XX 银行测试方案

上海泽众软件

2011/12/8

目录

1. 项目概述.....5

1.1. 内容与目标.....5
1.2. 术语定义.....5
2. 测试管理咨询服务方案.....5
2.1. 实现目标.....5
2.2. 实施内容.....5
2.3. 测试体系介绍.....6
2.4. 测试体系的优势.....7
2.5. 测试体系概述.....8
2.6. 测试方法论概述.....8
2.6.1. 测试场景分析法.....9
2.6.1.1. 设计需求场景.....9
2.6.2. 模型驱动设计法.....12
2.6.2.1. 整体架构.....12
2.6.2.2. 测试设计过程.....13
2.6.2.3. 自动执行.....14
2.6.2.4. 提交缺陷.....15
2.6.3. 测试案例自动生成.....15
2.6.3.1. 自动生成测试案例的目的.....15
2.6.3.2. 相关资料.....16
2.6.3.3. 脚本转换程序.....16
2.6.3.4. 转换规则.....17
2.7. 现状分析.....18
2.8. 测试体系建设的总体思路.....18
2.8.1. 测试内容补充.....19
2.8.2. 初步模型选型.....19
2.8.3. 引进有效的测试方法.....20
2.8.4. 建立规程与标准.....20
2.8.5. 发展方向与前景.....23
2.9. 体系建立.....23
2.9.1. 建设目标.....23
2.9.2. 体系组成.....24
2.9.3. 项目管理过程.....24
2.9.4. 流程与控制.....28
2.9.5. 项目测试标准.....35
2.10. 测试体系涵盖的其它内容.....38
2.10.1. 规范和强化测试子流程.....38
2.10.2. 规范和强化测试流程.....39
2.10.3. 标准化、规范化测试对象.....40
2.10.4. 测试对象复用，降低测试成本.....40
2.10.5. 建基于模型驱动的自动化测试架构.....40
2.10.6. 定制流程管理缺陷，定制查询.....41
2.10.7. 生成测试报告.....42
2.10.8. 测试环境管理.....42
2.10.9. 和版本管理工具集成，实现测试资源版本管理.....42

3. 测试管理平台方案.....43
3.1. 产品介绍.....43
3.2. 测试管理流程.....45
3.2.1. 准备阶段.....45
3.2.2. 需求导入与测试需求分析.....46
3.2.3. 测试案例设计.....49
3.2.4. 建立测试计划与执行.....59
3.2.5. 测试结果评估.....61
3.2.6. 测试度量.....65
3.3. 自动化测试支持.....66
3.4. 扩展接口说明.....68
3.4.1. 与第三方管理平台对接.....68
3.4.2. 与邮件系统对接.....69
3.5. 技术参数与功能响应.....70
3.5.1. 流程需求响应表.....70
3.5.2. 非流程需求响应表.....76
3.5.3. 非功能需求响应表.....79
4. Win32 自动测试工具方案.....84
4.1. 自动测试的意义.....84
4.2. 自动测试的要素.....84
4.3. AutoRunner 产品介绍.....85
4.4. 产品特点.....87
4.5. 技术参数与功能响应.....88
4.5.1. 功能需求响应表.....88
4.5.2. 非功能需求响应表.....97
5. 终端自动测试工具方案.....98
5.1. 产品介绍.....98
5.2. 技术参数与功能响应.....100
6. 测试平台建设方案.....101
6.1. 搭建环境.....101
6.1.1. 硬件环境.....101
6.1.2. 软件环境.....101
6.2. 网络拓扑图.....101
6.3. 用户角色.....102
6.3.1. 角色规划.....102
6.3.2. 角色配置.....103
6.4. 多项目（资源复用）.....105
6.4.1. 管理多个项目.....105
6.4.2. 测试案例复用.....107
6.4.3. 测试组件复用.....108
6.5. 备份与恢复.....110
7. 项目实施方案.....110
7.1. 阶段划分.....110
7.2. 时间计划.....111

7.2.1. 项目整体时间计划.....111
7.2.2. 实施内容.....112
7.3. 项目管理办法.....113
7.3.1. 组织架构图.....113
7.3.2. 岗位职责.....113
7.3.3. 沟通管理.....114
7.4. 风险管理.....115
7.4.1. 本项目风险.....115
7.4.2. 应对策略.....116
7.5. 变更管理.....116
7.5.1. 变更流程.....116
7.5.2. 变更的申请.....117
7.5.3. 变更的评定.....117
7.5.4. 变更的认可.....117
7.5.5. 变更的实施.....118
7.5.6. 变更申请表.....118
7.6. 里程碑与交付物.....118
8. 培训方案与售后服务.....119
8.1. 产品培训计划.....119
8.1.1. 培训目标和范围.....119
8.1.2. 培训安排.....120
8.1.3. 培训课程提纲.....121
8.2. 售后服务.....124
8.2.1. 技术支持方式与范围.....124
8.2.2. 响应时间与工具支持.....124
8.2.3. 工具在测试体系中的支持程度.....124
9. 版权声明.....125

1. 概述

1.1. 内容与目标

本项目的内容是为 XX 银行提供测试管理咨询服务，并在 XX 银行测试流程中引入测试管理平台以及自动化测试工具。目的在于帮助 XX 银行规范测试流程，运用先进的测试技术，加强系统测试力度，进而提高系统开发质量。

1.2. 术语定义

测试体系——用于指导软件测试组织实施测试活动的指示集合和工具集合，包括：测试模型，一系列测试规程、测试指南、测试模板。
TestCenter——泽众软件自主研发的测试管理平台，全面支持测试管理，包括：测试计划、设计、执行、评估、自动化，还包括一系列辅助工具，包括：案例设计插件、数据池、自动化框架。
AutoRunner——泽众软件自主研发的面向 win32 的自动化测试工具，可与 TestCenter 对接。
验证点——自动化测试工具中用于检验运行结果是否满足某个条件的工具。
Terminal AutoRunner——泽众软件自主研发的面向 Unix、Linux 字符终端的自动化测试工具，可以支持 VT100、VT220、5250 等多种协议，与 AutoRunner 功能对应，已与 TestCenter 对接。
终端模拟器——负责模拟字符终端图形环境，可以允许用户在字符终端界面上操作并录制脚本。

2. 测试管理咨询服务方案

2.1. 测试体系介绍

测试体系将测试工作共分为：

- 组织与项目管理、
- 测试生命周期模型、
- 测试过程管理、
- 测试技术与方法

四个部分。

“组织与项目管理”是贯穿在整个测试活动中的，包括各个部门、角色如何有效的协同工作。

“测试生命周期模型”决定了测试的阶段划分和测试流程标准。一次测试活动的开展首先应当确定该活动是适合于哪个类型的测试生命周期模型，针对不同的生命周期模型对测试阶段和测试过程进行相应的裁减、归并，以达到最佳的测试效果。常用的测试生命周期模型有“V 模型”、“瀑布模型”、“增量模型”、“进化模型”、“迭代模型”5 个模型。我们应该针

对具体的项目活动采用合理的测试模型建立测试体系。

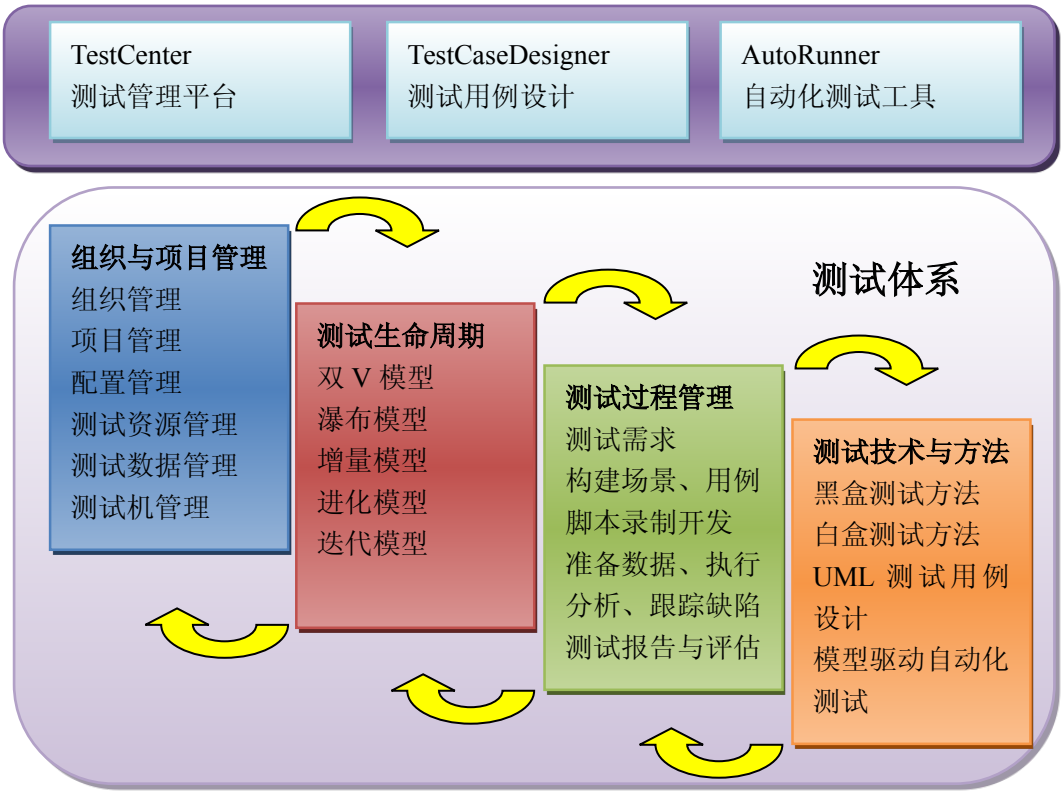
“测试过程管理”实现了测试生命周期模型。测试过程定义测试的目标、范围以及测试入口和出口，同时也描述了符合测试生命周期模型的测试流程。测试过程管理通过定义一系列的标准来衡量测试的结果，控制测试的流程。例如：设置测试需求、案例和测试结果评审标准、设置测试结果度量标准和质量记录标准等等。通过测试过程管理有效地控制各个测试阶段的质量，从而提高测试整体质量。

“测试的基本方法”有通过测试和失败测试。通过测试是确认软件具有哪些功能，能做些什么，而不考验其能力的测试；失败测试纯粹为了破坏软件而设计和执行的测试案例，也称为迫使出错测试，蓄意攻击软件的薄弱环节。在设计和执行测试案例时，首先要进行通过测试，在破坏性试验之前看看软件基本功能是否实现是很重要的。

测试运行时可以在测试生命周期中运用先进的测试技术和方法来提高测试的效率和质量，例如：可以在回归测试阶段运用自动化测试，可以实现在很短的时间内完成大规模全量回归；还可以使用模型驱动测试案例设计改善自动化测试案例难以维护的弊端，实现自动化测试前移到系统测试阶段。

使用优良的测试技术与方法可以：

- 提高测试效率；
- 提高测试需求覆盖率；
- 找到更多缺陷。



测试体系示意图

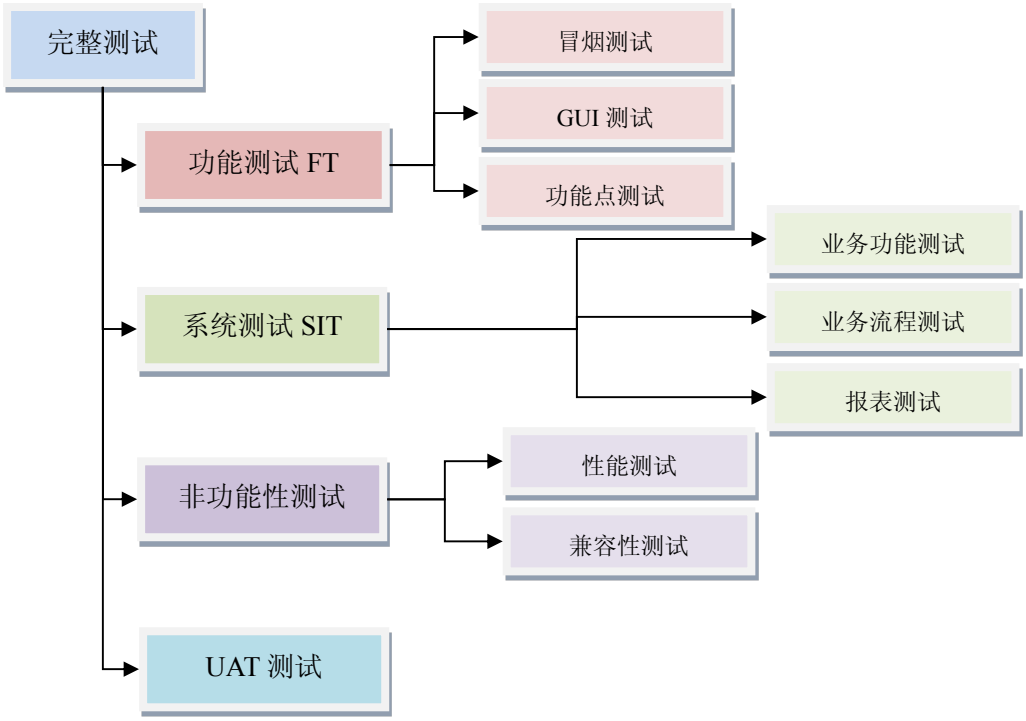
2.2. 测试体系建设的总体思路

详细描述以解决现有不足为目标并结合银行测试管理的现状而设计的测试管理总体解决方案的理念、思路、实现的方式方法。

根据上一节对 XX 银行测试工作的现状和现实环境的分析，我们了解到在行里建立符合现状和现需求的测试体系，并在该测试体系的指导下建立一批技术过硬的 IT 测试团队的必要性。本节将着重描述测试体系建设的整体规划和发展路线图。

2.2.1. 测试内容补充

为了进一步提高测试的覆盖度，保证系统质量，需要不断丰富测试的内容，使用“自底向上”的方式检验系统各个层面上的正确性和可靠性。在已有的 UAT 测试的基础上增加 FT 测试、SIT 测试以及非功能性测试，非功能性测试包含的内容有：性能测试、兼容性测试等等。



2.2.2. 初步模型选型

建立测试体系的第一步是选择适应于目前情况的测试模型。与当前情况相符合主要是指研究目前开发项目和系统的特点，其中包括：项目需求的规模，对测试周期的要求，以及项目所选择的开发模型。

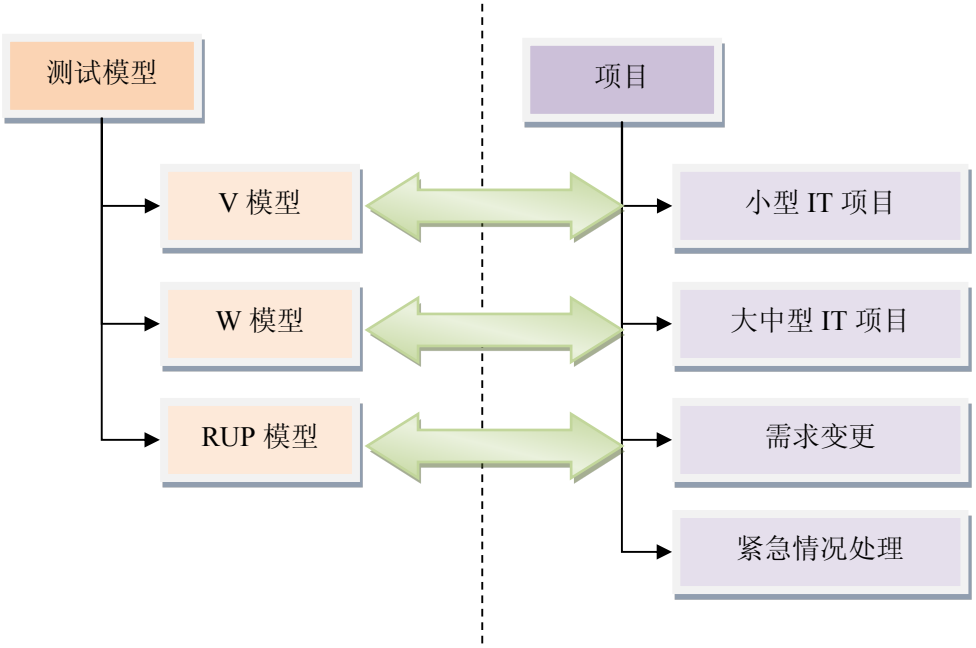
测试模型的选型目标主要是当前比较常用和成熟的测试模型：

- 瀑布模型
- V 模型
- W 模型
- 迭代模型
- 进化模型
- RUP 模型（增量迭代）

在选型过程中，需要选择多种不同的模型以满足现实中不同的开发需求，选型的方法可

以参考选择一个主模型以适应 IT 项目、一个子模型以适应新特性开发、需求变更或紧急情况应急处理。

最后，选型完成后，可根据自身的需要对模型定义的测试阶段进行删减和补充。



如上图所示，我们最终的选型目标是选出合适的测试模型与不同的项目类型相对应，在后期测试执行过程中，根据不同的项目类型进入不同的测试流程。

2.2.3. 引进有效的测试方法

通过测试体系可以在测试过程中引入一些更加行之有效的测试技术和方法，通过这些方法实现提高测试覆盖，增加测试案例数量，增强案例执行有效性等目的。这样的方法有：

- UML 模型驱动测试案例设计法
- 场景分析法
- 自动化回归测试
- 自动化数据测试
- 许多测试方法可通过引入测试工具来实现。

2.2.4. 建立规程与标准

在选择适合的测试模型后，测试活动被划分为多个测试阶段和多种针对不同测试目的的测试。例如：

- 单元测试
- 集成测试
- 功能测试（FT）
- 系统测试（SIT）
- 用户验收测试（UAT）

测试体系需通过为各种测试和阶段编写测试规程、指南来实现，另外，需要提供各种测

试文档模板用于测试过程管理。

以上各个测试阶段和各种测试属于测试体系总体所定义的内容，除此之外，测试体系还定义了总体之外的重要过程：

- 缺陷跟踪流程
- 需求变更过程
- 质量保证过程

测试体系还必须定义上述各个过程的规程和指南。

(略）上表列列举了测试体系规程指南和文档的部分内容。

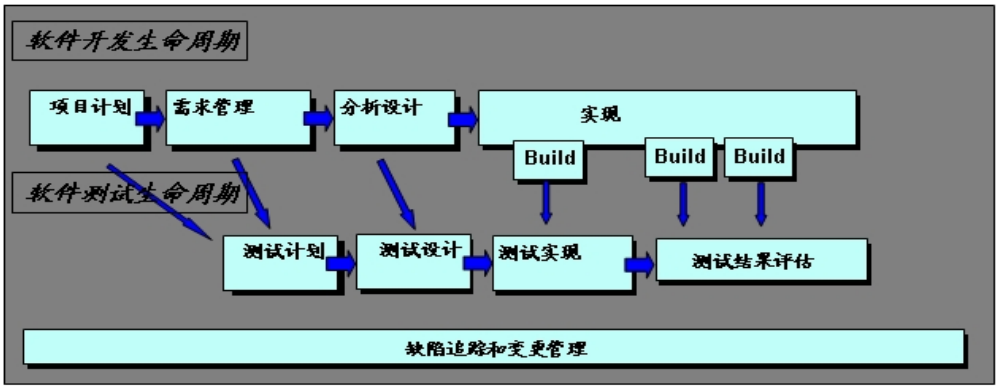
2.3. 体系建立

2.3.1. 建设目标

建立测试体系的目的是为测试工作制定周密的管理计划，为测试工作建立标准化流程和标准化文档，为测试单位提供运行的流程和规范。考虑到本项目的特点，我们知道该项目的测试工作需要横跨不同的业务系统，不同系统之间存在着网状的数据流。这种系统的复杂性为测试管理工作提出了严峻的挑战，据此我们需要通过建立测试体系的方法规范化测试流程，使得复杂的联调测试变得易于跟踪和控制，从而达到降低项目风险的目的。

建立测试体系的第一步是要确定一个生命周期模型从整体的角度描述整个项目。

我们根据项目的特点按照“V 模型”来描述整个测试活动的生命周期。“V 模型”将软件的测试活动划分为四个不同的级别：单元测试、集成测试、系统测试、验收测试。这些级别分别对应了软件开发周期的不同阶段。



V 模型表明测试不用等待代码编写出来就可以进行，软件的整个测试生命周期是与软件的开发生命周期基本平齐的过程。测试可以在需求分析阶段就可以及早开始，创建测试的准则，明确需要测试的内容。每个阶段都存在质量控制点，一旦测试准备结束，可以对此阶段进行评审形成质量控制点，当软件编码完成，即可对质量控制点进行验证，我们可以通过各种测试指标实时监控项目质量状况，提高对整个项目的控制和管理能力。

根据 V 模型，我们在集成测试之后增加功能测试。在功能测试之前增加先导测试来评估系统是否达到了进入功能测试的标准。采用黑盒的方法分别对各个系统的功能进行测试，功能测试通过后方可进行跨系统的接口测试，在系统测试阶段重点测试完整的业务流程，并保证系统之间数据传输的正确性。另外，在系统测试阶段增加性能测试和压力测试。

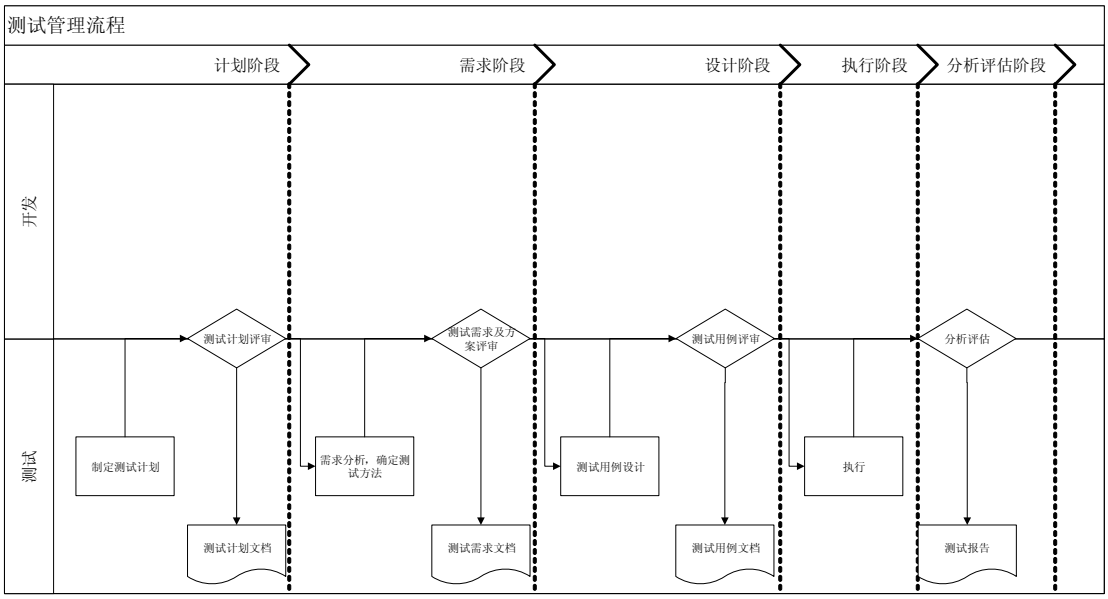
2.3.2. 体系组成

完整的测试体系将涵盖以下内容：

- 项目管理过程
- 差异分析过程
- 流程与控制
- 测试标准
- 质量评估

2.3.3. 项目管理过程

将项目的测试管理分为五个阶段和一个日常事务检查表，对每个阶段的工作任务进行说明，包括时间点、任务、提交物等。提供该体系给项目管理人员作为测试项目管理手册，对整个项目的测试工作进行系统的管理、监督。



阶段划分与任务说明：

2.3.4. 流程与控制

该体系是针对项目具体实施过程的，对大运会项目的测试过程实施，在各个里程碑阶段，我们将使用以下体系进行项目测试过程的执行，包括：里程碑接口、里程碑输入信息、参与角色、工作过程、工作内容、输出信息等。

1) 初始阶段

初始阶段主要是给客户做测试过程和测试标准的介绍，加强客户对测试过程和测试标准的了解。

面向对象：对象为项目参与人员（包括管理人员和技术人员）。

介绍内容：

- 介绍测试过程

- ✧ 介绍测试策略
- ✧ 介绍测试方法和特点
- ✧ 介绍测试结果评估、分析方法
- ✧ 需求分析阶段

前期接口：

- ✧ 初始阶段完成，项目组认可所使用的测试过程、方法等；
- ✧ 就基本的测试范围（功能测试、性能测试、自动化测试等）和使用何种测试工具等基本达成一致。

输入：

- ✧ 被测系统的开发文档
- ✧ 被测系统的客户文档

参与角色：

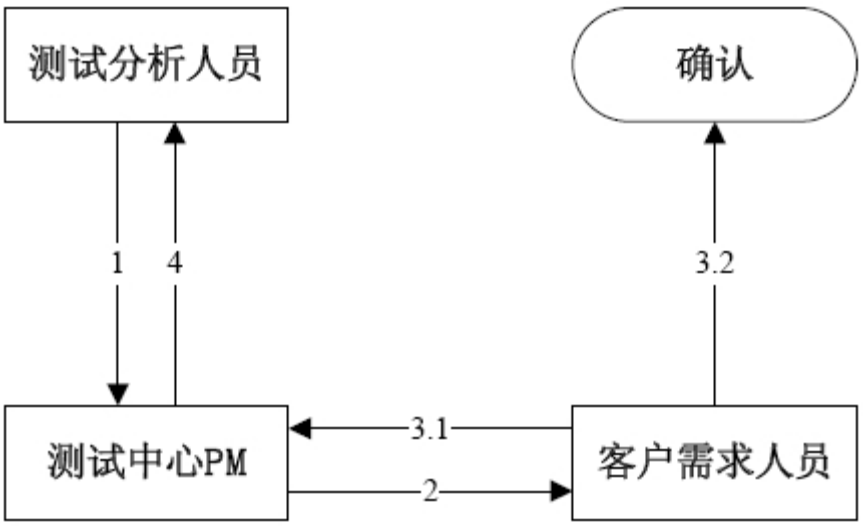
在测试项目中，开发厂商，测试专家，和测试组都有众多人员的参与，这里阐述了各方在项目中需要的角色和各自的职责。

2.3.5. 项目测试标准

1) 需求评审标准

测试需求的评审是一个贯穿测试需求制定的过程，在这个过程中，需要开发部，业务部人员的全程参与，对测试组测试设计人员制定测试需求提供咨询，并对制定了的测试需求进行需求评审并确认：

需求评审过程：



需求评审流程图

2) 缺陷相关标准

严重级别：

5 紧急

- ✧ 导致操作系统崩溃（如 Win NT/2000 的篮屏、Win 98 的系统致命错误等）
- ✧ 导致操作系统不响应
- ✧ 程序退出没有释放资源
- ✧ 导致其它应用程序出现异常（如无法启动、不响应、异常退出）
- ✧ 卸载时不提示客户确认即删除公用程序（DLL 等）
- ✧ 其它导致操作系统或其它应用程序异常的情况
- ✧ 造成重大安全隐患情况（如机密性数据的泄密）

4 很高

- ✧ 程序挂起
- ✧ 程序异常退出
- ✧ 系统无法正常安装、卸载或升级
- ✧ 其它导致被测系统本身出现无法正常运行的错误

3 高

- ✧ 导致输出的数据错误（数据内容出错、格式错误、无法打开等）
- ✧ 导致其它功能模块无法正常执行，如：
- ✧ 功能不完整或功能实现不正确；
- ✧ 导致数据最终操作结果错误
- ✧ 文件或数据传输不完整或不正确
- ✧ 对数据格式不进行检测
- ✧ 提示语句易误导用户，造成数据丢失等重大问题
- ✧ 其它导致被测应用系统其它模块无法正常运行或出现错误结果的情况

2 中等

- ✧ 影响当前操作结果
- ✧ 数据修改后没有保存提示
- ✧ 系统出错提示不正确或没有捕获系统出错信息
- ✧ 数据的重要操作（如删除、添加等）没有提示
- ✧ 其它影响被测模块/功能正常执行的情况

1 低

- ✧ 页面布局不合理
- ✧ 字体不一
- ✧ 错别字
- ✧ 语言不一致（如：中英文混合）
- ✧ 页面提示不明确
- ✧ 系统易用性不好
- ✧ 其它对被测模块功能实现没有影响的情况

3) 缺陷导入阶段

- ✧ 需求阶段
  - ➢ 未能真正了解客户需求，功能描述不正确
  - ➢ 需求定义有二义性
  - ➢ 需求中遗漏客户功能需求
- ✧ 概要设计阶段
  - ➢ 架构设计不正确
  - ➢ 业务流程设计错误

- ✧ 详细设计阶段
  - ➢ 功能模块间数据格式定义不一致
  - ➢ 开发规范
- ✧ 编码阶段
- ✧ 其它

4) 缺陷优先级：

- ✧ 3 必须修改
- ✧ 2 将要修改
- ✧ 1 有时间则改
- ✧ 0 未分配

5) 缺陷类型：

- ✧ 程序错误
- ✧ 环境设置
- ✧ 重复记录
- ✧ 需要完善
- ✧ 不可重现
- ✧ 并非问题

3. 测试管理平台方案

3.1. 产品介绍

TestCenter 是上海泽众软件科技有限公司自主研发的拥有自主知识产权的测试管理软件平台。它是采用 JAVA 实现的 B/S 架构应用程序，可以部署于不同的操作系统平台上。测试人员可以通过浏览器访问测试平台提供的功能。

TestCenter 可以和本公司的测试工具 AutoRunner、Terminal AutoRunner 系列集成，也可以和其他测试工具集成，提供强大的自动化功能测试。

TestCenter 面向的用户是所有需要提高软件开发质量的软件公司、软件外部企业，以及提供测试服务的部门。

使用 TestCenter，可以帮助用户明确测试目标、测试需求并建立完善的测试计划；可以帮助用户掌控测试过程并建立有效地质量控制点；可以帮助用户严谨地实施测试计划并对测试全过程进行针对性评估。

1) TestCenter 的模块组成

2) 测试计划管理

支持测试计划管理、多次执行手工测试和自动化测试；测试需求范围定义、测试集定义；数据模版的导入和导出。

3) 测试需求管理

支持测试需求管理、支持测试需求树，树的每个节点是一个具体的需求，也可以定义子节点作为子需求。每个需求节点都可以关联到一个或者多个测试案例。根据需求可以创建相同名称的测试案例组；也可以通过需求向导创建关联一个或者多个测试案例的测试集。

4) 测试业务组件管理

支持测试案例与业务组件之间的关系管理，通过测试业务组件和数据“搭建”测试案例，实现了测试案例的高度可配置和可维护性。主要是添加自动测试执行时所需要的脚本（通过 AutoRunner 或者 Tar 测试工具录制），同时也支持测试人员根据需求编写不同的脚本。

5) 测试案例管理

测试案例允许建立测试主题，通过测试主题来过滤测试案例的范围，实现有效的测试。主要是搭建测试案例跟业务组件之间的关联关系，以及组件与组件之间的依赖关系。可以为手工测试设计详细测试案例。并可以对测试案例进行复制、剪切以及粘贴，从而为客户的使用提供了方便。

6) 测试集管理

通过测试案例和数据“搭建“测试集，实现了测试集的高度可配置和可维护性。主要是搭建测试案例跟测试集之间的关联关系，以及案例跟案例之间的依赖关系。可以通过运行测试集来执行自动化测试并显示运行日志。导入、导出数据模版可以查看、修改当前测试集中包含的所有测试案例的详细信息。并可以对测试集进行复制、剪切以及粘贴，从而为客户的使用提供了方便。

7) 用户角色管理

主要功能是：添加、删除、修改角色信息，并且可以模拟实际场景中的不同角色（主要体现在自动运行中），当角色绑定案例后，系统会根据不同的角色查找并执行不同的案例。

8) 系统设置管理

用户管理（添加、删除用户），项目管理（查看 TestCenter 中包含的所有项目以及项目详细信息），自定义字段管理（添加自定义字段后体现在缺陷管理中），系统权限管理（为不同的角色设置不同的权限），邮件配置（配置邮件发送服务器，在缺陷管理中修改缺陷的状态，就可以通过自动发送邮件的形式发送给相关人员），登陆历史（记录用户登录和退出系统的信息）。

9) 测试执行管理

支持测试自动执行（通过调用测试工具）；支持手工执行（手工操作的方式执行案例，来验证需求。错误时可以直接提交 bug）。

在测试计划发起手工测试成功后会显示在“手工测试”标签页中（测试计划中包含测试集，测试集中必须包含案例），点击运行名称进入详细信息界面首先分配角色（给相关测试人员）并执行测试案例，当执行测试案例失败后提交 bug 到缺陷管理中。当所有案例执行结束后会自动转移到手工日志中。

在自动化测试执行之前，需要打开两个程序 tnameserv.ext 和 TestAgent.exe。可以在测试集中发动自动化测试执行，也可以在测试计划中发动自动化测试执行，同时还可以查看自动化运行的详细信息。

10) 测试结果日志察看

具有截取屏幕的日志查看功能。

11) 测试结果分析

支持多种统计图表，比如需求覆盖率图、测试案例完成的比例分析图、业务组件覆盖比例图、缺陷严重性图、缺陷所在功能模块图等。显示每次运行的手工测试和自动化测试的案例运行图和需求运行图，在自动测试日志中会显示自动运行失败时的错误截图，测试对比报告中可以根据需求生成每次运行的对比图。

12) 缺陷管理

支持从测试错误到曲线的自动添加与手工添加；支持自定义错误状态、自定义工作流的缺陷管理过程。

报告提交测试人员发现的 bug 并可以使用过滤器和添加搜索条件的方式查询所需要的 bug。显示当前所有提交 Bug 的状态（待确认、已分配、已完成等），同时统计报表中可以统计当前所有不同状态 bug 的数量以及所在模块的分布图。

3.2. 测试管理流程

TestCenter 是一个完整的测试解决方案，其功能特性已经涵盖了测试流程的各个方面，从测试分析到测试设计再到准备数据、执行、缺陷跟踪、测试评估。以下几节简要介绍一下 TestCenter 在各个测试环节中的使用。

3.2.1. 准备阶段

在测试的准备阶段，需要构建一个一般的测试流程。该测试流程即适用于手工测试，也适用于自动化测试。

首先在需求模块中创建需求节点。选中需求节点，点击添加子节点。自动弹出添加子节点对话框，见下图。在对话框中依次填写需求信息，然后点击“确认”按钮，即可方便地创建子需求。

Test Center 使用需求树描述测试需求，用以体现测试需求之间的包含和从属关系。

相应地，在 Test Center 中增加了需求和测试案例的评审模块，可以对测试需求和测试案例进行评审。

3.2.2. 需求导入与测试需求分析

1) 需求导入

Test Center 管理的是测试需求，第一步是把 SRS 导入到测试需求模块，再进行测试需求分析工作。Test Center 提供了插件，并且定义了需求模板，用户使用这些模板，就可以把 word、excel 格式的文件轻松的导入到系统中。

需求导入的步骤：

一、 提供需求文档；

二、 提供 Test Center 需求导入文档模板；

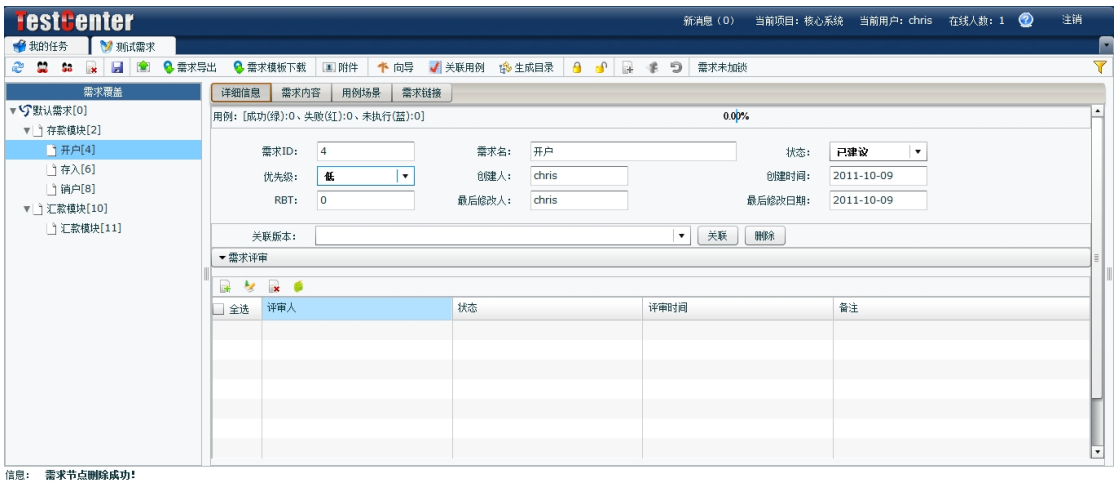
三、 应用需求导入文档模板，修改需求；

四、 导入需求；

五、 Review 导入的需求，以保证正确性。

Test Center 产品提供标准的编程接口（类和 interface），提供 java 的 jar 包和接口类说明，用户也可以使用此接口来自己编写需求导入程序。

从需求管理系统导入测试需求，并且建立需求树，见下图：



上图是 Test Center 测试需求管理的覆盖率视图。这个视图分为四个 tab 页来显示需求信息：详细信息、需求内容、关联场景、需求连接。

填写需求属性：

- ✧ 需求名称；
- ✧ 需求的描述；
- ✧ 需求的详细信息；
- ✧ 需求的附件；
- ✧ 需求的测试主题；
- ✧ 需求的种类；
- ✧ 需求的优先级；
- ✧ 需求的 RBT 值。

2) 测试需求分析

一、 根据需求的理解，进行需求分析。

二、 根据测试策略，给需求增加测试项。测试项，就是指这个需求如何被验证的项目。

三、 把需求模型根据测试项来转换为测试模型。

从测试的角度看，需求是测试的重要输入，这个输入应该具备以下几个部分：

- ✧ 业务操作过程描述；
- ✧ 本功能的数据和约束规则；

✧ 本功能涉及的活动图；

几个部分对应表：

需求功能	对应测试对象	说明
业务操作过程描述	测试脚本	
本功能涉及的活动图	场景（业务流程）	
本功能的数据和约束规则	测试数据	

3) 测试案例关联

需求节点能够关联一个或多个测试案例。

关联的测试案例用来计算需求覆盖率：当此节点关联的所有测试案例测试通过，我们就认为此需求被覆盖。否则认为没有覆盖。在执行一次测试计划之后，我们可以根据以上的规则来计算需求覆盖率。

在右边显示了选中的需求节点关联的测试案例列表，见上图。

4) 创建测试集

Test Center 支持从需求出发来创建测试集。

一次测试，都具备明确的测试目标、测试范围。对测试目标和范围的描述，都是定义自需求。测试案例的集合，称为测试集。测试集组成测试计划。

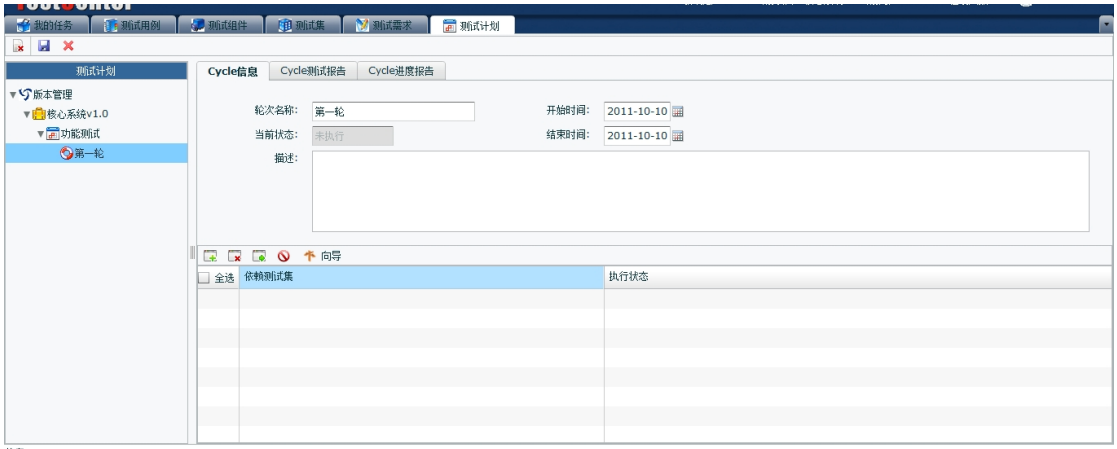
3.2.3. 建立测试计划与执行

1) 创建测试计划

测试计划是用来定义一个测试活动如何开展的对象。它包括：测试环境描述、测试目标和范围描述、具体的测试案例。

将要执行的测试集添加到一个测试计划中。

测试计划可以多次执行，并且通过比对来发现每次执行之间的差异。



信息：

TestCenter 通过固定的层次结构来组织计划。系统将构建一个树形结构，整个树分为 3 各层次：

- ✧ 测试版本
- ✧ 测试计划
- ✧ 测试轮次

测试人员需要为每个层次指定起始时间和结束时间，建立好 3 各层次后将待执行的测试集关联到测试轮次上。

2) 维护测试计划的数据场景

数据场景定义：

复杂的应用系统，往往使用数据库来存放复杂的数据信息，这些数据信息与测试案例关系紧密。当这些信息发生变化，很多测试案例就无法执行。对于自动测试而言，这是非常严重的。因为很多测试数据具有“只能使用一次”的特性。例如：一张车票，进行一次买票测试，就不能够再使用了，因为车票的状态已经改表。

为了改变这种情况，需要测试工程师把与环境相关的数据，设置为测试案例的输入参数。

我们把在一个测试集中，所有这些测试案例的输入参数集，叫做“数据场景”。

在每次执行测试之前，维护这些参数的输入值，如果保证这些输入值集合满足测试案例执行的需要，就可以提高测试案例执行的可靠性。

Test Center 支持测试工程师直接来修改这个数据场景，或者到处数据场景模板到 excel 文件中修改，然后再导入。通过这两种方法就可以：

把测试数据场景明确化，数据化；

提高测试执行的可靠性，避免由于数据场景问题反复执行一个测试计划。

计划执行完毕后，通过查看计划运行历史来快速了解执行结果。

上图，显示了测试历史会显示的统计信息。

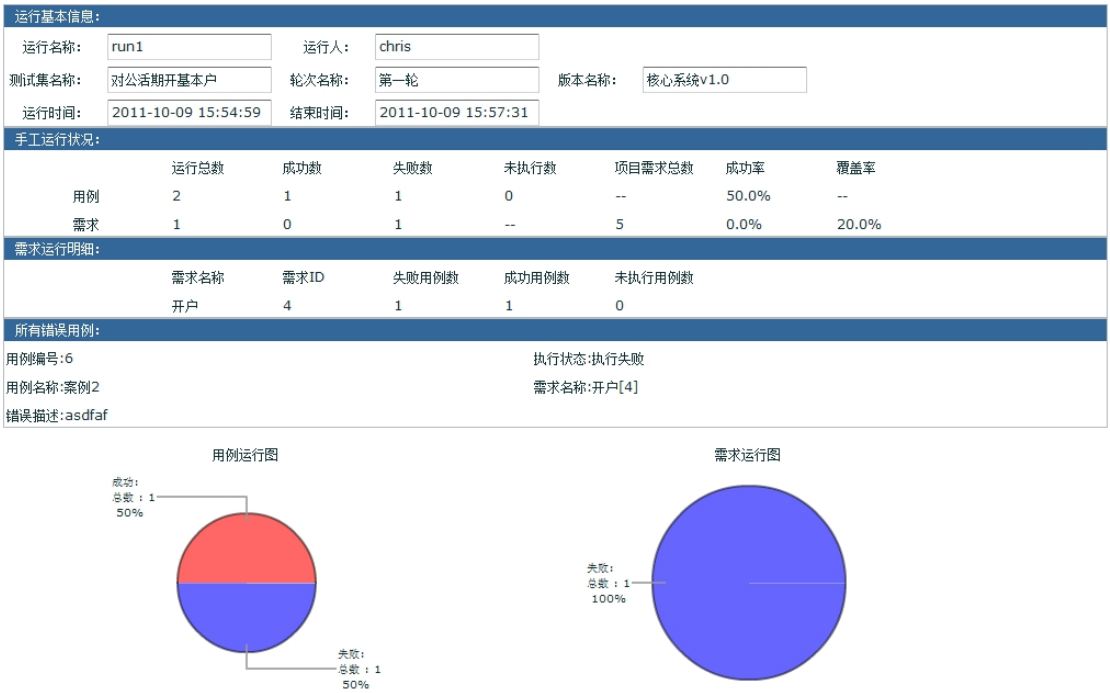
测试计划还会显示每次执行的日志信息，可以查看每个测试案例执行的日志，包括测试案例层和业务组件执行的成功/失败信息。

3.2.4. 测试结果评估

测试结果评估通过分析统计报表来实现。

1) 测试报告

在执行完毕后创建自动测试报告，使用表格和图表的形式展示执行的数据统计结果。

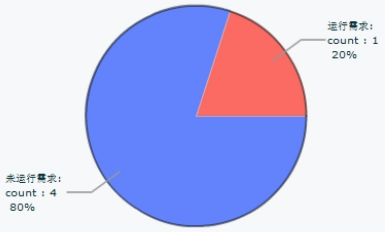


自动测试报告经过统计得到各种指标反映执行结果，包括：组件的执行成功率、案例的执行成功率、测试需求的成功率和覆盖率。

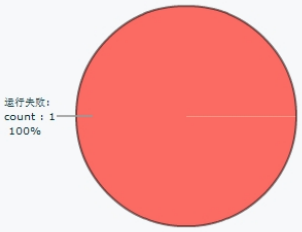
2) 测试进度报告

需求覆盖率统计

需求覆盖率图



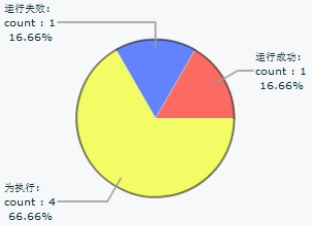
需求运行图



需求总数	运行总数	成功数	失败数	运行成功率	覆盖率
5	1	0	1	0.00%	0.20%

用例统计

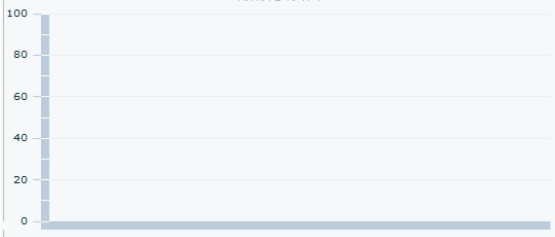
用例运行图



用例总数	执行成功数	执行失败数	未执行数	成功率
6	1	1	4	0.17%

缺陷统计

缺陷状态统计图



缺陷严重性统计图

状态统计

新建	确认	已分配	正在修改	修改完成	已驳回	重开	已关闭
0	0	0	0	0	0	0	0

严重性分布

新特性	微不足道	文字错误	不合理或别扭	次要错误	严重错误	系统崩溃	系统死锁
0	0	0	0	0	0	0	0

失败需求明细统计

需求名称	运行ID	成功用例	失败用例	未执行数	成功率
开户	4	1	1	0	

需求关联缺陷

需求名称	运行ID	缺陷个数
开户	4	0

3) 缺陷报告

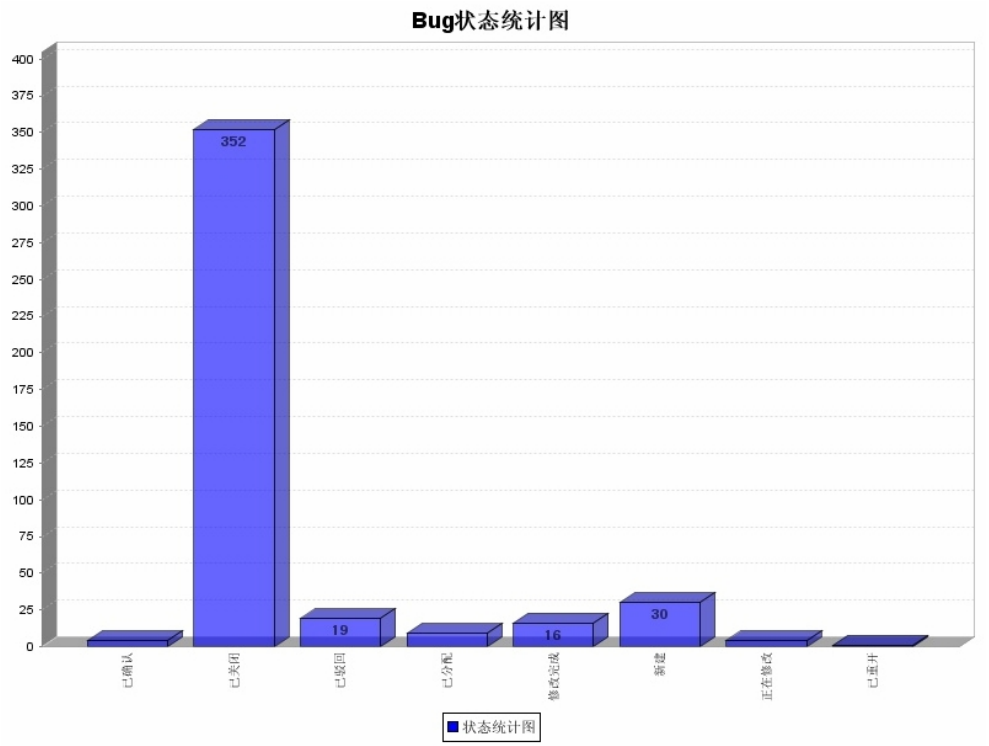
✧ 状态统计表、图

按照缺陷的各个状态统计缺陷的分布情况。通过状态统计图表，可以方便地找到解决缺陷、遗留缺陷、重开缺陷等等处在不同状态上的缺陷数量。开发人员重点关注已分配的缺陷。测试人员也可以通过这个统计图表分析缺陷的走势。

Bug 状态统计表

导出PDF

新建	已确认	已分配	正在修改	修改完成	已驳回	已重开	已关闭
30	4	9	4	16	19	1	352

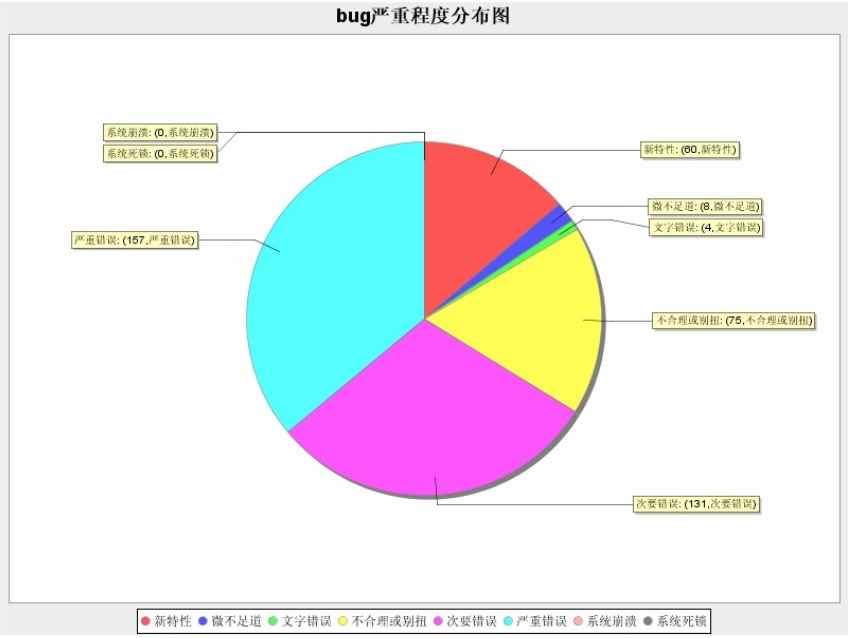


✧ 严重性统计表、图

严重性统计图、表统计不同严重性的缺陷数量。利用严重性统计图、表可以增强对严重性高的缺陷进行跟踪和解决。

缺陷严重性分布报表

新特性	微不足道	文字错误	不合理或别扭	次要错误	严重错误	系统崩溃	系统死锁
60	8	4	75	131	157	0	0



✧ 测试人员报告缺陷数

通过人员报告缺陷数统计表，了解每个测试人员报告缺陷的情况。

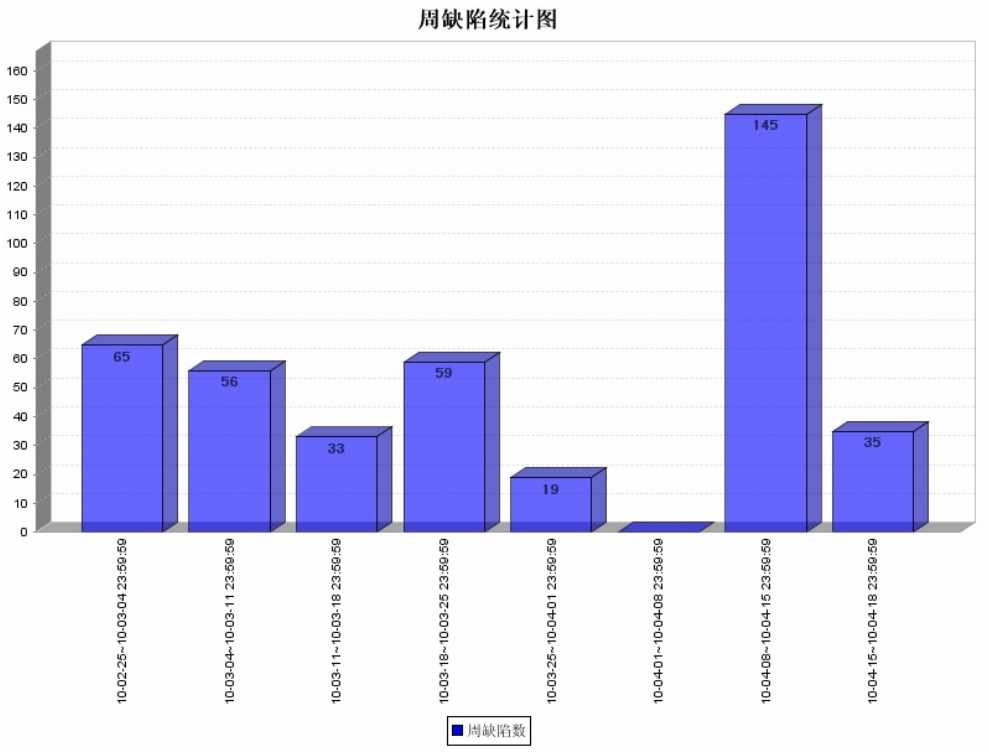
测试人员报告Bug数目统计表

导出PDF

测试人员登录名	测试人员	报告的bug数量
zhangjf	zjf	191
mark	mark	8
wxgt	wxg	59
lixs	李晓松	38
zhangpj	张配杰	42
likun	李昆	44
wangyf	王亚飞	53

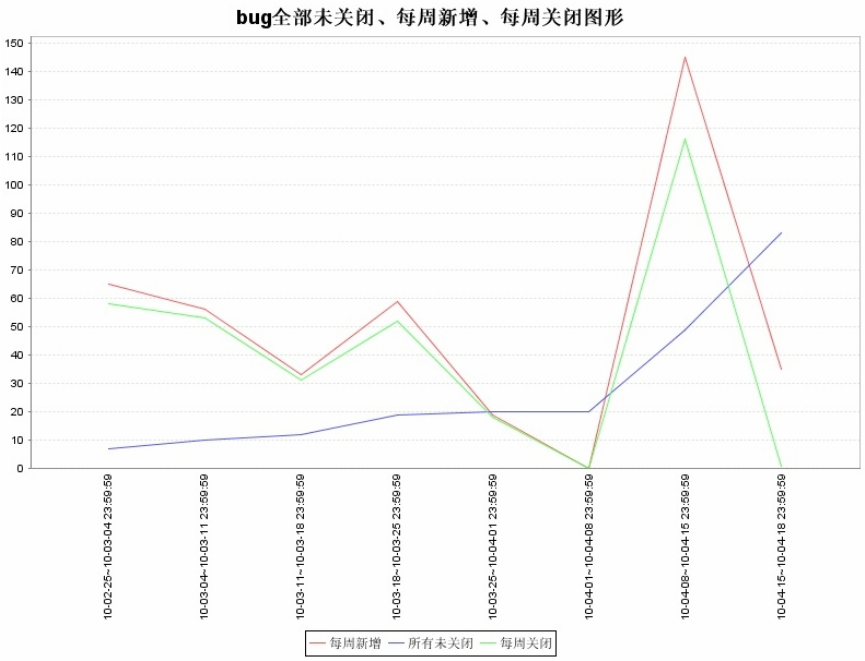
✧ 定期缺陷统计图

生成定期缺陷统计图来表示每日或每周新增的缺陷数，从而对当前系统质量做出评估。



✧ 缺陷新增、关闭图形

通过缺陷新增、关闭图形分析缺陷的发展趋势。



3.3. 自动化测试支持

组成 TestCenter 自动化测试框架的主要功能特性有：

1) 提出测试组件的概念，标识一个自动化测试脚本，允许进行参数化，并且提供了复用机制——一个自动化组件可以被不同案例根据自身实现的需要反复调用。

上图为添加组件界面，TestCenter 可以提供对多种不同自动化执行工具脚本的支持，并且实现了开放的架构，允许扩展开发对新的执行器的支持。

组件的基本信息

脚本显示

组件的参数表

2) 参数特性。参数特性包括四种，即在案例层分别实现了输入参数、输出参数、参数传递、验证点。通过这四种参数特性可以在执行的两个相邻的测试组件之间进行值

传递，并可以随时检查组件的数据。

上图显示了测试案例层实现的组件之间的四种参数特性，每个选项卡表示一种参数特征的配置。

3) 组件串联。通过案例串联组件，在组件之间设立依赖关系实现组件同步。

上图显示的是正在配置组件 c2 的情景，此时 c2 引用的真实组件式“小额往帐支票”，而 c2 依赖于组件 c1，因此 c2 当且仅当在 c1 执行成功时方可开始执行。

执行器

3.4. 扩展接口说明

TestCenter 提供多种接口，实现扩展：

- 提供二次开发 jar 包和相关二次开发说明。支持用户在 jar 包（开发包）的基础上，进行二次开发，实现外部访问、功能扩展。

- 提供数据库结构相关文档，支持通过外部系统直接访问数据库的方式来实现功能集成。

4. Win32 自动测试工具方案

4.1. 自动测试的意义

在银行软件生命周期过程中，出于金融行业竞争发展的需要、金融监管的需要等，都需要对软件进行需求变更，以满足各种需求。这样就导致需要通过自动化测试来实现在短时间内进行测试，以及通过自动化测试提升测试覆盖率。自动化测试能够帮助我们实现：

- 缩短测试周期：通过持续不断的执行，代替人工低效率的反复执行，来缩短测试周期；
- 提升测试的准确性：人工测试具有的随意性被自动测试的准确性代替；
- 提高测试覆盖率：自动测试的高效率，能够使得我们在相同的时间内使用更多的测试用例来进行测试覆盖；
- 提升测试执行的效率：对于银行交易而言，自动测试在 25 秒左右就可以执行一个用例（单个交易的用例），并且可以不间断的执行，大大提升测试效率；
- 24 小时持续执行：自动测试可以实现在夜间不间断执行，来缩短整个开发周期。

4.2. 自动测试的要素

要实现自动测试，需要以下基本条件：

- 随需而变的测试脚本和响应能力

在测试过程中，由于需求变更的频繁，需要经常性的修改测试脚本来达到和开发版本同步的目标。

Autorunner 具有开放的对象库，能够便于通过测试脚本自动生成的模式来“自动建立”测试脚本，达到“随需而变”的能力。

- 自动化测试工具

银行系统越来越复杂，需要对基于各种技术、平台的应用系统进行测试，如：网上银行基于“复杂的 web 系统”、信贷管理需要对工作流应用进行测试，核心业务需要对终端进行测试，未来的金融客户端还需要支持 java 界面等。

因此，就需要自动测试工具能够支持非常多的技术和平台：1）web；2）rich web；3）vt100、vt220；4）siverlight；5）wpf（.net）；6）win32 桌面应用；7）flex 等。

- 先进的、合理的自动测试框架

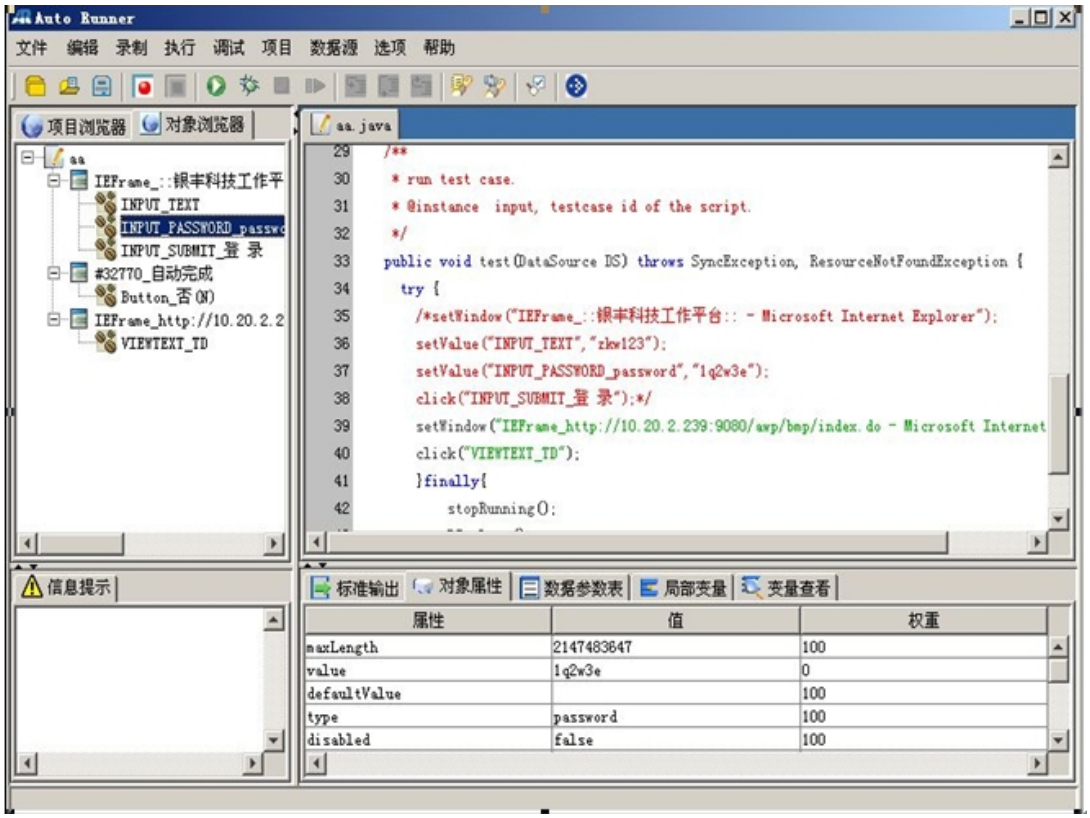
自动测试框架需要解决：

1） 脚本重用；
2） 测试案例的数据管理；
3） 数据池管理；
4） 流程测试；
5） 并发测试调度；
6） 测试日志记录等。

在本方案中，自动测试框架是由 TestCenter 来实现的，参见具体章节。

4.3. AutoRunner 产品介绍

泽众软件科技有限公司的自动化测试产品 Auto Runner 可以实现 Windows 环境下 B/S 和 C/S 架构系统的自动化测试，见下图。



Auto Runner

1) 支持的客户端：

✧ .net（VC、VB）

✧ IE

✧ Java GUI

2) 测试脚本语言：标准 java；

3) 与 Terminal Auto Runner 相同，支持录制和回放；

4) 允许用户以 Debug 方式执行测试脚本，即通过断点实现单步、跳入、跳出等调试技术；

5) 支持对象库管理。可以将测试中需要用到的测试对象保存在对象库中，使用对象库维护测试对象的属性。对象库中保存的测试对象可以被任何测试脚本反复引用；

6) 支持关键字驱动。除了直接编辑测试脚本外，用户还可以通过图形化的方式设计测试脚本的步骤。在关键字视图中，测试脚本被显示为树结构，每一个节点代表一个测试步骤。测试步骤可以是逻辑表达式，也可以是测试对象的 Action 调用。用户可以使用关键字视图创建、修改和删除执行步骤；

7) 与 Terminal Auto Runner 相同，支持脚本参数化。脚本参数也可保存在一个.csv 文件中；

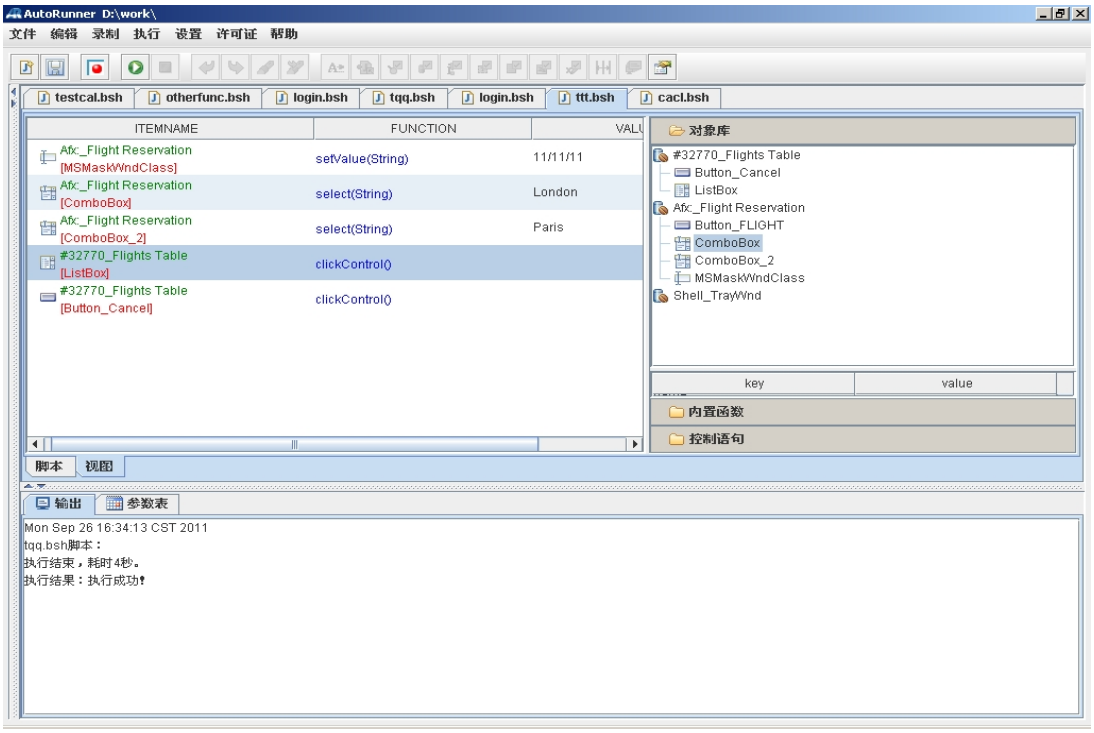
8) 虚拟设备，连接在 windows 系统串口上的设备：

➢ 密码键盘

➢ 磁卡读写器

4.4. 产品特点

- 关键字驱动



如上图，AutoRunner 支持通过“拖拽”而不是编写测试脚本来实现自动化测试脚本的编写，能够大幅度的降低测试脚本编写和维护的成本。

此外，支持“专家视图”，对于初学者很容易实现关键字视图与脚本视图之间的切换，实现两者之间的对应关系，降低学习成本。

- 支持更广泛的技术与平台

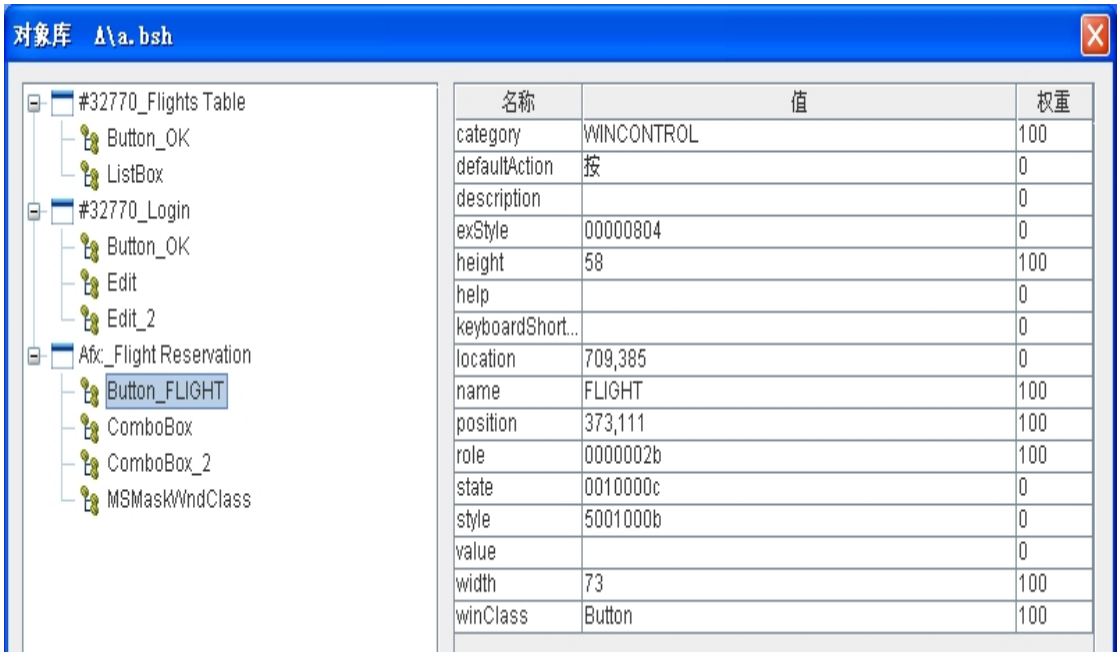
  支持的平台包括：

  a) web 应用；
  b) rich web 应用，如网上银行；
  c) win32 桌面应用，如桌面客户端程序，包括 java 应用程序、vc 编写的应用程序、vb 开发的桌面程序等；
  d) siverlight 应用；
  e) WPF 应用；
  f) Java GUI（awt、swing、swt 等）；
  g) Flex；
  h) QT 等。

- 描述性编程，解决复杂应用测试

  支持通过描述性编程来解决动态对象识别和组件执行问题。

- 开放的对象库



如上图，为对象库。Autorunner 支持直接查看对象库的各种属性，并且试用 XML 来存放对象库。

开放的对象库，使得测试脚本、测试对象库生成成为可能。

5. 终端自动测试工具方案

5.1. 产品介绍

泽众软件科技有限公司的自动化测试产品 Terminal Auto Runner 可以很好地兼容 VT100 以及其它多种模式下的字符终端测试，见下图。

1) 录制与回放

可以采用录制的方式获得自动化脚本，这是最简单的自动化脚本生成方式。但是通过这种方式获得的脚本数据是写死的，脚本和数据耦合度高，很多可复用的部分没有被分离出来，因此复用性差。更多情况下，都是通过手写代码来实现更好的脚本结构。

回放功能用于执行自动化测试脚本，回放的过程实际上是在模拟人工录入。

2) 自动获取系统返回结果数据

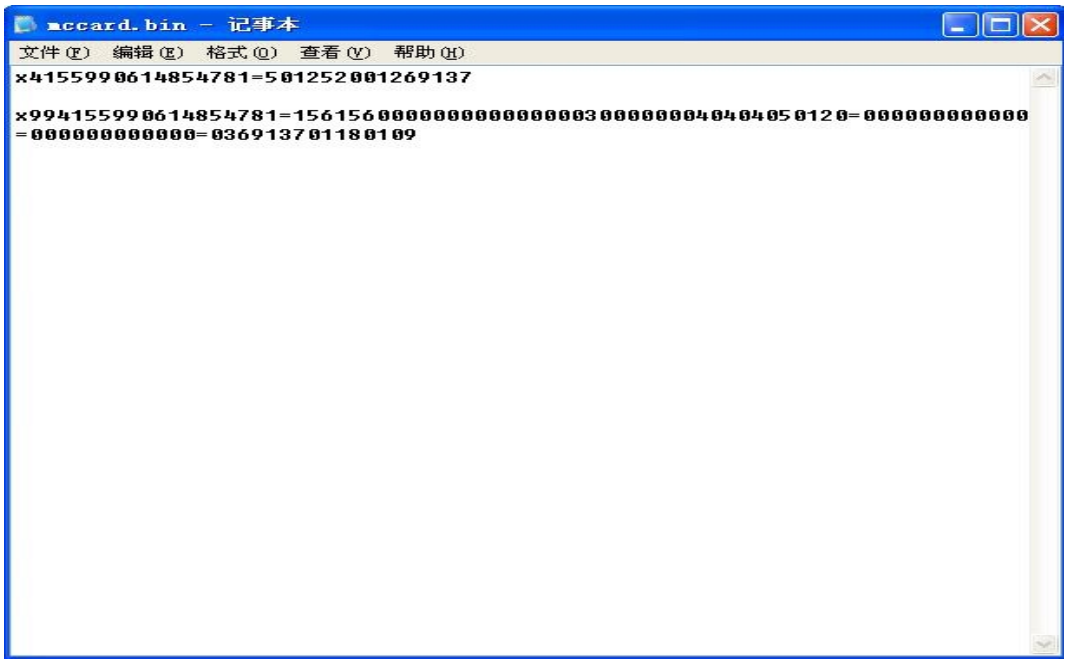
通过 Terminal Auto Runner 内置的函数 tarTest.getField(String key)可以获取系统返回的结果数据。通过这个功能可以设置检查点来检查系统输出是否符合系统需求的定义。

3) 虚拟设备支持

虚拟设备是 Terminal Auto Runner 提供的用来在测试中处理外设输入输出的功能。交易执行过程中，将会遇到一些与设备有关的操作，比如读卡，打印，密码键盘输入等。我们将会支持某些虚拟设备，主要是虚拟打印机、虚拟密码键盘和虚拟磁卡读写器。

在执行完交易后，有时需要打印。这里将会提供虚拟打印机，不是直接打印在纸上，而是虚拟地打印到 Terminal Auto Runner 安装目录下的 printer.txt 文件里（见图 57）。同时 Terminal Auto Runner 也支持虚拟磁卡的读入。当遇到需要刷磁卡的时候，不需要真的去刷卡，只要把卡的数据存放在 Terminal Auto Runner 安装目录下的 mccard .bin 文件下，当遇到需要刷卡的栏位，系统将自动从该文件中读取数据，以保证自动化测试高速无间断的执行，见下图。

虚拟打印机打印文件



磁卡读写器读取文件

1) 提供脚本参数化。通过定义数据源的方式，将脚本中的测试数据以键值对的格式抽取到一个.csv 文件中。用户可以在测试脚本中使用参数的 key 值引用保存在.csv 文件中的数据。

2) 截取屏幕。可以使用截取屏幕的函数 tarTest.captureScreen(String screenkey);在自动执行出错时保存系统屏幕。通过这种方法可以准确地告知测试人员出错的位置，帮助定位问题。

3) 测试效率：

✧ 高可靠性

✧ 高效执行

6. 能够提供的测试服务与方案

6.1. 性能测试方案与服务

6.1.1. 性能测试方法论

6.1.1.1. 性能测试分类

性能测试（Performance Testing）：性能**测试方法**是通过模拟生产运行的业务压力量和使用场景组合，测试系统的性能是否满足生成性能要求。即在特定的运行条件下验证系统的能力状况。

负载测试（Load Testing）：在给定的测试环境下，通过在被测系统上不断增加压力，直到性能指标超过预定指标或某种资源使用已经达到饱和状态，目的是了解系统性能容量和处理能力极限。负载测试的主要用途是发现系统性能的拐点，寻找系统能够支持的最大用户、业务等处理能力的约束。也可以理解为扩展性测试（Scalability Testing），即在固定测试环境，在其它测试角度（负载方面）不变的情况下，变化一个测试角度并持续增加压力，查看系统的性能曲线和处理极限，以及是 否有性能瓶颈存在（拐点）。主要意义是从多个不同的测试角度去探测分析系统的性能变化情况，配合性能调优。测试角度可以是并发用户数、业务量、数据量等不 同方面的负载。

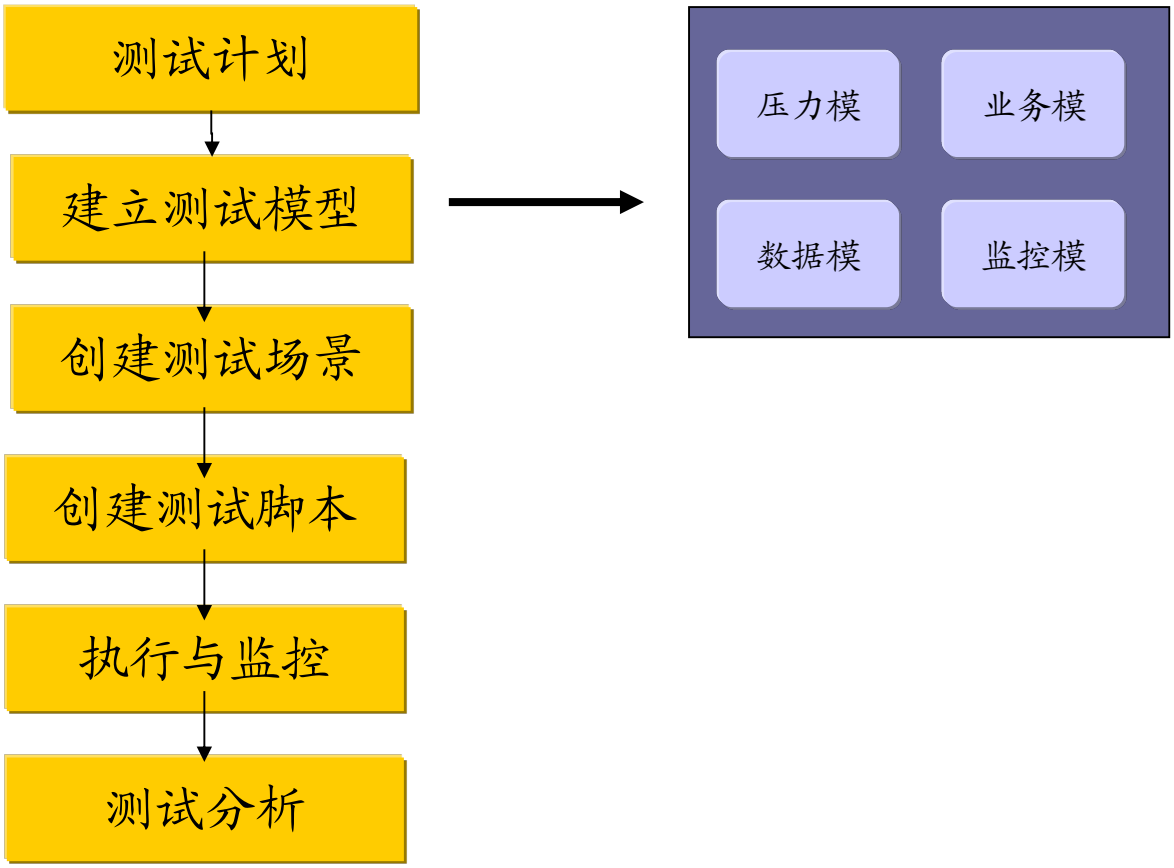
压力测试（Stress Testing）：测试系统在一定饱和状态下系统能够处理的会话能力，以及是否出现错误，一般用于稳定性测试。可以理解为资源的极限测试。测试关注在资源处于饱和或超负荷的情况下，系统能否正常运行，是一种在极端压力下的稳定性测试。其主要意义是通过测试调优保证系统即使在极端的压力情况下也不会出错甚至系统崩溃。

配置测试（Configuration Testing）：通过对被测系统的软硬件环境的调整，了解各种不同环境对性能影响的程度，从而找到系统各项资源的最有分配原则。主要用于性能调优，在经过测试获得了基准测试数据后，进行环境调整（包括硬件配置、网络、操作系统、应用服务器、**数据库**等），再将测试结果与基准数据进行对比，判断调整是否达到最佳状态。

并发测试（Concurrency Testing）：模拟并发访问，测试多用户并发访问同一个应用、模块、数据时是否产生隐藏的并发问题，如内存泄漏、线程锁、资源争用问题。测试目的并非为了获得性能指标，而是为了发现并发引起的问题。

可靠性测试（Reliability Testing）：通过给系统加载一定的业务压力的情况下，让应用持续运行一段时间，测试系统在这种条件下是否能够稳定运行。需要和压力测试区分开，两者的测试环境和测试目的不一样。压力测试强调在资源极限情况下系统是否出错，可靠性测试强调在一定的业务压力下长时间（如 24×7）运行系统，关注系统的运行情况（如资源使用率是否逐渐增加、响应是否是否越来越慢），是否有不稳定征兆。

6.1.1.2. 性能测试的一般过程



6.1.1.3. 性能测试模型

性能测试模型，分成：压力模型、业务模型、数据模型、监控模型、风险模型等。

6.1.2. 环境环境

分为：测试拓扑结构、硬件环境、软件环境、数据环境几个部分。
对这几个部分进行分析，发现可能的瓶颈。

6.1.3. 测试策略

制订测试策略，首先有对测试进行分析，识别在影响性能测试的风险项。然后根据风险项来制订测试策略。

6.1.4. 测试场景模型分析

对一般营业日和特殊营业日进行分析，获得测试数据。

6.2. 功能测试与系统测试方案与服务

根据业务需求，对系统进行功能测试与系统测试，并提供服务。

一般是在软件开发的瀑布模型中，对新增需求、需求变更，以及新建项目提供功能测试与系统测试服务。

6.3. 自动回归方案与服务

银行系统使用过程中，由于需求变更和程序修改，导致需要进行版本更新。这就需要经常性的回归测试。

银行需要建立标准化的自动测试用例库，用来对每个修改和变更的版本发布进行回归测试，以达到防止引入新的缺陷的目标。

此服务的目标是建立自动化的测试用例，可以随时进行自动化回归测试执行。