CS10102302
行为驱动开发

赵君峤 副教授

计算机科学与技术系 电子与信息工程学院

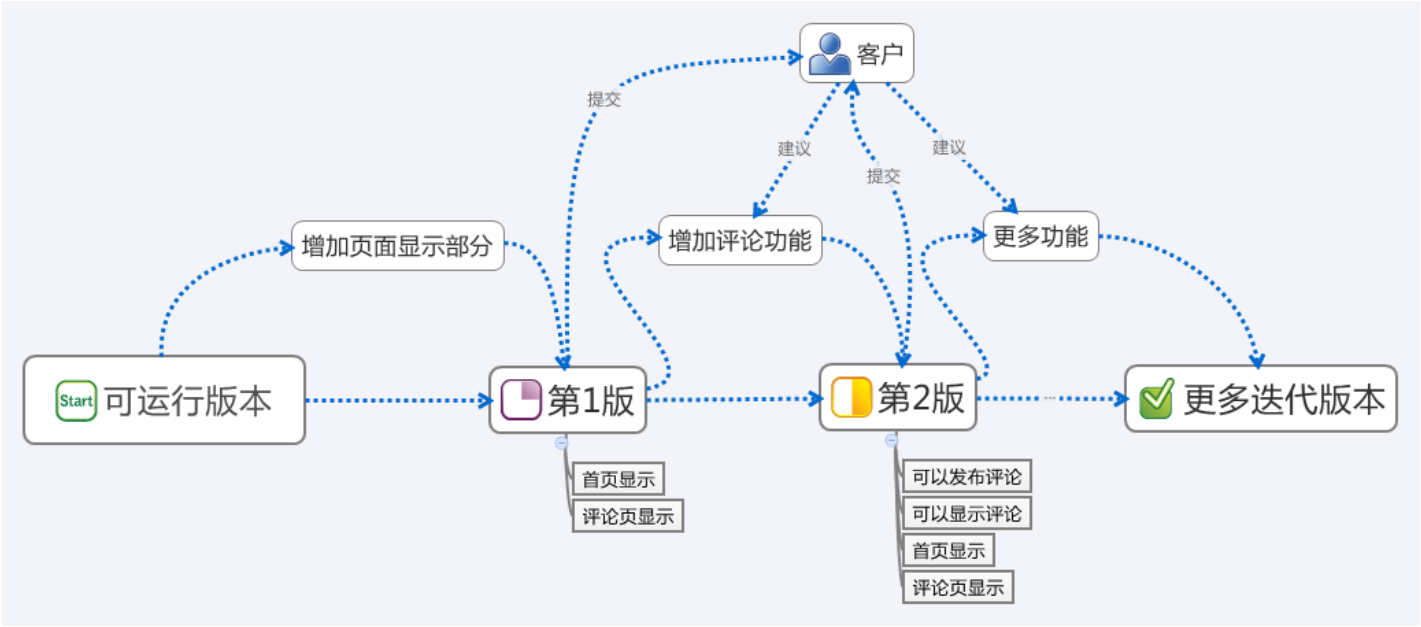
同济大学

软件开发之道

您是想完成一个更有价值的交付产品呢，
还是只想完成作业要求？！

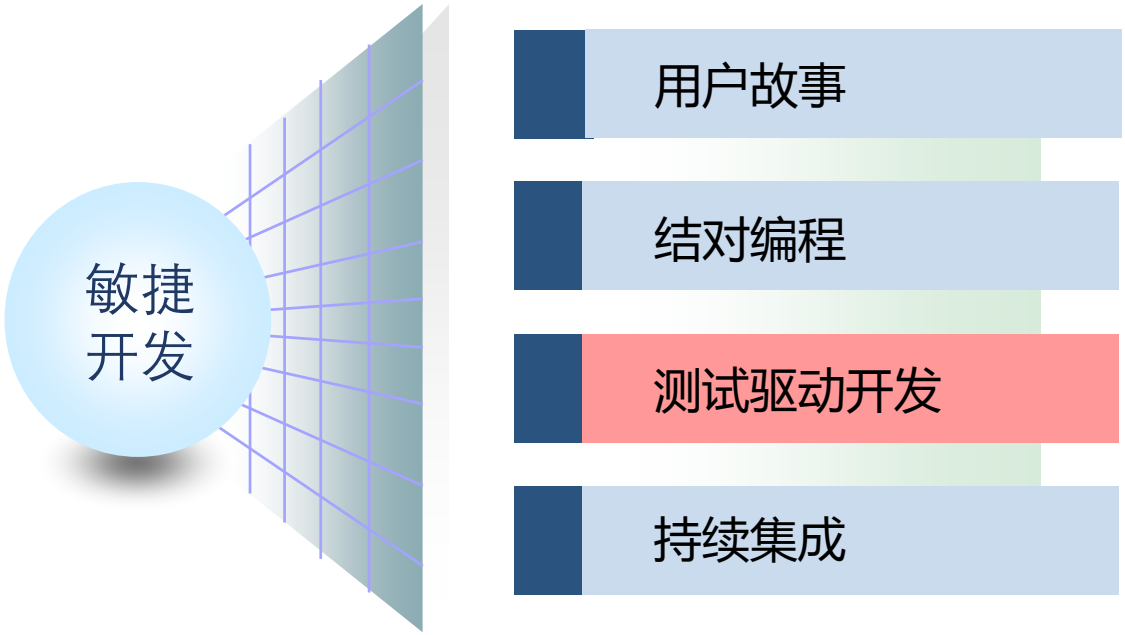
正确的软件	一个软件要能够满足用户的需求，为用户创造价值。这里的价值可以体现在两个方面，即为用户创造利润和减少成本。
软件运行正确	软件没有或者有很少缺陷，具有很强的扩展性、良好的性能以及较高的易用性等。

回顾：敏捷开发方法



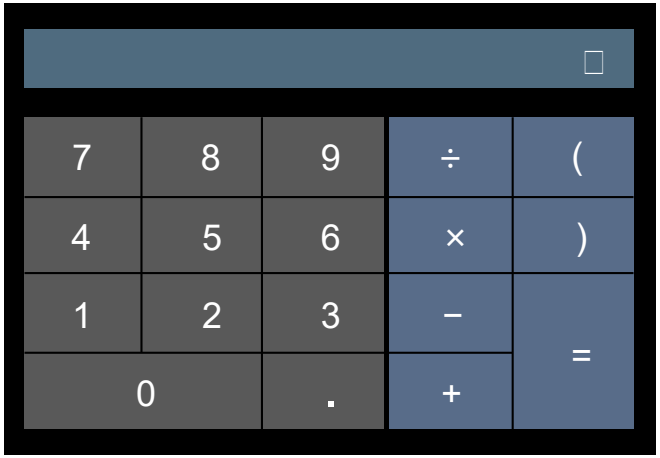
有效交流 | 迭代开发 | 持续交付 | 简洁为本 | 自组织团队

回顾：敏捷开发方法



思考：计算器程序

某学生打算开发一个计算器程序，可以实现加、减、乘、除等四种操作。应该如何编写该程序?



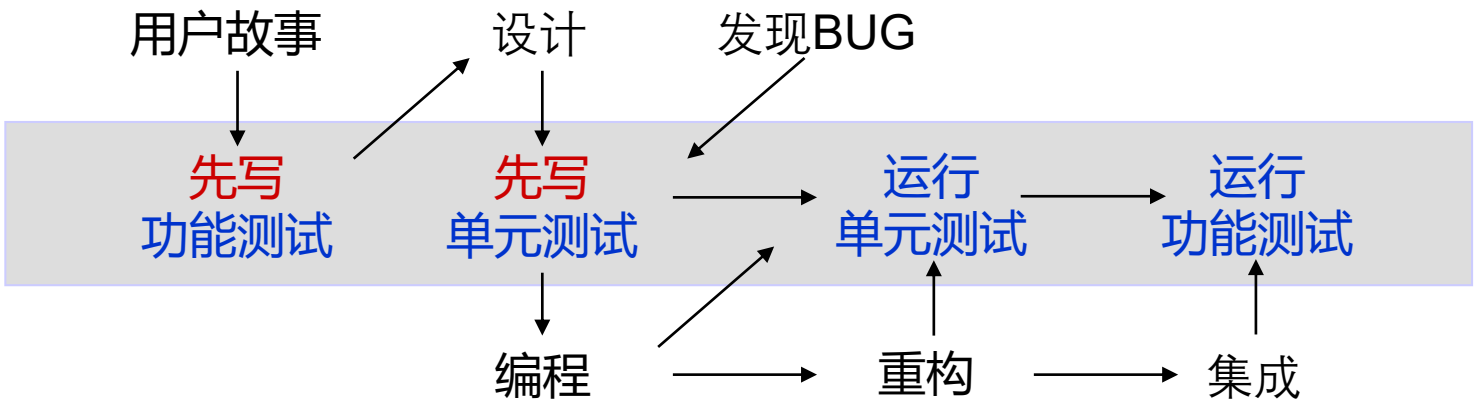
传统的开发方法

软件开发过程：先设计，再实现，最后测试

迫于时间压力，程序员往往
会简化或者不执行单元测试

测试驱动开发

测试驱动开发是以测试作为编程中心，要求在编写任何代码之前，首先编写定义代码功能的测试用例，编写的代码要通过测试，并不断进行重构优化。



测试驱动开发

测试驱动开发过程

- 对要解决的问题建模，编写测试代码
- 编写解决问题代码，并使其通过测试
- 思考意外情况，编写额外的测试
- 解决意外情况，使测试通过
- 重复上述流程，直到想不到意外

- TDD Seminar
- http://www.51testing.com/ddimg/uploadsoft/20090817/TddSeminar_Blog.pdf

测试驱动开发

- 先写测试可以帮助开发者集中精力于编写真正需要的代码
- 测试实际上展示了代码是如何工作的，在一定程度上测试代码变成了实现代码的使用文档，部分体现了“代码即文档”的思想
- 编写测试有助于开发者发现代码中可以抽象出来的API，从而将测试变成了设计过程的一部分
- 随着测试驱动开发的深入，我们会发现测试代码逐渐演变为对系统行为的定义描述

行为驱动开发

定义系统的行为才是测试驱动开发的真实价值！

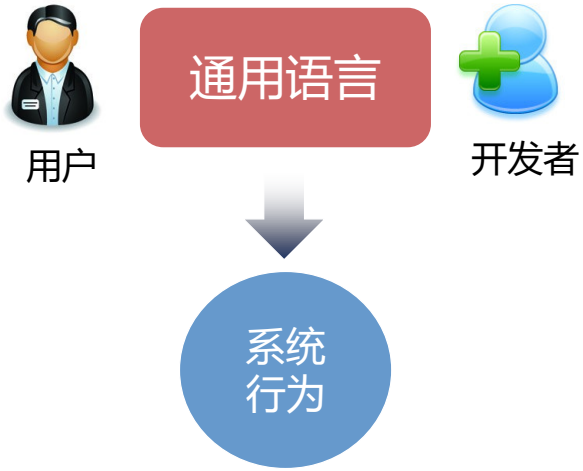
为什么不把测试与设计
更紧密地结合起来？

行为驱动开发（Behavior-Driven Development，BDD）是测试驱动开发的进化，但关注的核心是设计，它鼓励软件项目中开发者、测试人员和非技术人员或商业参与者之间的协作。

行为驱动开发

行为驱动开发是一种由外到内的开发方法

- 关注客户的商业价值
- 实现商业价值系统应具有的行为
- 对系统行为使用一种通用语言进行描述
- 用系统行为的定义来验证所实现的代码
- 由验收测试驱动开发



行为驱动开发

在行为驱动开发中，我们需要使用一种通用语言来定义系统行为。而通用语言实际上是一个最小化的词汇表，我们使用这些词汇来书写故事。

注意：词汇表的词汇必须具有准确无误的表达能力和一致的含义

例如：“系统”和“自动提款机”哪一个更符合要求？

故事是对系统某一方面的行为描述，它具有特定的格式，为接下来检验所实现的代码提供一个规范。

故事的书写格式

故事的书写格式

Story: 标题（描述故事的单行文字）
As a [角色]
I want [特征]
So that [利益]

用一系列场景
定义验证标准

Scenario 1: 标题（描述场景的单行文字）
Given [上下文]
And [更多的上下文] ……
When [事件]
Then [结果]
And [其他结果] ……

故事的书写格式

```
Story: Customer withdraws cash
As a customer,
I want to withdraw cash from an ATM,
so that I don't have to wait in line at the bank.
```

顾客的取款行为可以用哪些具体场景进行描述?

故事的书写格式

```
Scenario 1: Account is in credit
Given the account is in credit
And the card is valid
And the dispenser contains cash
When the customer requests cash
Then ensure the account is debited
And ensure cash is dispensed
And ensure the card is returned
```

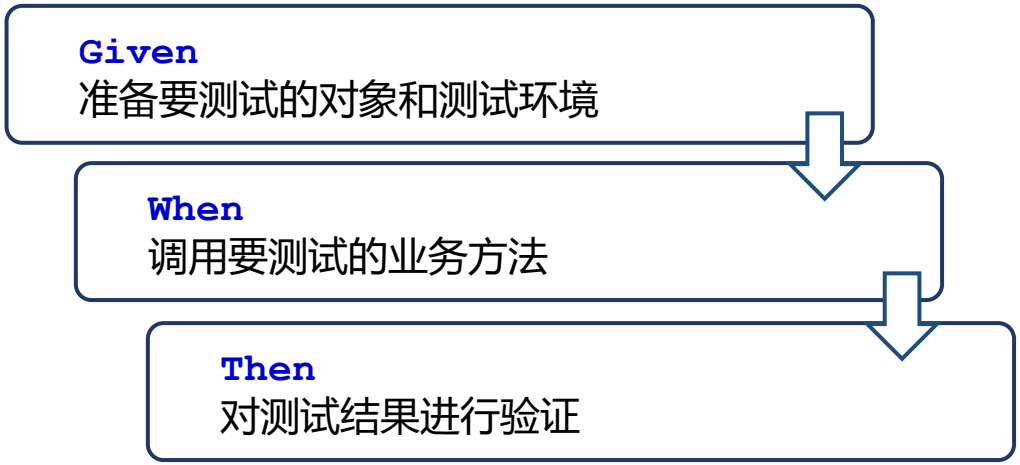
取款金额比实际存款少

```
Scenario 2: Account is overdrawn past the overdraft limit
Given the account is overdrawn
And the card is valid
When the customer requests cash
Then ensure a rejection message is displayed
And ensure cash is not dispensed
And ensure the card is returned
```

取款金额比实际存款多

故事即测试

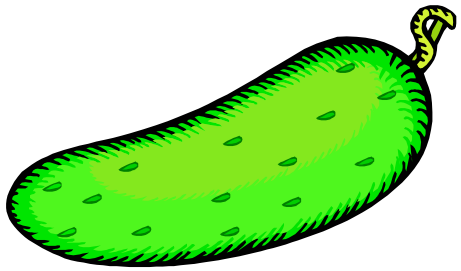
Given ... When ... Then 定义了一个完整的测试



测试者可以通过一些BDD测试框架将故事转成测试代码，开发者实现代码并保证测试代码通过。

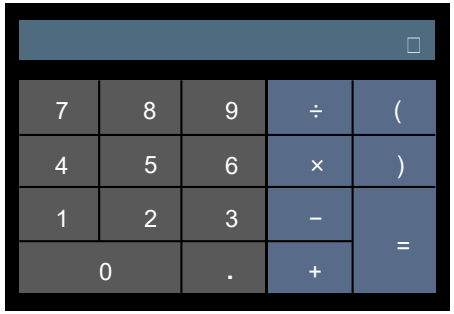
Cucumber

Cucumber是一个基于行为驱动开发的测试框架，它能够帮助团队使用业务上的语言对软件系统的行为进行测试。



Cucumber

Feature是软件系统所提供的某个功能，一般对应 User Story




```
Feature: Adding the numbers by calculator
As a customer,
I want to calculate the two numbers,
so that I can get the right results.
```

通常一个Feature文件中会包括一个或者多个Scenario

Cucumber

Scenario描述了用户使用软件的场景，即在某种前提条件下，当用户使用软件触发某个行为，那么应该会得到一个预期的结果。

```
Scenario: Add two numbers
Given the input "2+2"
When the calculator is run
Then the output should be "4"
```

通常有三种步骤：
- 给定的条件(Given)
- 触发的行为(When)
- 期待的结果(Then)

通常Scenario由一组Step（步骤）组成，Step描述了在完成一次与软件交互的过程中所需要的每个动作。

Cucumber

针对业务层面的描述语言，必然有对应的技术层面的实现，才能将这些业务描述语言转变成对系统进行测试的代码，这里称这种对业务语言的技术实现为Step Definition。

```
Given /^the input([^"]*)"$/ do |input_number|
  //get the input expression by the regexp match
  input = input_number
end

When /^the calculator is run$/ do
  //run the calculator and trigger it to do calculation by inpu
t
  result = calculator(input)
end

Then /^the output should be ([^"]*)"$/ do |expected_result|
  //get the expected result by the regexp match
  //verify whether the result is as we expected
   result == expected_result
end
```

Cucumber

Cucumber可以和以下测试框架集成，进行自动化的验收测试

- Webrat
- Selenium
- Capybara
- WebDriver
- Watir
- Celerity

NAME	TOTAL LINES	LINES OF CODE	TOTAL COVERAGE	CODE COVERAGE
app/controllers/activities_controller.rb	99	80	51.52%	46.25%
app/controllers/admin/administrators_controller.rb	7	4	28.57%	0.00%
app/controllers/admin/application_controller.rb	5	4	80.00%	75.00%
app/controllers/application_controller.rb	36	26	52.78%	50.00%
app/controllers/assets_controller.rb	9	8	44.44%	37.50%
app/controllers/attachments.rb	48	40	22.92%	22.50%
app/controllers/base_controller.rb	29	23	89.66%	86.96%
app/controllers/code_repositories_controller.rb	69	57	73.91%	70.18%
app/controllers/intros_controller.rb	3	3	66.67%	66.67%
app/controllers/sessions_controller.rb	43	31	48.84%	45.16%
app/controllers/site_controller.rb	13	11	53.85%	54.55%
app/controllers/topics_controller.rb	96	78	68.75%	64.10%
app/controllers/users_controller.rb	132	112	63.64%	62.50%
app/helpers/activities_helper.rb	32	26	75.00%	73.08%
app/helpers/admin/administrators_helper.rb	2	2	50.00%	50.00%
app/helpers/admin/application_helper.rb	2	2	50.00%	50.00%
app/helpers/application_helper.rb	9	6	66.67%	66.67%
app/helpers/audiences_helper.rb	2	2	50.00%	50.00%

Reference

- 清华大学国家级精品课程 《软件工程》主讲人 刘强 副教授 刘璘 副教授
- https://www.icourses.cn/sCourse/course_3016.html
- https://www.xuetangx.com/course/THU08091000367/5883555?channel=learn_title

谢谢大家！

THANKS