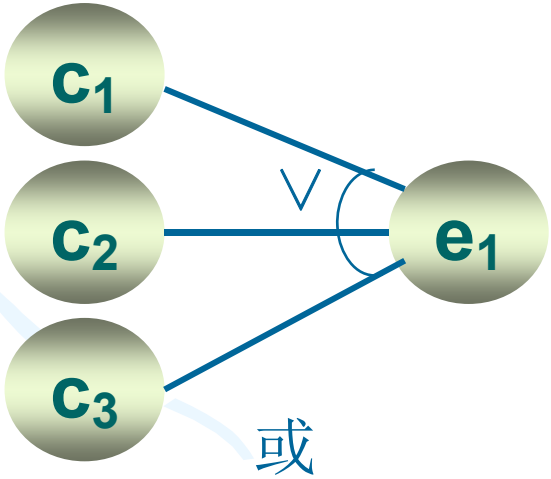


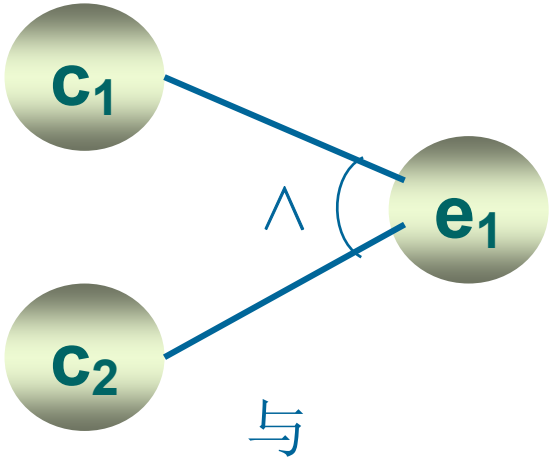




因果图基本符号

- 因果图中用来表示4种因果关系的基本符号：





c_i表示输入状态（因），e_i表示输出状态（果）
取值0或1：0—某状态不出现，1—某状态出现

因果图中的约束

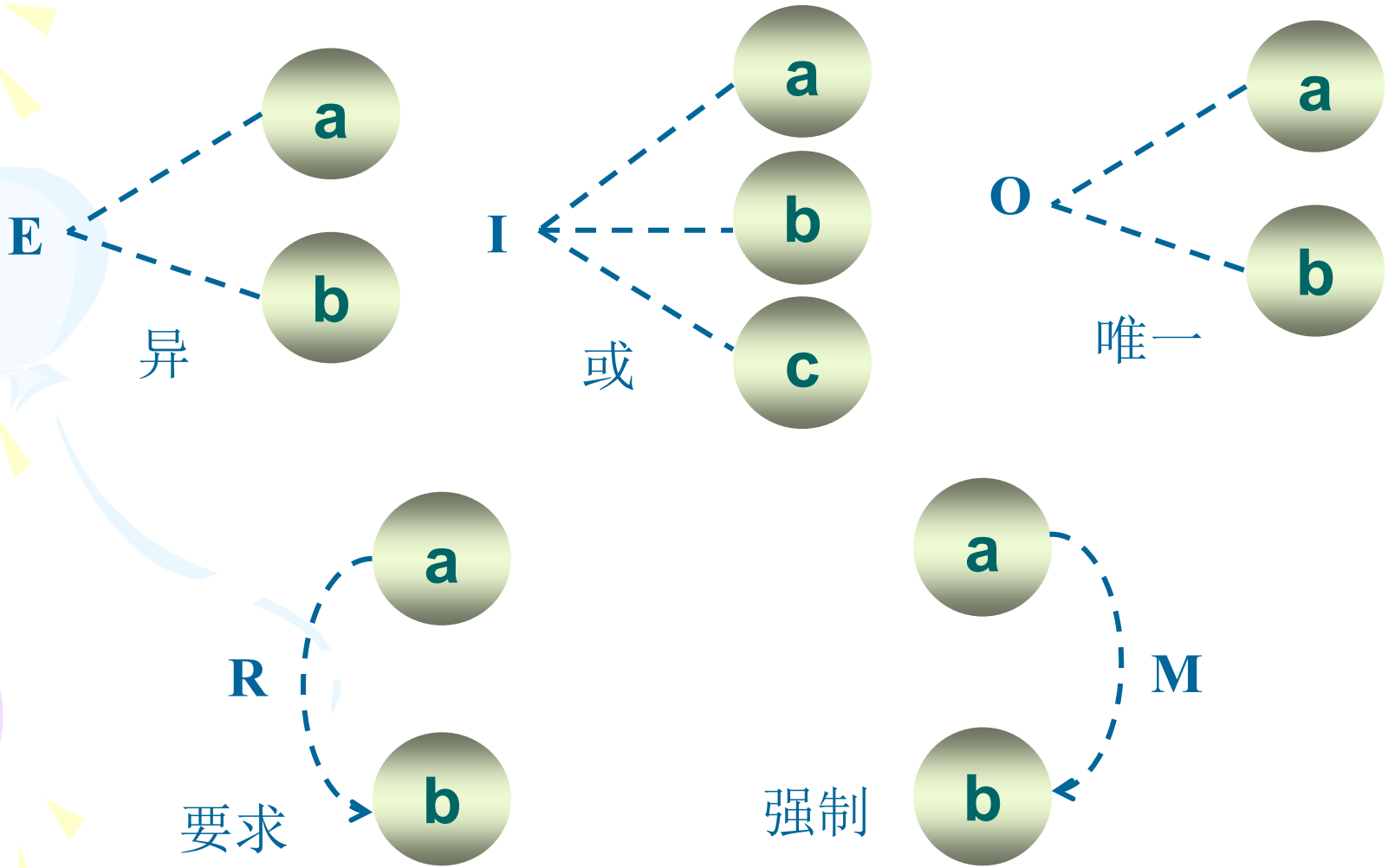
- 因果图中的约束

  实际问题中输入状态之间、输出状态之间可能存在某些依赖关系，称为“约束”。对于输入条件有E、I、O、R四种约束，对于输出条件只有M约束。

  - E约束(异)：a和b中最多有一个可能为1。
  - I 约束(或)：a、b、c中至少有一个必须为1。
  - O约束(唯一)：a和b必须有一个且仅有一个为1。
  - R约束(要求)：a是1时，b必须是1。
  - M约束(强制)：若结果a为1，则结果b强制为0。

因果图

- 因果图中用来表示约束关系的约束符号：



因果图的获得

- 具体步骤

（1）**因果分解**：将规格说明分解成可操作的块，分析其中哪些是因（输入条件或其等价类），哪些是果（输出条件），并给每个原因和结果赋予一个标识符。

（2）**关系识别**：分析规格说明中的语义，找出原因-结果、原因-原因之间对应的关系， 根据这些关系画出因果图。

（3）**约束限制**：由于语法或环境的限制，有些原因-原因、原因-结果之间的组合不可能出现。对此，在因果图上用一些记号表明约束或限制条件。

因果图举例

- **规格说明要求：**

  输入的**第一个字符必须是#或*，第二个字符必须是一个数字**，此情况下进行文件的修改；

  如果第一个字符不是#或*，则给出信息N；

  如果第二个字符不是数字，则给出信息M。

- **步骤：**

（1）分析程序的规格说明，列出原因和结果（必要时进行谓词抽象）。

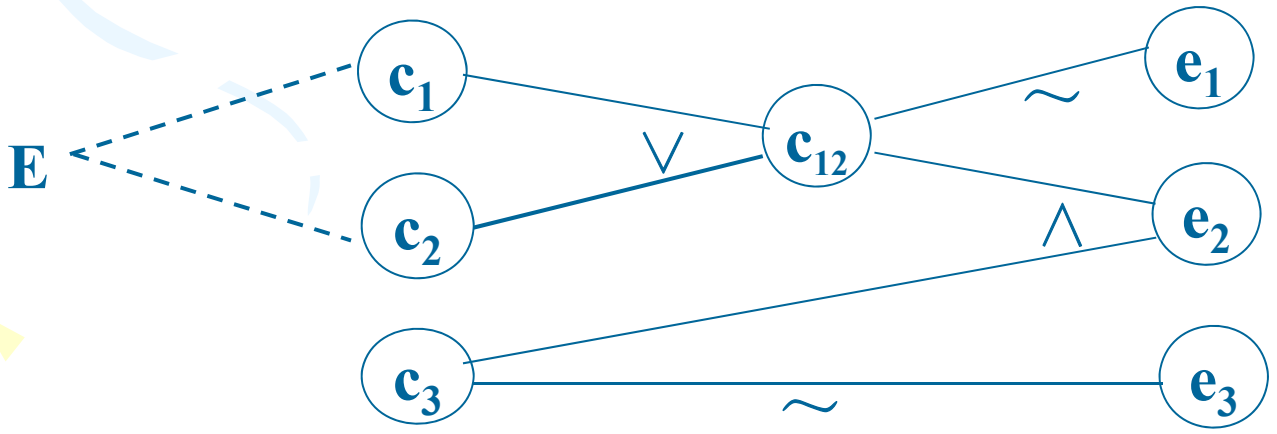
（2）找出原因与结果之间的因果关系、原因与原因之间的约束关系，画出因果图。

因果图举例（续）

（1）分析程序规格说明中的原因和结果：

原因	结果
c1：第一个字符是#	e1：给出信息N
c2：第一个字符是*	e2：修改文件
c3：第二个字符是一个数字	e3：给出信息M

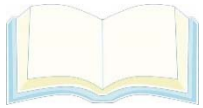
（2）画出因果图（中间结点c_{12}是导出结果的进一步原因）：



3.2 决策（判定）表

- 基于决策表的测试是最为严格、最具有逻辑性的测试方法之一。
- 决策表的概念
  - 决策表是分析和表达多逻辑条件下执行不同操作的情况的工具。

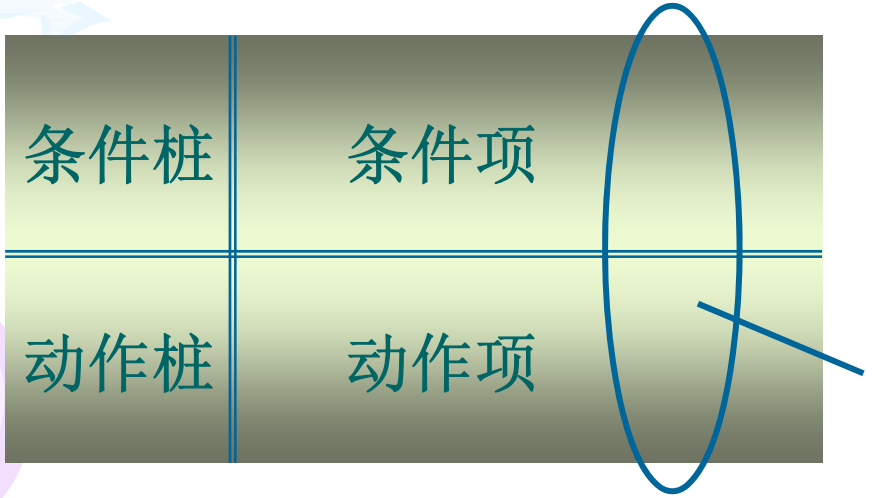
决策表——“阅读指南”



选项 \ 规则		1	2	3	4	5	6	7	8
问题	觉得疲倦？	Y	Y	Y	Y	N	N	N	N
	感兴趣吗？	Y	Y	N	N	Y	Y	N	N
	糊涂吗？	Y	N	Y	N	Y	N	Y	N
建议	重读					√			
	继续						√		
	跳下一章							√	√
	休息	√	√	√	√				

决策表的组成

- 决策表通常由以下4部分组成：
  - 条件桩—列出问题的所有条件
  - 条件项—针对条件桩给出的条件列出所有可能的取值
  - 动作桩—列出问题规定的可能采取的操作
  - 动作项—指出在条件项的各组取值情况下应采取的动作



将任一条件组合的特定取值及相应要执行的动作称为一条规则。

在决策表中贯穿条件项和动作项的一列就是一条规则。

决策表的优点

- 能够将复杂问题按照各种可能的情况全部列举出来，简明并避免遗漏。利用决策表能够设计出完整的测试用例集
- 决策表适合处理的问题：一些数据处理问题中，某些操作的实施依赖于多个逻辑条件的组合，即针对不同逻辑条件的组合值，分别执行不同的操作。
- 从代码看，决策表技术适用于具有以下特征的应用程序：
  - **If-then-else**逻辑很突出
  - 条件和规则的顺序不影响执行哪些操作。
  - 输入变量之间存在逻辑关系。
  - 输入与输出之间存在因果关系。
  - 程序有很高的圈复杂度。

构造决策表

- **5个步骤：**

（1） 确定规则的个数。

n个条件的决策表有2^n个规则（每个条件取真、假）

（2） 列出所有的条件桩和动作桩。

（3） 填入条件项。

（4） 填入动作项，得到初始决策表。

（5） 简化决策表，合并相似规则。

- 若表中有两条以上规则具有相同的动作，并且在条件项之间存在极为相似的关系，便可以合并。
- 合并后的条件项用符号“-”表示，说明执行的动作与该条件的取值无关，称为无关条件。

3.3 契约(contract)

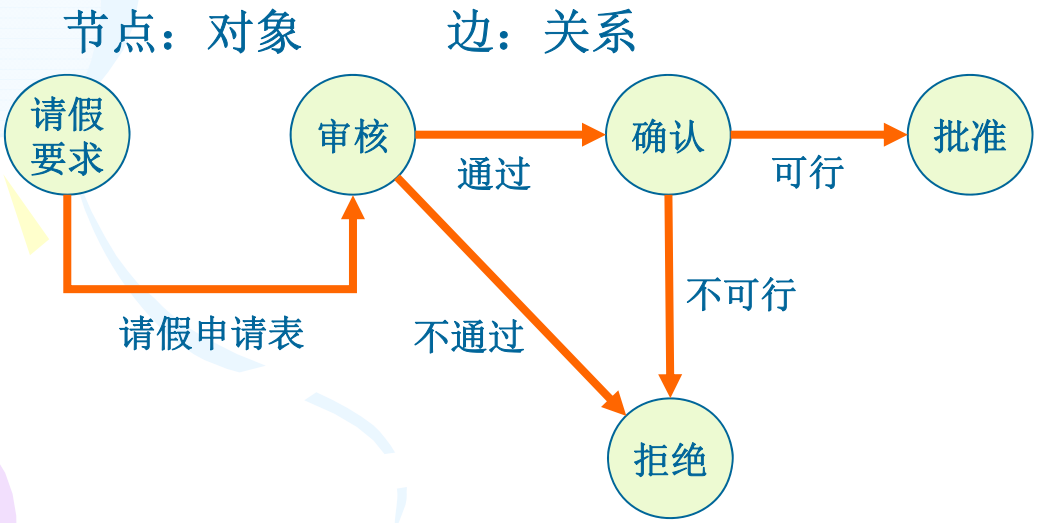
- 契约的形式
  - 前置条件：模块执行前应该满足的条件
  - 后置条件：模块执行后应满足的条件
  - 不变式：模块执行中应维持的一组性质

```
/*@ requires   amount >= 0;
  @ assignable balance;
  @ ensures    balance == \old(balance) - amount
  @            && \result == balance;
  @ signals (PurseException) balance == \old(balance);
  @*/
int debit(int amount) throws PurseException {
    if (amount <= balance) { balance -= amount; return balance; }
    else { throw new PurseException("overdrawn by " + amount); }
}
```

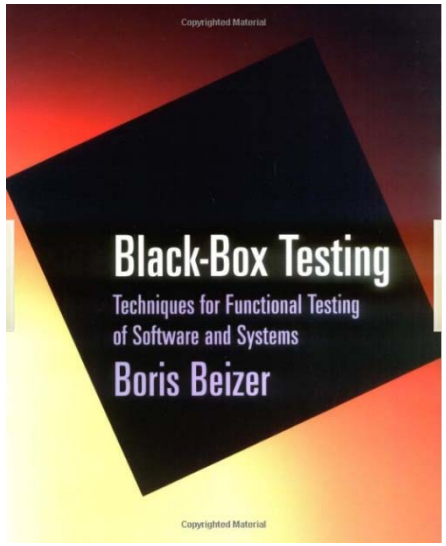

```
indexing
    description: "Simple bank accounts"
class
    ACCOUNT
feature -- Access
    balance: INTEGER                -- Current balance
    deposit_count: INTEGER is
      -- Number of deposits made since opening
      do
        ……
      end
feature -- Element change
    deposit (sum: INTEGER) is       -- Add sum to account.
      require
        non_negative: sum >= 0
      do
        ……
      ensure
        one_more_deposit: deposit_count = old deposit_count + 1
        updated: balance = old balance + sum
      end
feature { NONE } -- Implementation
    all_deposits: DEPOSIT_LIST
invariant
    consistent_balance: (all_deposits /= Void) implies
                        (balance = all_deposits . total)
    zero_if_no_deposits: (all_deposits = Void) implies
                         (balance = 0)
end
```

3.4 自动机（状态转换图）

- 软件所要实现的功能可用自动机（状态转换图）描述

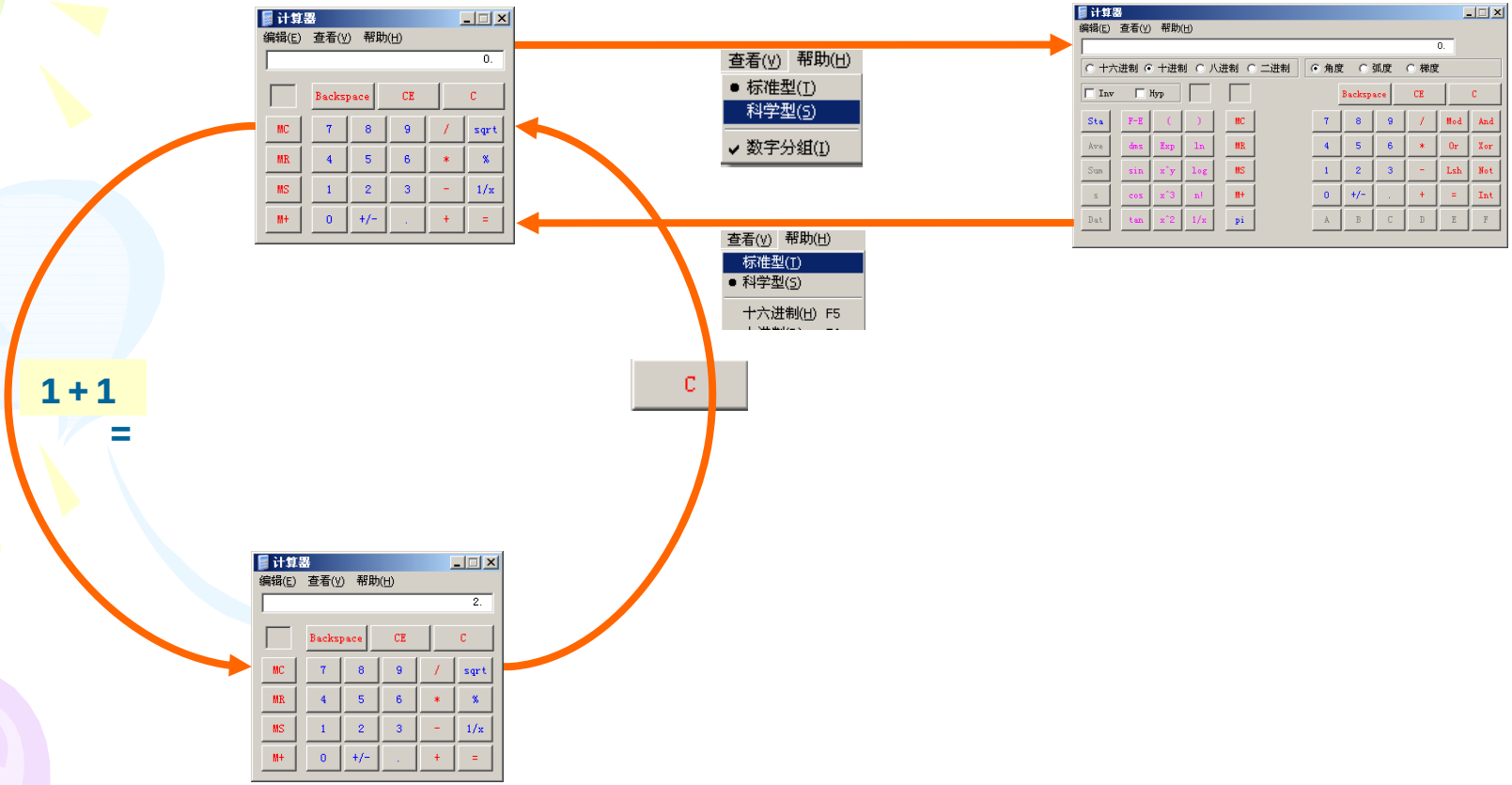


一个简单的工作流



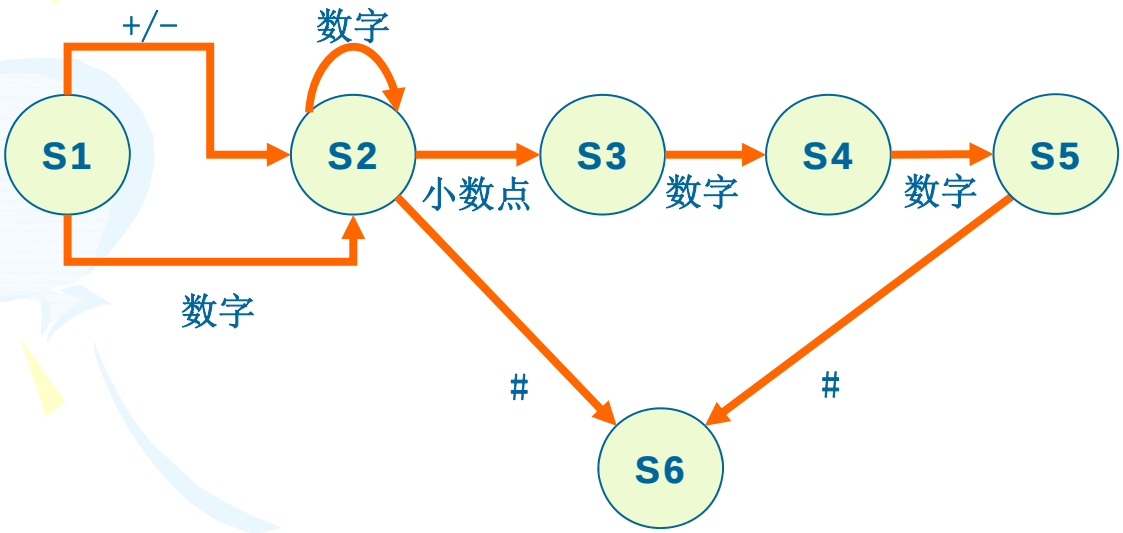
Boris Beizer, *Black-Box Testing*, Wiley,1995

计算器的功能—GUI



GUI

计算器的功能—输入数字的格式



确认数字有效性

1. 数字可以带符号
2. 符号后可接任意位数字
3. 可以有小数点
4. 小数点位数为2位
5. 任何数字都以#结束

输入格式

主要内容

1. 黑盒测试的概念
2. 测试需求的描述
3. 测试规格的描述
4. **从需求规格构造测试用例**
5. 其它常用测试技术
6. 测试方法的选择

4 从需求规格构造测试用例

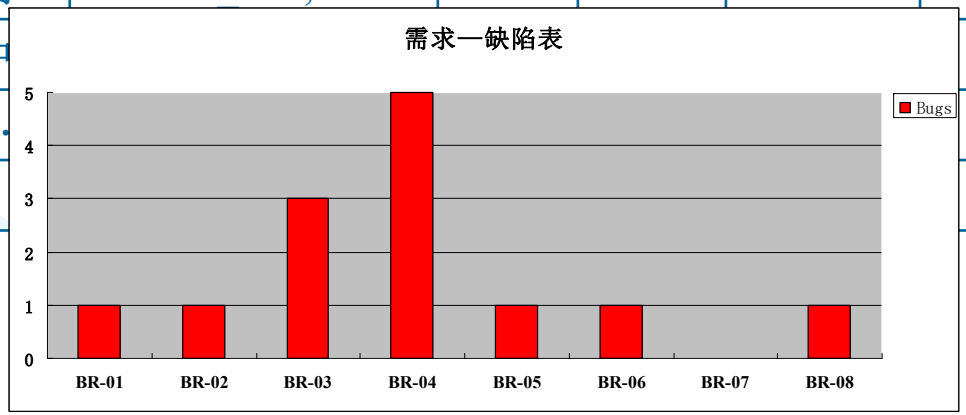
- 从需求跟踪矩阵构造测试用例
- 从用例场景构造测试用例
- 从决策表生成测试用例
- 从因果图生成测试用例
- 从自动机（状态图）生成测试用例

4.1 从需求跟踪矩阵构造测试用例

从需求跟踪矩阵出发，可直接为每个需求构造测试用例，执行并获得统计数据

1. 已通过/未通过的测试用例数
2. 各需求对应的缺陷总数
3. 已完成的需求数
4. 未通过的需求数

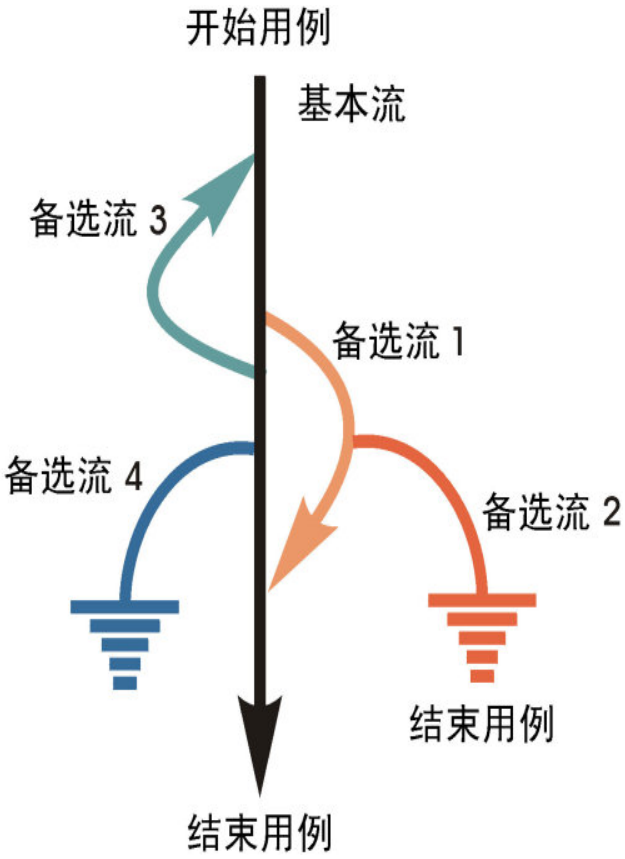
序号	需求标识	优先级	测试用例	用例总数	通过用例	未通过用例	通过率	缺陷数
1	BR-01	高	Lock_001	1	1	0	100	1
2	BR-02	高	Lock_002	1	1	0	100	1
3	BR-03	高	Lock_003,004	2	1	1	50	3
4	BR-04	中					67	5
…	…	…					…	…
合计	8						83	12



通过需求跟踪矩阵可对测试结果展开统计分析

4.2 从场景构造测试用例

- 列出每个场景
- 对每一个场景生成相应的测试步骤
- 测试步骤确定后，对每一个步骤确定测试数据值



场景法例子：在线购书系统

- 为每个场景生成一个测试用例

测试用例ID	场景/条件	帐号	密码	选购书籍	预期结果
1	场景1：购物成功	xu	123456	《软件测试艺术》《软件测试自动化》	成功购物
2	场景2：帐号不存在	zhang	n/a	n/a	提示帐号不存在
3	场景4：密码错误	xu	123$%^	n/a	提示密码错误 返回基本流步骤3
4	场景5：无选购书籍	xu	123456	空	提示选购书籍 返回基本流步骤5

购书的例子

4.3 从决策表生成测试用例

- 找出所有规则
- 对每条规则，根据相关条件要求给出可行输入
- 从动作桩获得预期输出

三角形问题的决策表

选项 \ 规则		规则 1-8	规则 9	规则 10	规则 11	规则 12	规则 13	规则 14	规则 15	规则 16
条件	c1: a,b,c构成三角形?	N	Y	Y	Y	Y	Y	Y	Y	Y
	c2: a=b?	-	Y	Y	Y	Y	N	N	N	N
	c3: a=c?	-	Y	Y	N	N	Y	Y	N	N
	c4: b=c?	-	Y	N	Y	N	Y	N	Y	N
动作	a1: 非三角形	√								
	a2: 一般三角形									√
	a3: 等腰三角形					√		√	√	
	a4: 等边三角形		√							
	a5: 不可能			√	√		√			

4.4 从因果图生成测试用例

- 具体步骤

（1）按顺序跟踪因果图中的状态条件，将图转换为决策表，表中的每列代表一个测试用例。

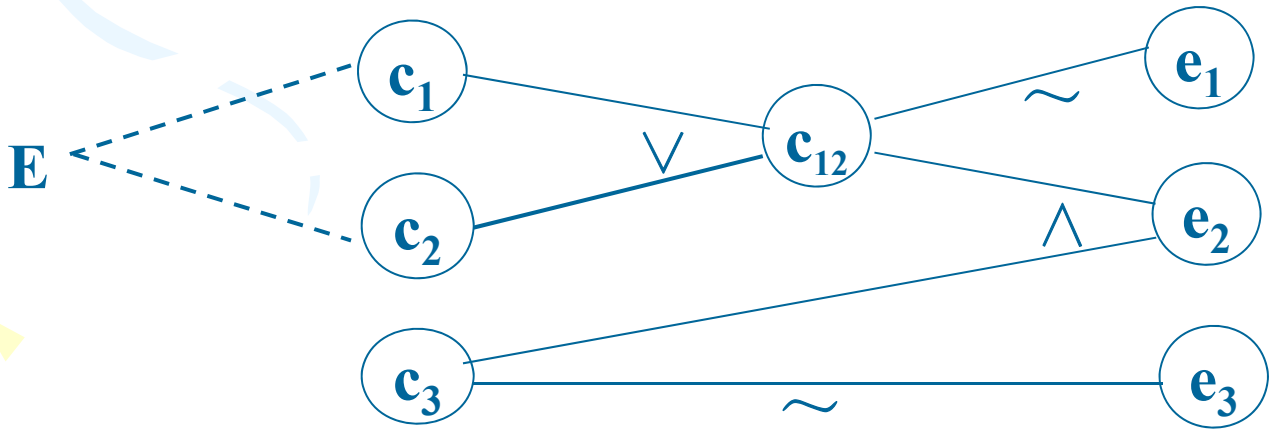
（2）根据决策表中的每一列设计测试用例。

举例

- **规格说明：** 输入的第一个字符必须是#或*，第二个字符必须是一个数字，此情况下进行文件的修改；如果第一个字符不是#或*，则给出信息N；如果第二个字符不是数字，则给出信息M。

原因	结果
c1：第一个字符是#	e1：给出信息N
c2：第一个字符是*	e2：修改文件
c3：第二个字符是一个数字	e3：给出信息M

- 因果图（中间结点c_{12}是导出结果的进一步原因）：



因果图法举例

（1）将因果图转换成如下所示的决策表：

选项 \ 规则		1	2	3	4	5	6	7	8
条件	C1：第一字符#	1	1	1	1	0	0	0	0
	C2：第一字符*	1	1	0	0	1	1	0	0
	C3：第二字符数	1	0	1	0	1	0	1	0
	C12： C1 ∨ C2			1	1	1	1	0	0
动作	e1			√		√		√	√
	e2				√		√		√
	e3								
	不可能	√	√						
测试用例				#3	#A	*6	*B	A1	GT

因果图法举例

（2）根据决策表中的每一列设计测试用例：

测试用例编号	输入数据	预期输出
1	#3	修改文件
2	#A	给出信息M
3	*6	修改文件
4	*B	给出信息M
5	A1	给出信息N
6	GT	给出信息N和信息M

因果图法的优点

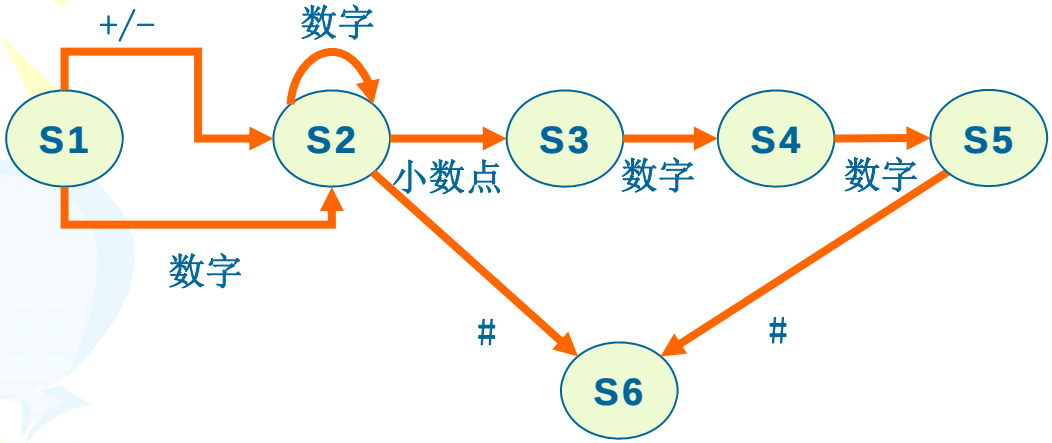
（1）考虑到了输入情况的各种组合以及各个输入情况之间的相互制约关系。

（2）能够帮助测试人员按照一定的步骤，高效率的开发测试用例。

（3）因果图法是将自然语言规格说明转化成形式语言规格说明的一种严格的方法，可以指出规格说明存在的不完整性和二义性。

4.5 从自动机（状态图）生成测试用例

- 遍历图，覆盖图中所有的边，就可以导出测试用例



确认数字有效性

1. 数字可以带符号
2. 符号后可接任意位数字
3. 可以有小数点
4. 小数点位数为2位
5. 任何数字都以#结束

当前状态	输入	期望状态
1	+	2
2	3	2
2	.	3
3	1	4
4	4	5
5	空格	6

主要内容

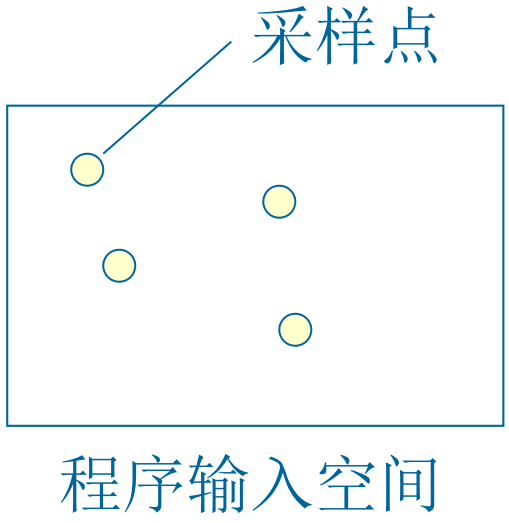
1. 黑盒测试的概念
2. 测试需求的描述
3. 测试规格的描述
4. 从需求规格构造测试用例
5. **其它常用测试技术**
6. 测试方法的选择
7. 黑盒测试一些基本问题的探讨

5.其它常用测试技术

- 随机测试法
- 等价类划分法
- 边界值分析法
- 正交实验设计
- 错误推测与探索式测试

5.1 随机测试法

- 测试的本质
  - 测试是一种采样分析
    - 程序输入空间太大
    - 资源有限
- 最简单的采样是随机采样
  - 简单随机
  - 过程随机
  - ……

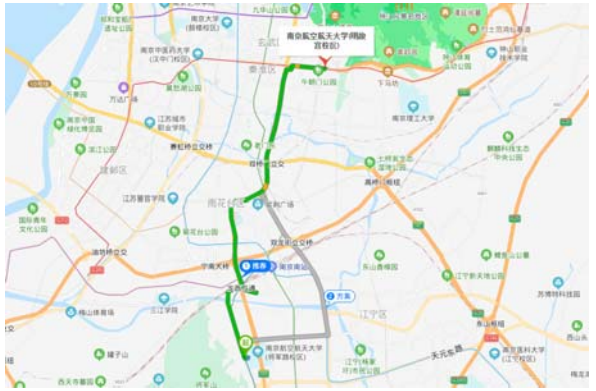


5.1 随机测试法

- 简单随机
  - Y = sin(X)

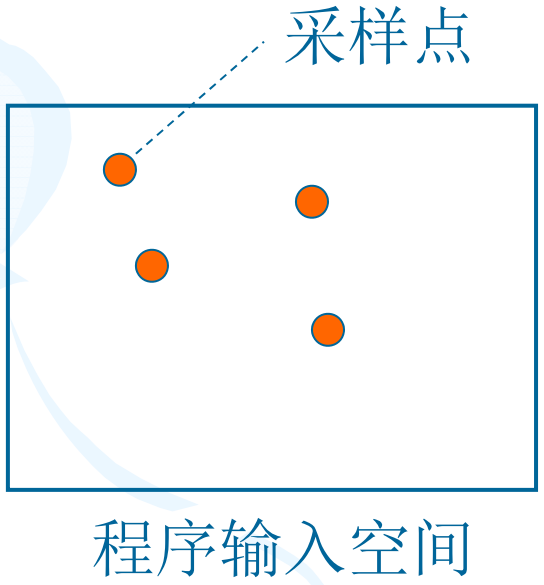
  - 在实数域上随机取值进行测试
    - 34, 57, 102.1

- 过程随机
  - Q: 将军路校区到本部随机走条路线该怎么办？



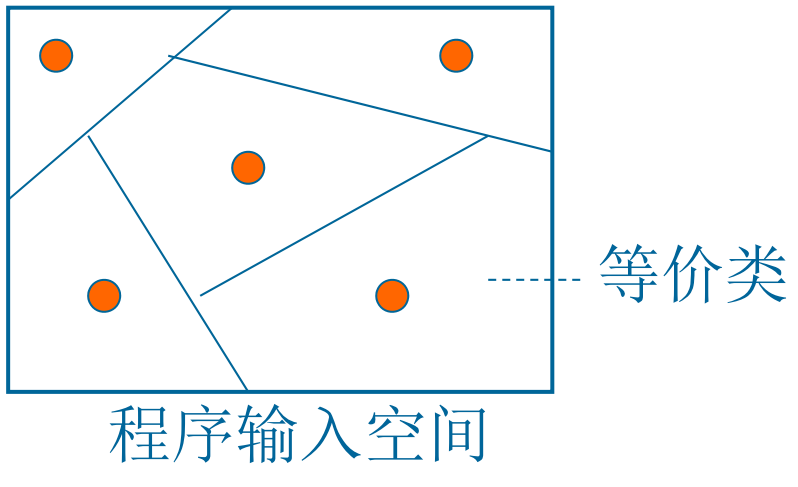
5.2 等价类划分法

- 如何采样？



随机测试

避免随机测试的盲目性
但需要关于输入域的更多知识



等价类划分法

等价类划分法

- 等价类划分法
  - 根据规约，将输入空间根据测试的等效性，划分为多个等价类，每个分类仅取一个或少量测试用例来进行测试
- 测试的等效性
  - 等价类中各输入对于揭露程序中的错误等效
    - 合理地假定：测试某等价类的代表值就等价于对这一类其他值的测试。
- 优点
  - 代表性：少量测试数据
  - 全面性：取得较好的测试结果

等价类的划分

- 划分等价类可分为两种情况：

（1）有效等价类

对软件规格说明而言，有意义的、合理的输入数据所组成的集合。能够检验程序是否实现了规格说明中预先规定的功能和性能。

（2）无效等价类

对软件规格说明而言，无意义的、不合理的输入数据所构成的集合。可以鉴别程序异常处理的情况，检查被测对象的功能和性能的实现是否有不符合规格说明要求的地方。

等价类的划分

- 划分的总体思路
  - 识别关键输入条件，根据输入数据本身的特征，以及可能造成的程序行为差异，采用各种方法逐步细分
  - 三点
    1. 根据输入数据本身的特征进行划分
    2. 根据可能造成的程序行为差异进行划分
    3. 对划分的结果进一步细分

	有效等价类	无效等价类
输入数据自身特点		
输出差别		

Q: 一个成绩管理系统，59.5分以上取整判定为及格，以下判定为不及格
输入本身什么特征，输出差异什么特征？

一些等价类划分方法
—根据输入数据本身的特征进行划分

- ① 按输入值个数划分：一个有效等价类，两个无效等价类
  - 例：输入条件为“值个数10”
    - 有效类：　个数10
    - 无效类：　个数<10, 个数>10
- ②按输入值区间划分：一个有效等价类，两个无效等价类
  - 程序输入条件为10<x<100，则
    - 有效等价类 10<x<100
    - 无效等价类 x≤10, x≥100

一些等价类划分方法
—根据输入数据本身的特征进行划分（续）

- ③ 按数值集合划分
  - 在输入条件规定了输入值的集合或规定了“必须如何”的条件下，可以确定一个有效等价类和一个无效等价类（有效值集合外）

  - 例：程序输入条件为取值为奇数的整数x，则
    - 有效等价类：x的值为奇数
    - 无效等价类：x的值不为奇数。

一些等价类划分方法
—根据输入数据本身的特征进行划分（续）

- **④ 按限制条件或规则划分**

  在规定了输入数据必须遵守的规则或限制条件的情况下，可确定一个有效等价类（符合规则）和若干个无效等价类（从不同角度违反规则）。

- 例：程序输入条件为以字符‘a’开头、长度为8的字符串，且字符串不含‘a’~‘z’外的其它字符，则
  - 有效等价类：满足了上述所有条件的字符串
  - 无效等价类：
    - 不以‘a’开头的字符串
    - 长度不为8的字符串
    - 包含了‘a’~‘z’之外其它字符的字符串

一些等价类划分方法

——根据可能造成的程序行为差异进行划分

- ⑤ 按输入处理的不同划分
  - 在输入条件规定了输入数据的n种可能情况，且程序对每种情况分别进行处理的情况下，可确定 n 个有效等价类（一正常情况一有效等价类）和一个无效等价类（所有不允许的输入值的集合）。
  - 例：某程序的输入 x 取值于一个固定的枚举类型{1,3,7,15}，且它对这4个数值分别进行了处理，则
    - 有效等价类为x=1、x=3、 x=7、x=15
    - 无效等价类为x≠1,3,7,15的值的集合

一些等价类划分方法
—细分等价类

- ⑥ 细分等价类
  - 在测试资源允许的情况下，可将等价类进一步划分为更小的等价类，并建立等价类表。

等价类法使用实例

- 问题
  - 输入三个正整数作为三边的边长，构成三角形。当此三角形为一般三角形、等腰三角形及等边三角形时，分别作计算……
- 分析：
  - 关键输入条件：
    - 整数
    - 三个数
    - 正数
  - 程序行为差异：
    - 一般三角形：应满足两边之长大于第三边
    - 等腰三角形
    - 等边三角形

等价类表

			有效等价类	无效等价类
输入条件	三个正整数	① 按输入值个数	三个数	只给一个边
				给了两个边
				给了三个边以上
		②按输入值区间划分	正数	一边≤ 0
				两边≤ 0
				三边≤ 0
		③ 按数值集合划分	整数	一边非整数
				两边非整数
				三边非整数

等价类表

			有效等价类	无效等价类
输入数据的处理	⑤ 按输入处理的不同划分	非三角形		$a+b\le c \vee a+c\le b \vee c+b \le a$
		一般三角形	$a+b>c \wedge a+c>b \wedge c+b>a \wedge a\neq b \wedge b\neq c \wedge a\neq c$	
		等腰三角形	$a=b$	
			$b=c$	
			$a=c$	
		等边三角形	$a=b=c$	

仍可细分：“或”可进一步划分，≥ 可分为 > 和 = ，.....

等价类划分法的测试用例设计

1. 为等价类表中的每个等价类分别规定一个唯一编号。
2. 设计一个新的测试用例，使它能够尽量多地覆盖尚未覆盖的有效等价类。重复该步，直到所有的有效等价类均被测试用例所覆盖。
   - 有效等价类（多条件**AND**）：一个测试用例可覆盖多
3. 设计一个新的测试用例，使它仅覆盖一个尚未覆盖的无效等价类。重复该步，直到所有的无效等价类均被测试用例所覆盖。
   - 无效等价类 （多条件**OR**）：尽可能一个测试用例覆盖一个

5.3 边界值分析法

- 边界值分析法就是对输入、输出或内部的边界值进行测试的一种测试方法。
  - 经验得知，大量的错误发生在输入、输出范围的边界上，而不是其内部
  - 针对各种边界情况设计测试用例，可发现更多错误
- 通常边界值分析法是作为对**等价类划分法的补充**，这种情况下，其测试用例来自等价类的边界。
  - 着重测试输入等价类、输出等价类的边界情况

边界的类型

1. 输入边界

(1) 若输入条件规定了值的范围，则应取刚达到这个范围以及刚超过这个范围的边界值作为测试输入数据。

(2) 若输入条件规定了值的个数，则用最大个数、最小个数和比最大个数多1个、比最小个数少1个的数作为测试数据。

(3) 若程序的规格说明给出的输入域或输出域是有序集合（如有序表、顺序文件等），则应选取集合中的第一个和 最后一个元素作为测试用例。

(4) 分析程序规格说明，找出其它可能的边界条件。

边界的类型

2. 输出/行为边界

(5) 根据程序规格说明的每个输出条件，使用原则 (1)-(4)

3. 内部边界

(6) 如果程序中使用了一个内部数据结构，则应当选择这个内部数据结构的边界上的值作为测试用例。

常见的边界值

- 输入边界
  - 16位整数输入的 32767 和 -32768
- 输出边界
  - 屏幕上光标在最左上、最右下位置
  - 报表的第一行和最后一行
- 内部边界
  - 数组元素的第一个和最后一个
  - 循环的第 0 次、第 1 次和倒数第 2 次、最后一次

排序程序的边界值

- Q: 有哪些边界值？

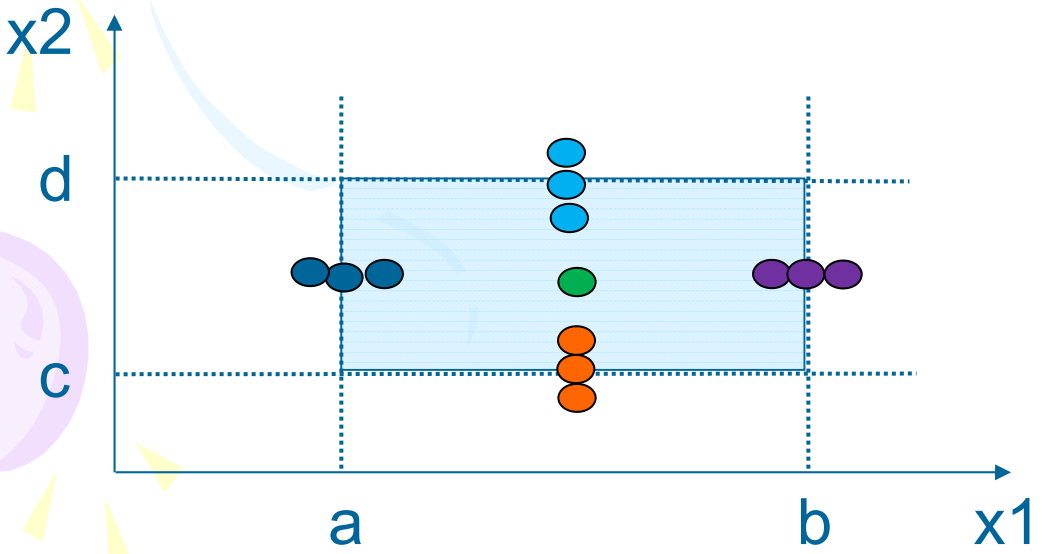
排序程序的边界值

- 输入特征
  - 排序序列为空；
  - 排序序列仅有一个数据；
  - 排序序列满；
- 输出/行为特征
  - 排序序列已经按要求排好序（未有动作）；
  - 排序序列的顺序与要求的顺序恰好相反（极端动作）；
- 内部特征
  - 排序序列中的所有数全部相等（比较、交换等处于边界）

数域上的输入边界测试

例1：有两个输入变量x1(a≤x1≤b)和x2(c≤x2≤d)的程序的边界值测试用例如下：

<x1nom,x2nom>,
<x1nom, c->, <x1nom,c>, <x1nom,c+>,
<x1nom,d->, <x1nom,d>,<x1nom,d+>,
<a-,x2nom>, <a,x2nom>, <a+,x2nom>,
<b-,x2nom>, <b,x2nom>, <b+,x2nom> }



单个变量测试用例

- 略小于最小值(min-)
- 最小值(min)
- 略大于最小值(min+)
- 输入值域内的任意值(nom)
- 略小于最大值(max-)
- 最大值(max)
- 略小于最大值(max+)

如测试资源充裕，应考虑多变量同时取边界得组合（组合数量可能比较大）

边界值法的局限

- 系统性略差，结合等价类法更好
- 更多地基于“单故障”假设
  - 两个或更多故障同时出现而导致软件失效的情况很少，也就是说，软件失效基本上是由单故障引起的。
- 不易测试多个输出都不处在边界，但组合后处于变化极点的情况
  - a+b<=c, 刚好=的情况即一种边界

5.5 正交试验设计法（正交测试）

- 对于因果、逻辑关系相对简单，或者不清晰，但组合关系复杂的程序可采用正交试验设计法生成测试用例
- 正交试验设计法是从大量的数据中挑选适量的、有代表性的点，从而合理地安排测试的一种科学的试验设计方法
- 正交测试法使用已经造好了的正交表格来安排试验并进行数据分析的一种方法
- 它简单易行并且计算表格化，应用性较好。特别适用于输入为枚举值的情况。

正交试验设计案例：确定最佳生产条件

- 为提高某化学产品的转化率，选择了三个有关因素进行条件试验：反应温度(A)，反应时间(B)，用碱量(C)，并确定了它们的试验范围如下：
  - A：80 —— 90℃；
  - B：90 —— 150分钟；
  - C：5%—— 7%。
- 为控制实验代价，A、B、C三个因素均取相关范围内3个代表性的值来进行测试

A：A1＝80℃，A2＝85℃， A3=90℃
B：B1＝90分，B2＝120分，B3=150分
C：C1＝5%， C2＝6%， C3＝7%

全面试验法取点

- 试验值

A：A1＝80℃

A2＝85℃

A3=90℃

B：B1＝90分

B2＝120分

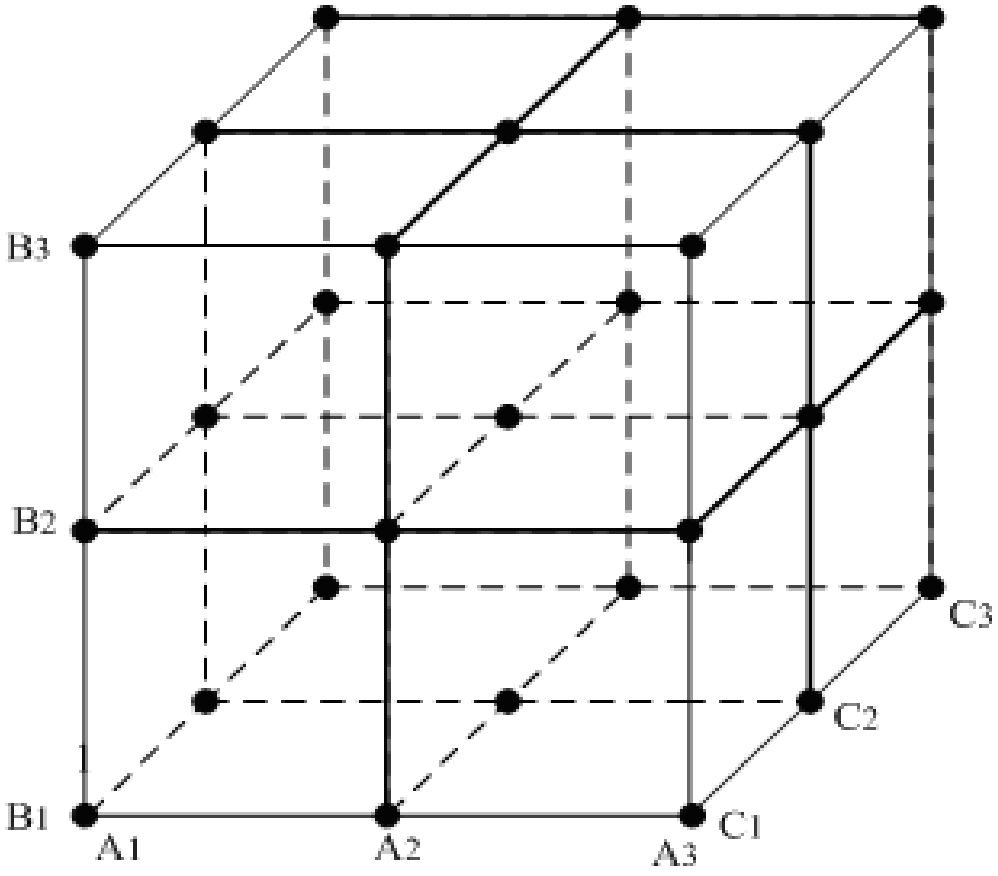
B3=150分

C：C1＝5%

C2＝6%

C3＝7%

取点太多



全面试验法取点

简单对比法

- 试验值

A：A1＝80℃
　　A2＝85℃
　　A3=90℃
B：B1＝90分
　　B2＝120分
　　B3=150分
C：C1＝5%
　　C2＝6%
　　C3＝7%

↗A1
B1C1 →A2
↘A3（好结果）

如得出结果 A3 最好，则固定 A 于 A3，C 还是 C1，使 B 变化：

↗B1
A3C1 →B2 （好结果）
↘B3

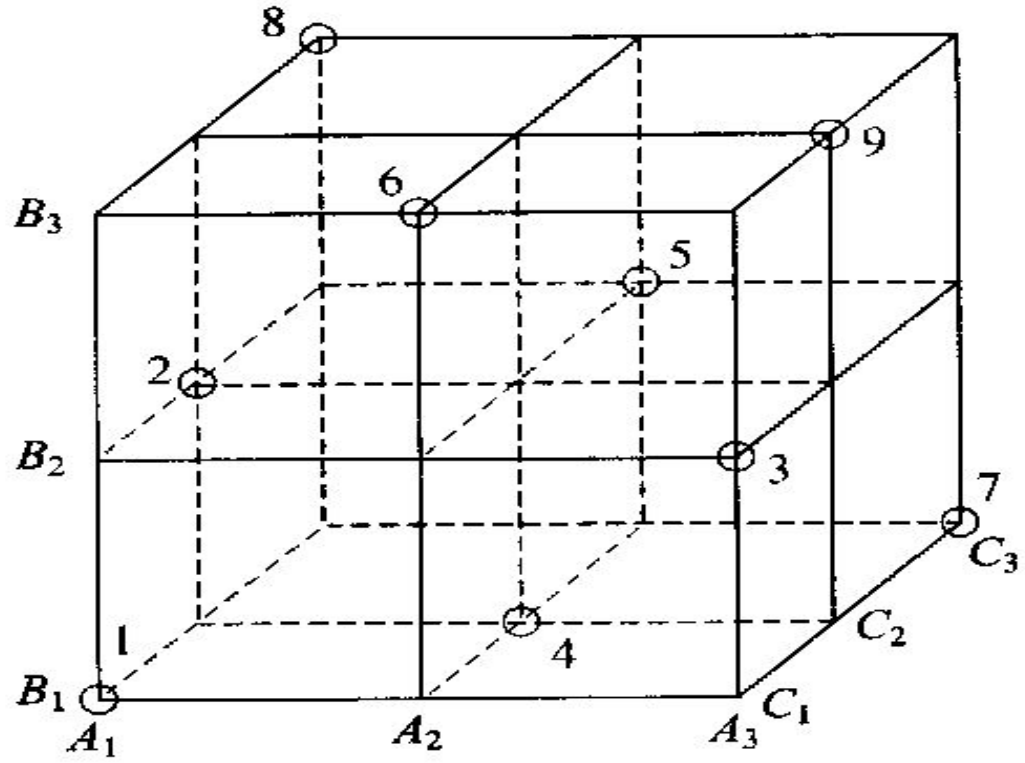
得出结果以 B2 为最好，则固定 B 于 B2，A 于 A3，使 C 变化：

↗C1
A3B2→C2 （好结果）
↘C3

- 试验次数少，但容易取点不均匀，造成试验结论不稳定

正交试验设计法

- 全面、均匀，测试点较少，且具有典型性、代表性
  - 九个平面上的试验点都一样多（每个因子同等看待）
  - 每行、每列上的点一样多（每个水平同等看待）



正交表

- 当因子数和水平数都不太大时，可通过作图的办法来选择分布均匀的试验点。但是因子数和水平数多时，作图的方法就难以施行，为此创造出了所谓的正交表。

- 一般用 L 代表正交表，常用的有
  - $L_8(2^7)$
  - $L_9(3^4)$
  - $L_{16}(4^5)$
  - ……

- 正交表可查询数学手册获得

$L_8(2^7)$正交表

- 其中，7 为此表列的数目（最多可安排的因子数）；2为因子的水平数；8为此表行的数目（试验次数）。

	f1	f2	f3	f4	f5	f6	f7
1	1	1	1	1	1	1	1
2	1	1	1	2	2	2	2
3	1	2	2	1	1	2	2
4	1	2	2	2	2	1	1
5	2	1	2	1	2	1	2
6	2	1	2	2	1	2	1
7	2	2	1	1	2	2	1
8	2	2	1	2	1	1	2

- 覆盖每个因素的每个水平
  - 充分测试
- 每个因素的每个水平出现的次数是完全相同的。
  - 均匀测试
- 对任意两个因素，覆盖状态水平组合，且个组合次数相同
  - 充分、均匀

案例——试验方案

$L_9(3^4)$

列号 / 行号	A 1	B 2	C 3	 4
1	1	1	1	1
2	1	2	2	2
3	1	3	3	3
4	2	1	2	3
5	2	2	3	1
6	2	3	1	2
7	3	1	3	2
8	3	2	1	3
9	3	3	2	1

⇨

试验号	水平组合	试验条件		
		温度（℃）	时间（分）	加碱量（%）
1	$A_1B_1C_1$	80	90	5
2	$A_1B_2C_2$	80	120	6
3	$A_1B_3C_3$	80	150	7
4	$A_2B_1C_2$	85	90	6
5	$A_2B_2C_3$	85	120	7
6	$A_2B_3C_1$	85	150	5
7	$A_3B_1C_3$	90	90	7
8	$A_3B_2C_1$	90	120	5
9	$A_3B_3C_2$	90	150	6

正交测试测试用例设计步骤

- 分解功能说明，发现因子
  - 影响软件行为的输入因素
- 确定各因子的状态（水平），构造因子-状态表
  - 取值较广时，状态需采样，可采样典型值、边界值等
- 筛选因子和状态
  - 因子、状态较多时，可能影响测试效率，必要时，要对因子和状态进行筛选。可通过权重控制实现
- 利用正交表构造测试数据集

实例

- 测试个人信息查询系统。
  - 3个输入因子：姓名、身份证号码、手机号码
  - 每个因子的状态有两个：填与不填。

选择正交表依据：

- 正交表中的因素数 >=3；
- 正交表中至少有3个因素数的水平数 >=2；
- 对符合条件1、2的正交表，取行数最少的一个。

在正交表系统中查找，结果：

- $L_4(2^3)$

生成测试用例

- 替换正交表中的因素和水平

		列 号		
行号		1	2	3
	1	0	0	0
	2	0	1	1
	3	1	0	1
	4	1	1	0

0--填　　1--不填

	列　号		
行号	姓名	身份证号	手机号码
1	填	填	填
2	填	不填	不填
3	不填	填	不填
4	不填	不填	填

生成测试用例

- 生成测试用例：
  - 1：填姓名、填身份证号、填手机号
  - 2：填姓名、不填身份证号、不填手机号
  - 3：不填姓名、填身份证号、不填手机号
  - 4：不填姓名、不填身份证号、填手机号

- 不足： 正交表不保证覆盖两个以上的因素组合
  - 补充用例：不填姓名、不填身份证号、不填手机号

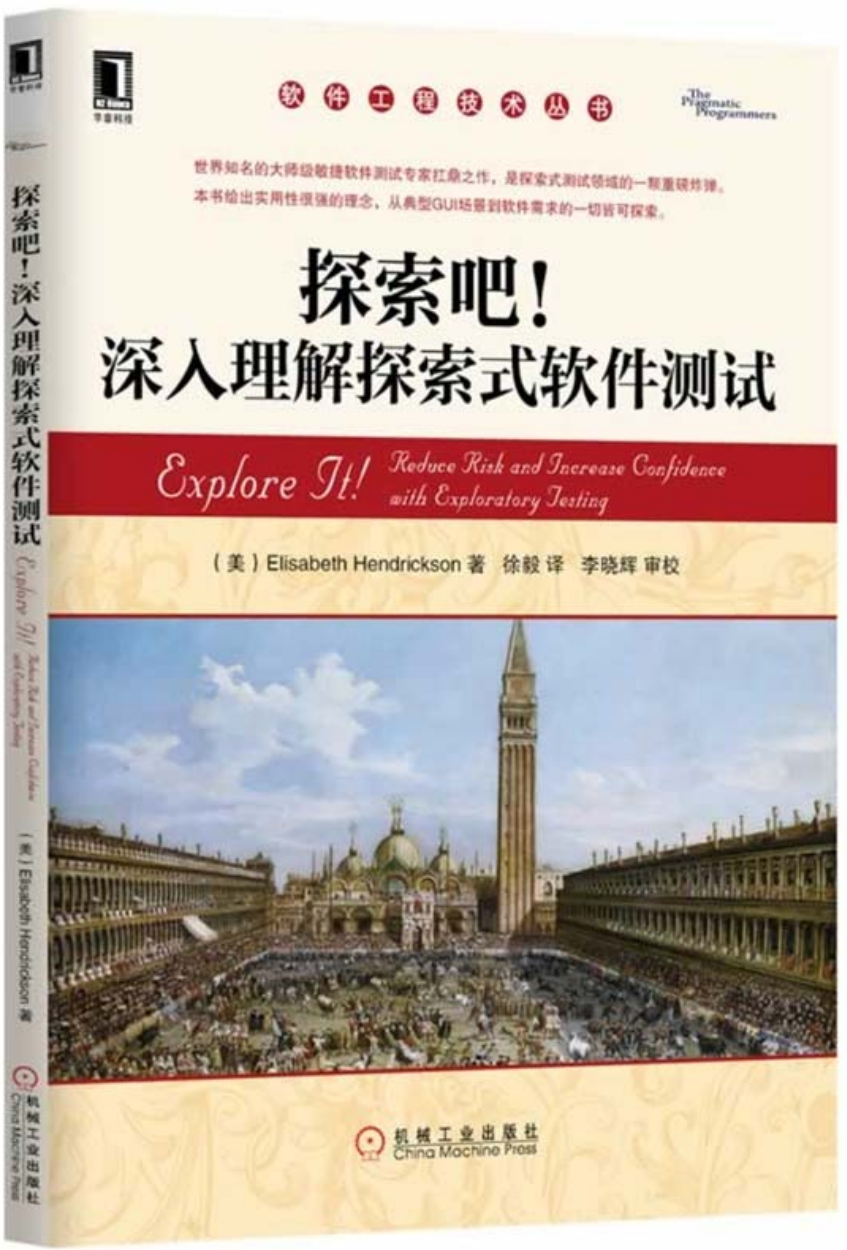
正交测试小结

- 优点
  - 节约测试工作工时
  - 可控制生成的测试用例的数量
  - 测试用例具有一定的覆盖率
- 缺点
  - 需要数学工具支持
  - 忽略因素间的内在联系
  - 不保证覆盖两个以上的因素组合

https://github.com/Microsoft/pict

5.4 错误推测与探索式测试

- 错误推测
  - 基于经验和直觉推测程序中所有可能存在的各种错误，从而有针对性的设计测试用例
- 探索式测试
  - 边学习、边设计、边测试、边思考
  - 大量借助对风险的推测


软件工程技术丛书
The Pragmatic Programmers
世界知名的大师级敏捷软件测试专家扛鼎之作，是探索式测试领域的一颗重磅炸弹。
本书给出实用性很强的理念，从典型GUI场景到软件需求的一切皆可探索。
探索吧！
深入理解探索式软件测试
Explore It!
Reduce Risk and Increase Confidence with Exploratory Testing
（美）Elisabeth Hendrickson 著 徐毅 译 李晓辉 审校
机械工业出版社
China Machine Press

探索式测试方法

- 商业区测试类型
  - 指南测试法：要求测试人员通过阅读用户手册并严格遵照手册的建议执行操作
  - 卖点测试法：找到那些最能卖钱的特性，应该观摩那些销售演示
  - 地标测试法：选择一个地标，到达目标后，再选择另一个地标
  - 极限测试法：向软件提出很多难以回答的问题
  - **快递测试法：测试人员专注于数据，确认那些被存储起来的输入数据，并跟随它们走遍软件**
  - 深夜测试法：对于应用程序自动做的事情有时可以强制去让他执行
  - 遍历测试法：选择一个目标，使用可以发现的最短路径来访问目标包含的所有对象

- 历史区测试类型
  - 恶邻测试法：缺陷通常扎堆，某个代码区域缺陷多，可以对邻近功能使用遍历测试法进行测试
  - 博物馆测试法：那些老代码或者接受重新修改，或者是没被改动就放到新环境中运行，很容易发生失效的情况
  - 上一版测试法：必须运行先前版本上支持的所有场景和测试用例

- 娱乐区测试类型
  - 配角测试法：越紧邻那些主要功能，越容易被人注意，这些特性不能忽视
  - 深巷测试法：应该测试该使用情况中排在最下面的几项特性
  - 通宵测试法：软件能多长时间运行而不崩溃，将软件一直处于开启状态
- 旅游区测试类型
  - 收藏家测试法：收集软件的输出，确保能观察到软件能生成的任何一个输出
  - 长路径测试法：到达目的地前尽量多地在应用程序中穿行
  - 超模测试法：只是测试界面，测试中注意观察界面上的各种元素
  - 测一送一测试法：同时打开同一文件或者让它们同时在网络上传输数据
  - 芬格兰酒吧测试法：有些功能很难找到，需要花大量的时间深入了解待测的应用程序
- 旅馆区测试类型
  - 取消测试法：可以对任何提供取消选项的功能或者需要较长时间才能完成的功能做同样的操作
  - 懒汉测试法：测试人员做尽量少的实际工全，接受所有默认值，保持表单为空等
- 破旧区测试类型
  - 破坏测试法：强迫软件做一些操作，掌握软件成功完成操作必须使用的资源，在不同程度上移除那些资源或限制用那些资源
  - 反判测试法：要求输入最不可能的数据，或者已知的恶意输入
  - 强迫症测试法：一遍又一遍地输入同样的数据

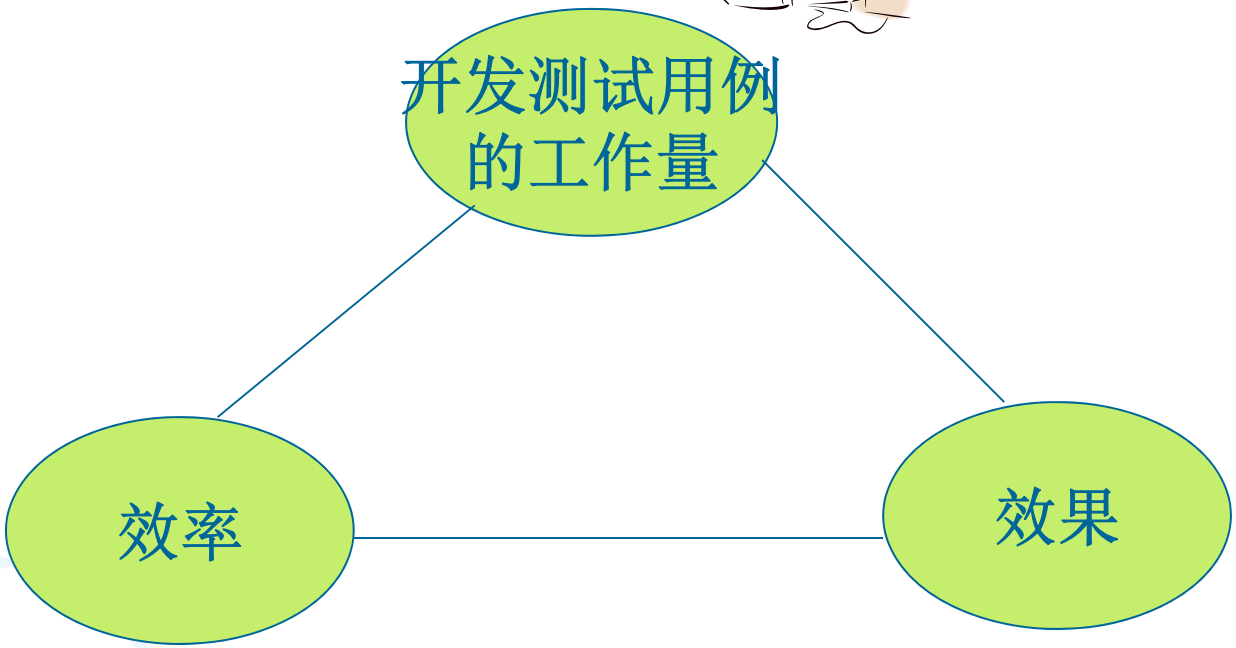
各方法的特点

- 随机方法适用与没有其它信息可用，或者不想受既有信息约束的情况
- 等价类是一种比较系统化的方法
- 边界值方法揭示错误能力强，但不够系统
- 正交测试适用与输入是离散数据组合的情况
- 因果图法、决策表法适用与输入条件存在复杂的组合关系、输入条件之间的相互制约的情况。

主要内容

1. 黑盒测试的概念
2. 测试需求的描述
3. 测试规格的描述
4. 从需求规格构造测试用例
5. 其它常用测试技术
6. 测试方法的选择

6 测试方法的选择



覆盖的问题/测试用例数

测试用例覆盖的问题

总体原则

- 分析风险：根据程序的重要性和一旦发生故障将造成的损失来确定测试等级和测试重点

- 认真选择测试策略，以便用尽可能少的测试用例，发现尽可能多的程序错误
  - 测试不充分，程序中遗留的错误过多并且严重，意味着让用户承担隐藏错误带来的危险
  - 测试过度又会带来资源的浪费。
  - — 测试需要找到一个平衡点

一些参考原则

（1）在任何情况下都必须采用边界值分析法。这种方法设计出的测试用例发现程序错误的能力最强。

（2）必要时采用等价类划分法补充测试用例。

（3）采用随机或错误推断法再追加测试用例。

（4）对照程序逻辑，检查已设计出的测试用例的覆盖程度。如未达覆盖标准，则应当再补充更多的测试用例。

（5）如果输入是逻辑量、包含诸多条件或程序的功能说明中含有输入条件的组合情况，则应一开始就选用因果图法或决策表法。

（6）对于业务流清晰的系统，可利用场景法贯穿整个测试案例过程，在案例中综合使用各种测试方法。

测试资源往往受限，一般先考虑效率高的方法，再考虑效果好的方法