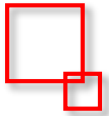
软 件 测 试 技 术

第6讲 集成测试 系统测试

软件工程 吕敬钦
创新大楼 西楼403



集成测试

- 集成测试
- 系统测试
- 测试过程管理
- 自动化测试

集成测试

- 实践经验

  尽管经过单元测试，集成后会有模块不能正常工作

- 主要由于模块互相调用时，会有新问题：

  1) 数据经过接口会丢失

  2) 模块A对模块B会有不应有的不良影响

  3) 几个子功能单元组合后，不能实现主功能

  4) 误差不断累积

  5) 全局数据结构出现错误

集成测试

- 集成测试

  将通过测试的模块，按设计要求组合成子系统，进行测试。

  以确保组合后，能按既定期望协作运行。

- 人员安排

  要求：熟悉单元的内部细节，且能够从高层次上观察整个系统。

  由有经验的测试人员和开发人员共同完成测试计划和执行。

集成测试

- 集成测试的内容

  检查穿越模块接口的数据是否会丢失

  判断各子功能组合起来能否达到预期要求的父功能

  检查模块 A 的功能是否会对其他模块的功能产生不利影响

  检查全局数据结构是否正确，以及在完成模块功能的过程中是否会被异常修改

  单个模块的误差累积起来，是否会放大到不可接受的程度

集成测试

集成测试用例

- 成对集成

  ND-NextDate, GD-GateDate

  VD-ValidDate,

  ID-IncrementDate,

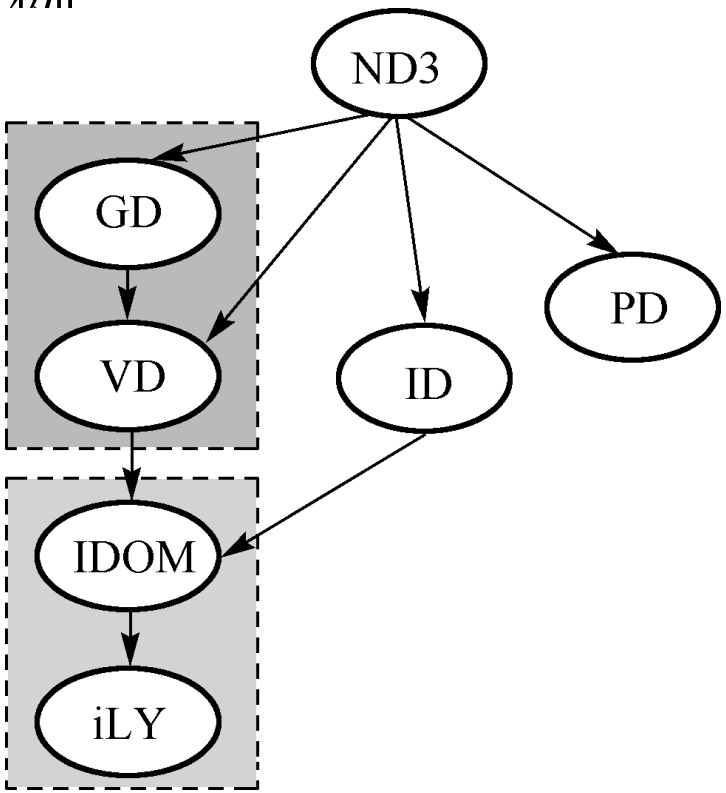
  IDOM-lastDayofMonth … 7个

  集成用例对应一对调用单元

  仅涉及一对调用接口。

  容易定位缺陷

  需要开发桩模块和驱动模块



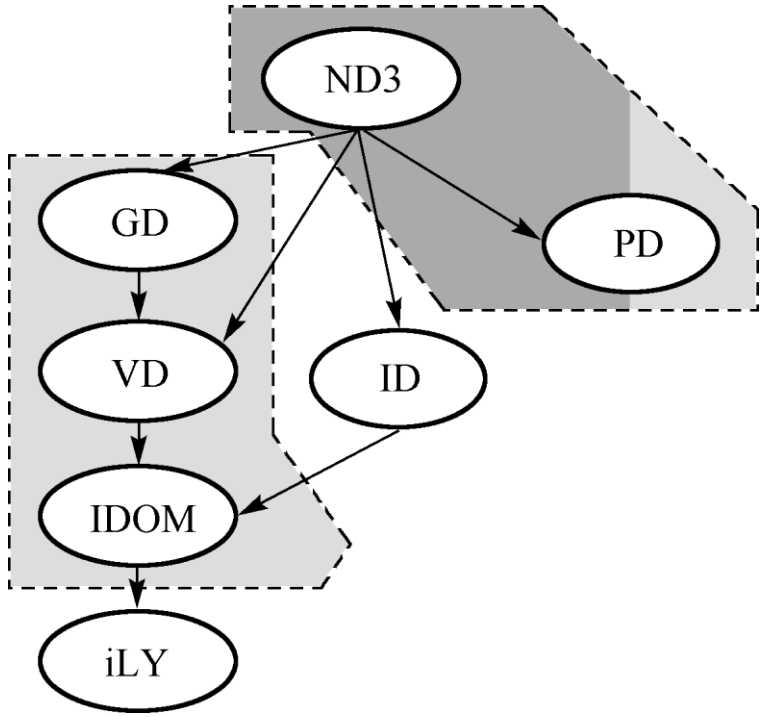
集成测试

- 邻居集成

  集成用例对应在某个节点的前后邻居

  扩大了测试范围

  桩模块和驱动模块的开发降低

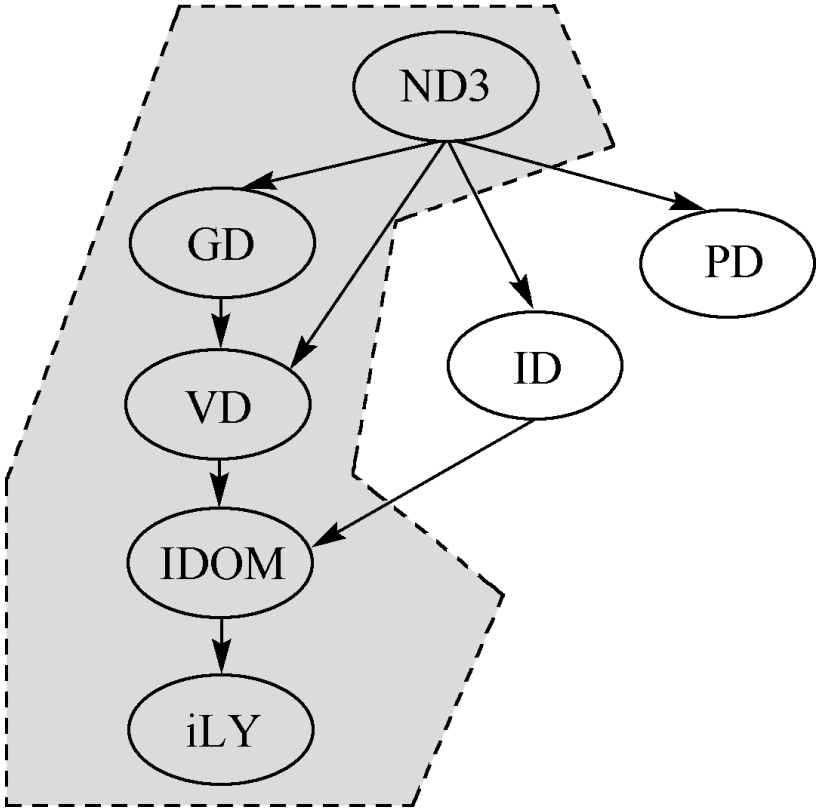


集成测试

- 基于独立路径的集成

  将函数调用图看作流程图，每个从根节点到叶节点的调用形成了路径，每条独立路径即可构成一个集成测试用例
- 测试范围更大

  缺陷定位难度大



集成测试的遍历模式

非渐增式：分别测试模块，再把所有模块按设计要求结合起来，进行测试。如大爆炸式。

- 大爆炸式

将所有模块一次性组装到被测系统中进行测试。

如NextDate 的7个小函数。

可能发现大量错误，定位

可能会漏测不少接口缺陷

此时，修改一个错误时，

仅在模块和接口数量少时，使用小范围的爆炸集成

集成测试的遍历模式

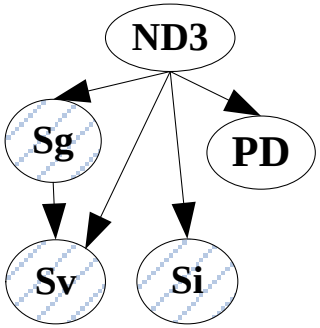
渐增式：把下一个要测试的模块，同已经测试好的模块集相结合起来进行测试。如自顶向下，自底向上，三明治式。

- 自顶向下

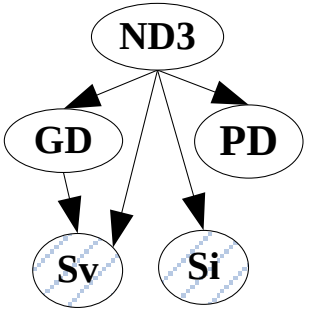
　　从主控模块开始，沿着控制层次，从上往下，逐渐将各模块组装起来。

集成测试

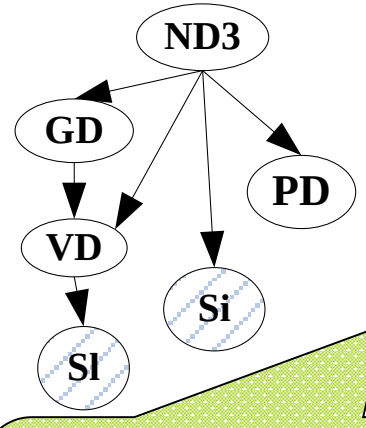
- 宽度优先的自顶向下集成——用例设计
  需开发桩模块

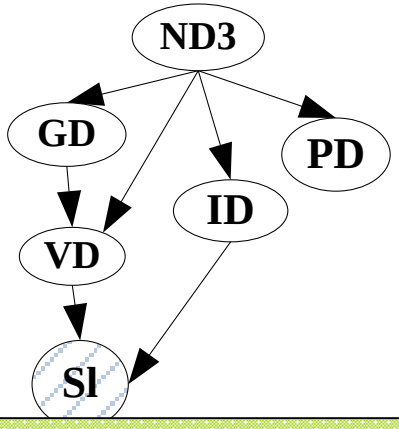


（a）测试NextDate3



（b）加入GetDate





为了保证加入模块没有引进新的错误，可能需要进行回归测试。

集成测试

- 自顶向下的优点

  1) 有助于早期实现并验证系统主要功能，给开发团队和用户带来成功的信心，

  也便于早期验证主要的控制和判断，避免主控程序的缺陷，确保开发进度

  2) 单个测试用例包含多个模块，可从整体上降低测试用例规模

  3) 采用递增方式展开测试，每个新的集成测试用例一般仅加入一个新的模块，便于缺陷定位

集成测试

- 自顶向下的不足

  1) 桩模块的开发和维护工作量较大

  2) 难以早期发现底层模块中复杂算法的缺陷，且随着测试的进行，系统越来越复杂，底层模块的测试很难保证充分性

  3) 不利于测试的并行，

- 自底向上
- 三明治集成

  将自顶向下和自底向上结合起来的集成策略。

通常根据开发进度，将完成的模块尽可能早地进行集成

集成测试

- 集成测试
- 系统测试
- 测试过程管理
- 自动化测试

系统测试

- 系统测试

在实际使用环境下，对计算机进行一系列严格测试，以保证用户能正常使用软件。

将经过良好集成测试的软件，与计算机硬件、外部设备、支持软件、数据及人员等元素结合在一起，

最后一个测试环节，具有重要的地位。

主要针对需求规格书里的系统功能和非功能需求

功能测试、性能测试、安全性测试、兼容性测试等

系统测试

- 功能测试

  主要针对系统的功能需
  是否满足用户的功能使

有时也可能采用白盒或灰盒方法，如查看代码、变量在数据库中的值

- 测试方式

  结合黑盒测试的基本思想，从以下三方面设计用例

  从系统输入（合法、非法）

  系统内部处理（包括数据计算和存储）

  系统输出（正常输出、错误提示输出和各种输出设备）

系统测试

- 性能测试

  测试软件的运行性能指标，判断软件在实际环境下，能否稳定、可靠地运行，是否满足预期的性能需求

  1) 响应时间

  如新增/修改操作在3秒内，查询操作在7秒内

  2) 运行时消耗的系统资源

  CPU、I/O、内存的使用情况，系统吞吐量等

- 具体有：常规性能测试、压力测试、负载测试、可靠性测试、大数据量测试

系统测试

性能测试和安全性测试，对测试员的技术水平要求高

- 安全性测试

  检查系统对非法入侵的防范能力

  测试人员扮演非法入侵者的角色，采用各种方法试图突破系统的安全防线。

- 兼容性测试

  检验软件与其他软、硬件相互是否能够正确交互和实现信息共享

  还包括，不同版本之间的兼容，数据方面的兼容

系统测试

- 用户界面测试

  规范化、正确性、直观性 … …

- 可安装性测试
- 容错性测试

  检查软件在异常条件下，是否具有防护性措施。

- 验收测试

  检验产品和产品规格说明书的一致性。（包括软件开发的技术合同）

  也叫现场测试，有用户参与

集成测试

- 集成测试
- 系统测试
- 测试过程管理
- 自动化测试

测试过程管理

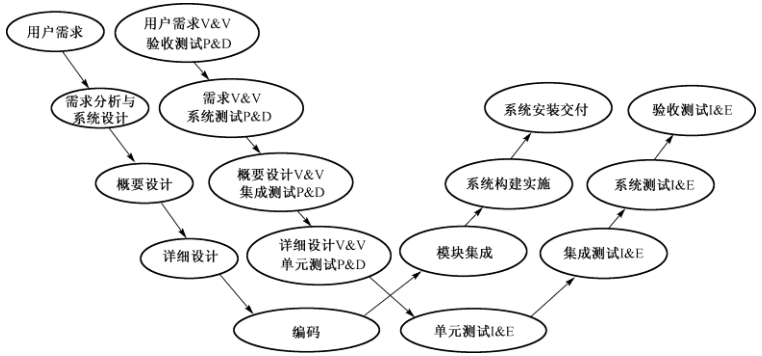
软件测试过程模型

- W模型 JPGs\10\p245.png

  两个V字，代表开发和测试

  左边：开发文档的静态测试，及对应的测试计划、设计和实施的过程等

  右边：实际代码和软件的动态测试



测试与开发行为对应，有助于早期发现缺陷，加快项目进度

测试过程管理

测试用例的管理

- 测试用例的组织和跟踪

  对系统各模块的功能需求，检查、理解和整理

  提取测试需求，撰写测试计划

  设计测试思路，设计测试

  专家评审测试用例

  修改更新测试用例

  执行测试用例

  评估测试用例

微软关于用户登录功能的用例多达5000个，需要对用例进行良好的管理

测试过程管理

- 测试用例报告

  用例的核心属性：ID、输入和预期输出

  主要内容还有：

  项目/软件、程序版本、编制人、编制时间、功能模块、功能特性、测试需求、优先级、测试环境、预置条件、初始化和清除环境、操作步骤、测试结果

  管理：商业工具，开源工具

软件缺陷的管理

- 缺陷的属性

  严重性：对被测系统造成的破坏程度

  优先级：缺陷必须被修复的紧急程度

  可重现性：在同样的条件下bug可反复出现
- 编写缺陷报告

  开源缺陷管理工具 BugFree
- 缺陷的处理

  测试员、项目经理、开发人员

集成测试

- 集成测试
- 系统测试
- 测试过程管理
- 自动化测试

课程《自动化测试技术》

- QTP 自动化功能测试

  Loadrunner 性能测试

  单元测试和工具
- PPT不要流到网络上

谢谢大家！