

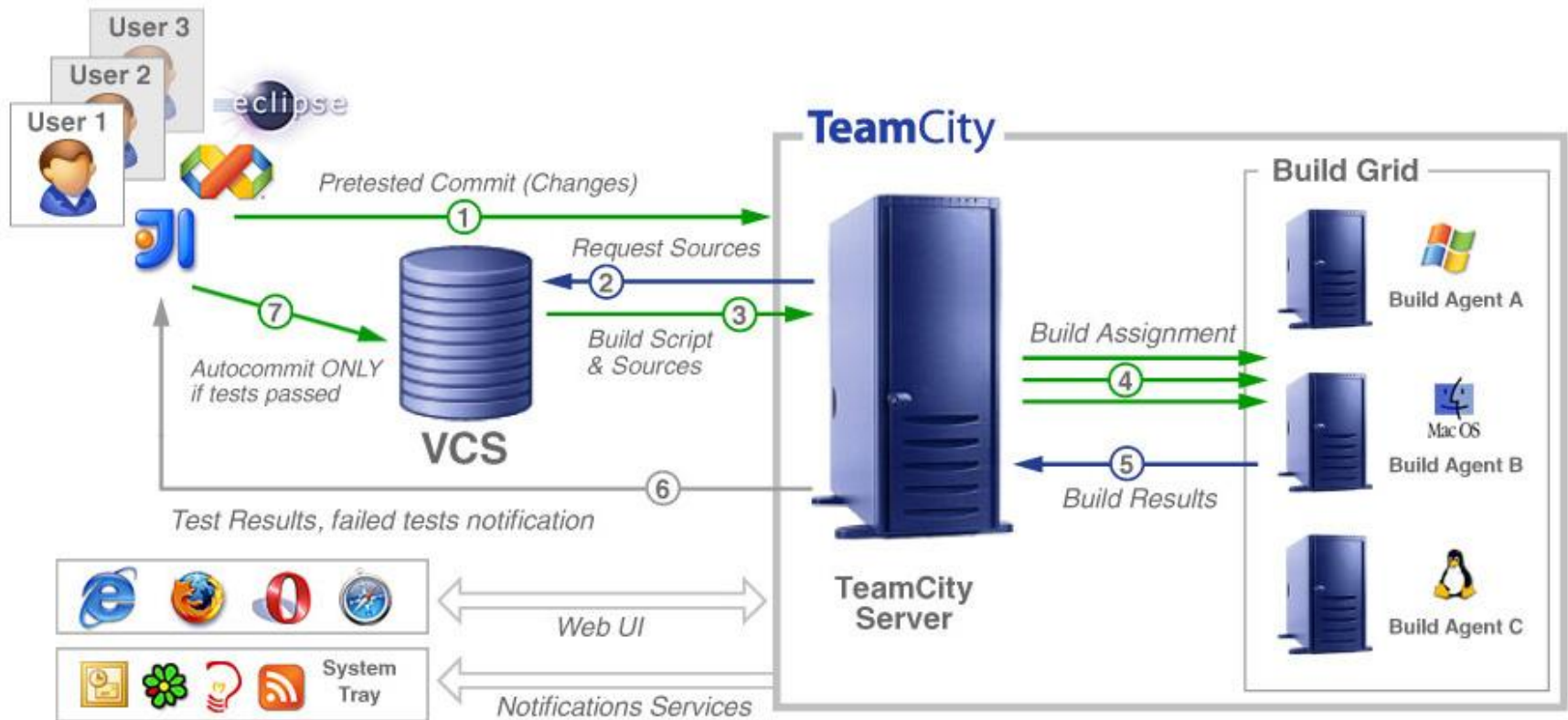
Unit-тестирование и метрики покрытия кода тестами

Сергей Андреев, JetBrains
29 февраля 2012

Это очень важный слайд



Continuous Integration



Что такое unit-тесты?

Тестирование - процесс запуска/выполнения программы с целью найти ошибки. (Гленфорд Майерс)

Unit-тестирование - процесс в программировании, позволяющий проверить на корректность отдельные модули исходного кода программы.

Наглядное пособие

Объект исследования – программа
«Треугольник», (автор – «студент
гуманитарного колледжа»)

Инструменты:

- IntelliJ Idea 11 (<http://www.jetbrains.com/idea/>)
- IDEA Code Coverage tool
- TestNG Framework
(<http://testng.org/doc/documentation-main.html>)

Statements

Покрытие операторов - все операторы программы выполнены.

3 4 5

2 4 4

3 3 3

1 -1 1

Даёт 100% покрытие по операторам

Code Coverage

YouTrack > Test Packaged Selenium - Google Chrome > #118 (10 Nov 10 14:51)

Run

Build Actions

Edit Configuration Settings

Overview

Changes (8)

Tests

Build Log

Build Parameters

Dependencies

Issues

Artifacts

Code Coverage

<< #117

All history

#119 >>

EMMA Coverage Report (generated Wed Nov 10 15:50:03 MSK 2010)

[all classes]

OVERALL COVERAGE SUMMARY

name	class, %	method, %	block, %	line, %
all classes	51% (2418/4697)	30% (5838/19505)	31% (208353/674609)	31% (47481,4/152319)

OVERALL STATS SUMMARY

total packages: 104
total executable files: 1325
total classes: 4697
total methods: 19505
total executable lines: 152319

COVERAGE BREAKDOWN BY PACKAGE

name	class, %	method, %	block, %	line, %
jetbrains.charisma.smartui.colorPicker.pallete	0% (0/1)	0% (0/7)	0% (0/245)	0% (0/25)
jetbrains.charisma.plugins	97% (30/31)	82% (116/142)	68% (672/994)	73% (208/286)
jetbrains.charisma.customfields.ui	93% (67/72)	67% (170/254)	76% (7404/9763)	76% (1822/2410)
jetbrains.charisma.smartadmin.admin2	69% (9/13)	44% (20/45)	76% (2325/3065)	80% (742,5/928)
jetbrains.charisma.smartui.inplaceIssueEdit	100% (25/25)	67% (61/91)	77% (4142/5362)	75% (954/1269)
jetbrains.charisma.customfields.plugin	100% (2/2)	100% (8/8)	100% (27/27)	100% (13/13)

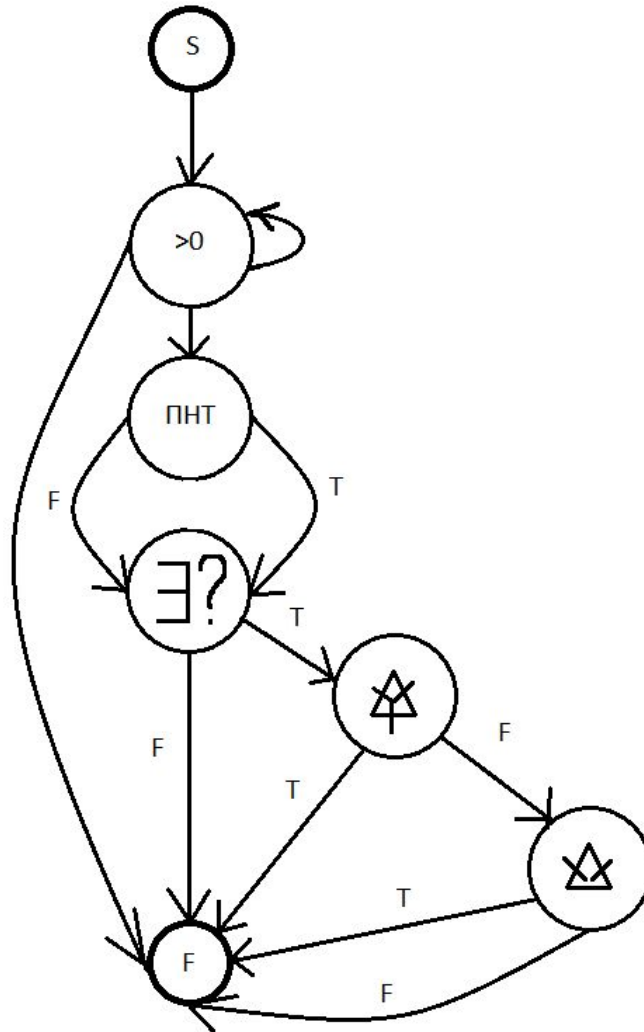
Control-flow graph

Граф потока управления - в теории компиляции — множество всех возможных путей исполнения программы, представленное в виде графа.

В графе потока управления каждый узел (вершина) графа соответствует базовому блоку — прямолинейному участку кода, не содержащего в себе ни операций передачи управления, ни точек, на которое управление передается из других частей программы. Имеется лишь 2 исключения: точка, на которую выполняется переход, является первой инструкцией в базовом блоке, и базовый блок завершается инструкцией перехода. Направленные дуги используются в графе для представления инструкций перехода.

Также, в большинстве реализаций добавлено два специализированных блока: входной блок, через который управление входит в граф и выходной блок, который завершает все пути в данном графе.

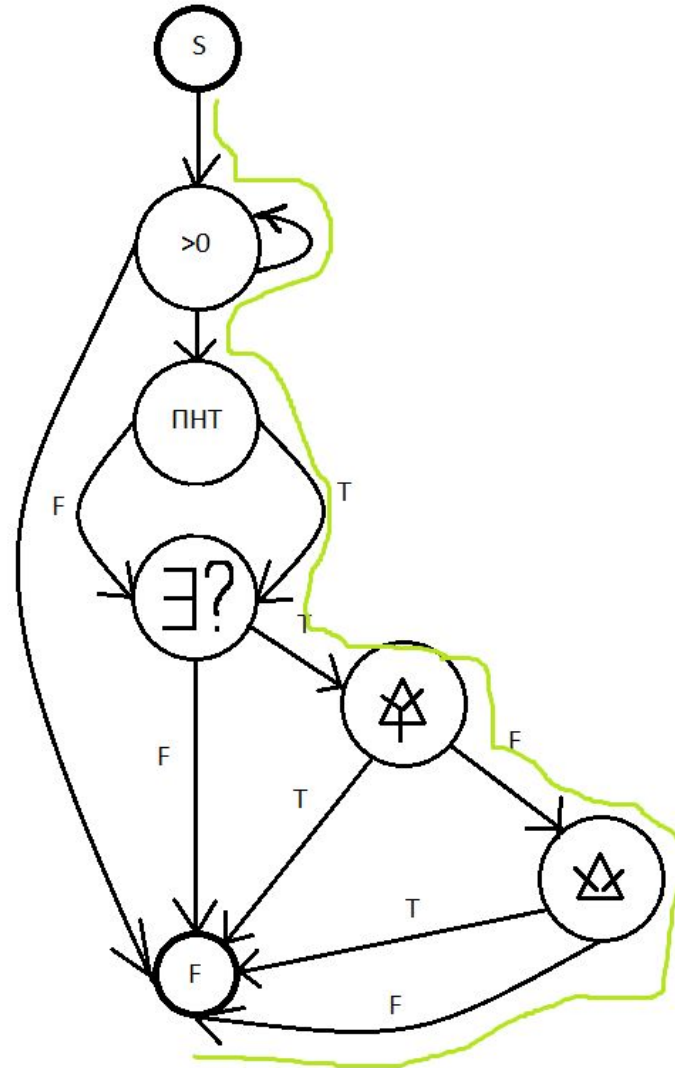
Mad Skillz



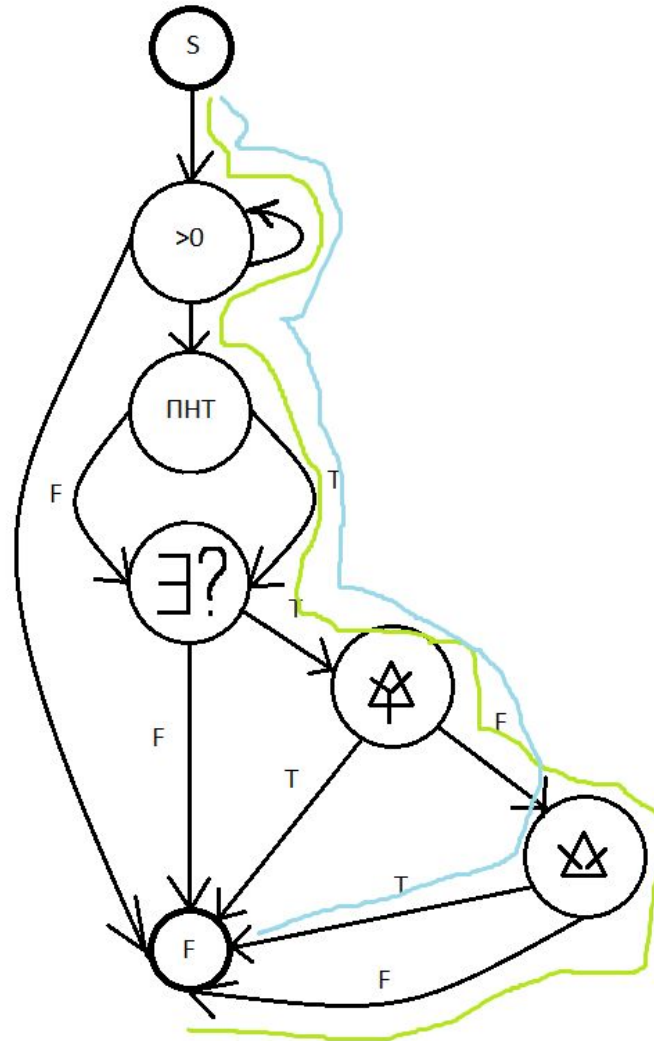
Branches

Покрытие ветвей - пройдены все ветви графа. Т.е. существует множество путей, проходящих по всем ветвям графа.

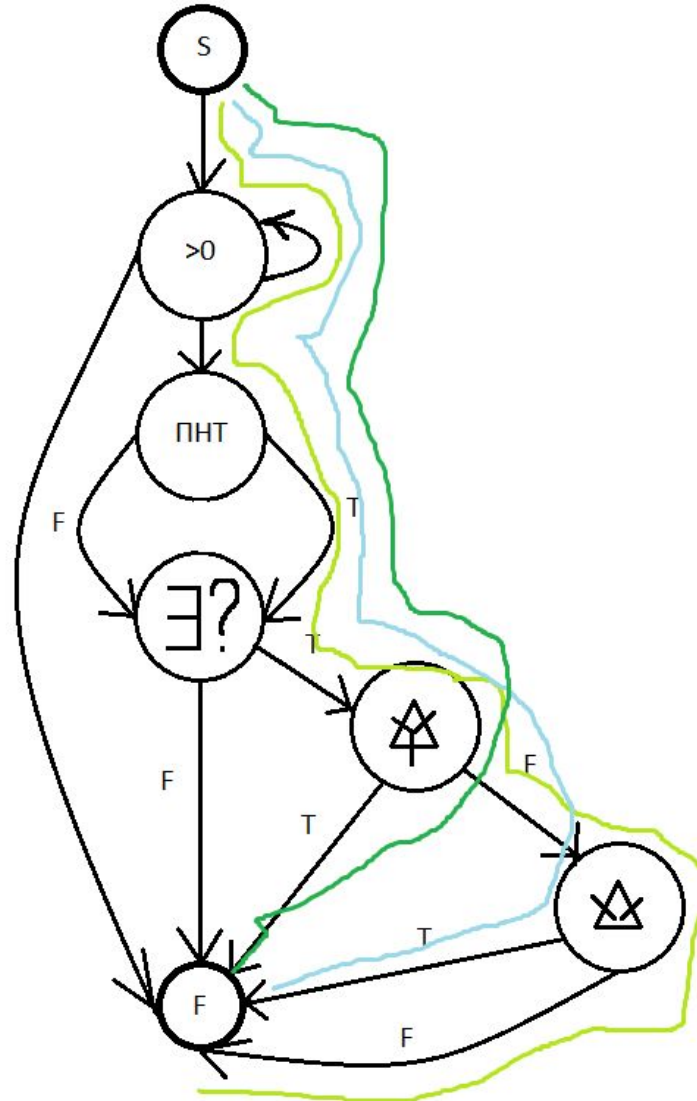
Тест 3 4 5



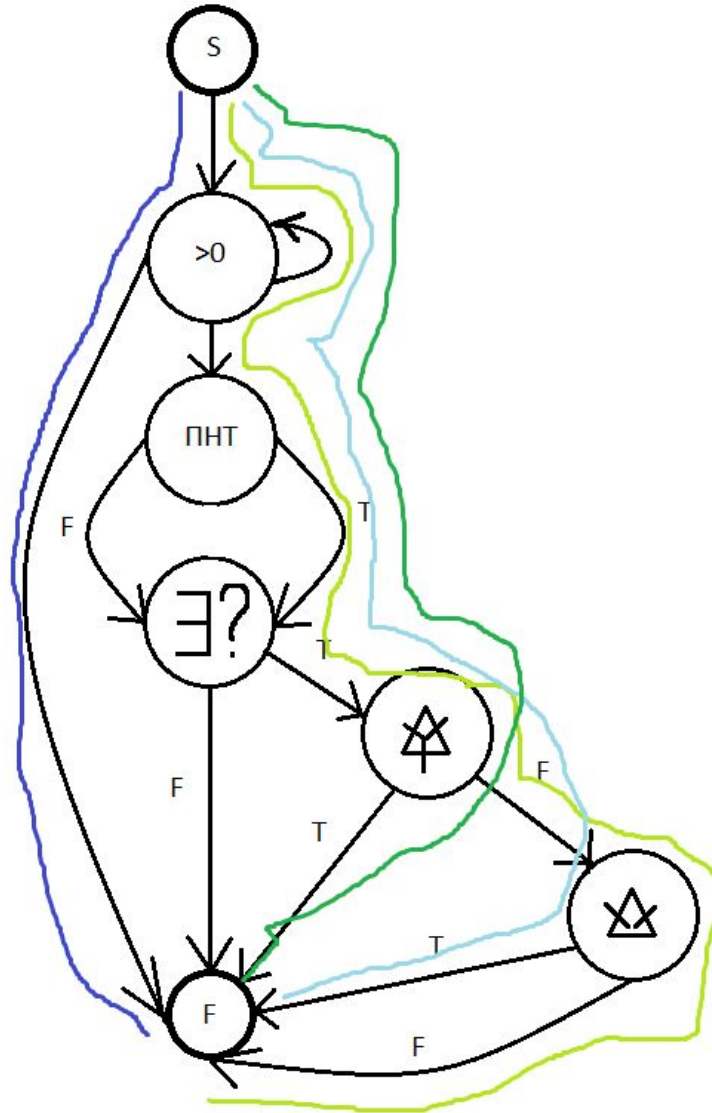
Добавим тест 2 4 4



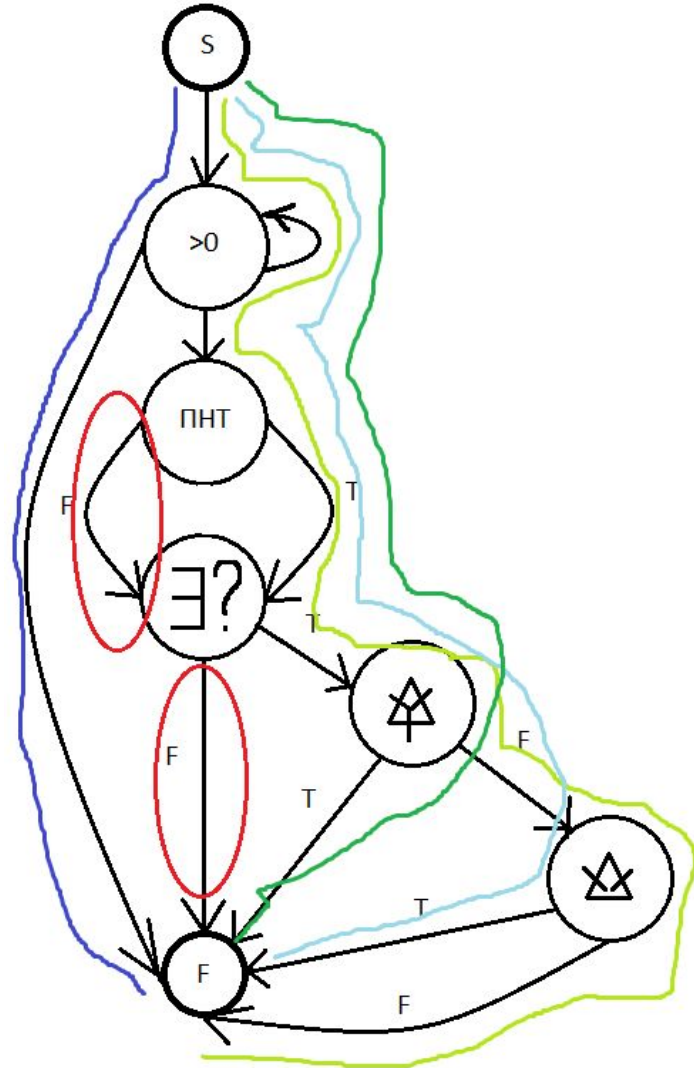
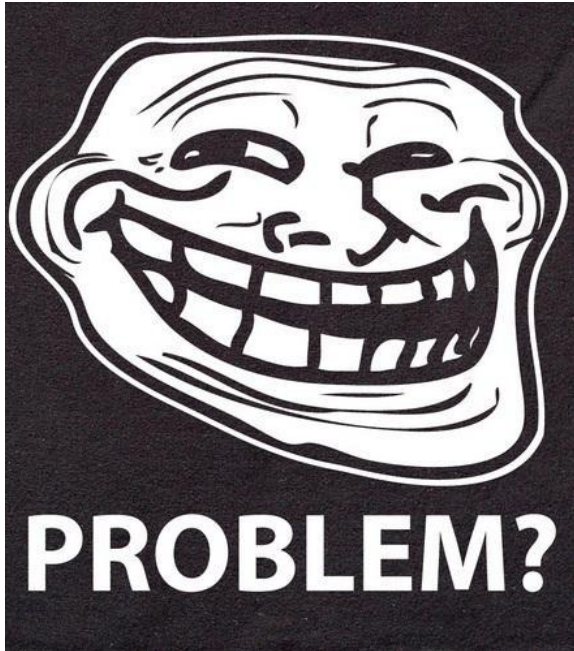
Добавим тест 3 3 3



Добавим тест 1 -1 1



Проблемы, офицер?



All Path testing

Покрытие всех путей в графе. В общем случае получаем почти неразрешимую задачу. Нужно выбирать репрезентативное множество. Убрать лишнюю информацию из множества:

- лишние вершины
- лишние ребра

Более частные случаи

Multiple Conditions – покрытие предикатов.

В отличие от Branches - не рассматривает выход узлов решений как 1 единицу, а рассматривает каждый предикат по отдельности.

Loop coverage – покрытие циклов.

Отдельно рассматриваются циклы. Они должны быть выполнены 0, 1, ... max, max-1, max+1 ?

Пожалуй хватит.
Спасибо за внимание!

sergey.andreev@jetbrains.com
smandreev@gmail.com

