

# Quality Control in Continuous Integration

---

Konstantin Zhukov

- Что такое Continuous Integration?
- Риски процесса разработки
- От «Continuous Integration» к «Build Pipeline»
  - *Quality Control*
  - *Практические моменты реализации*

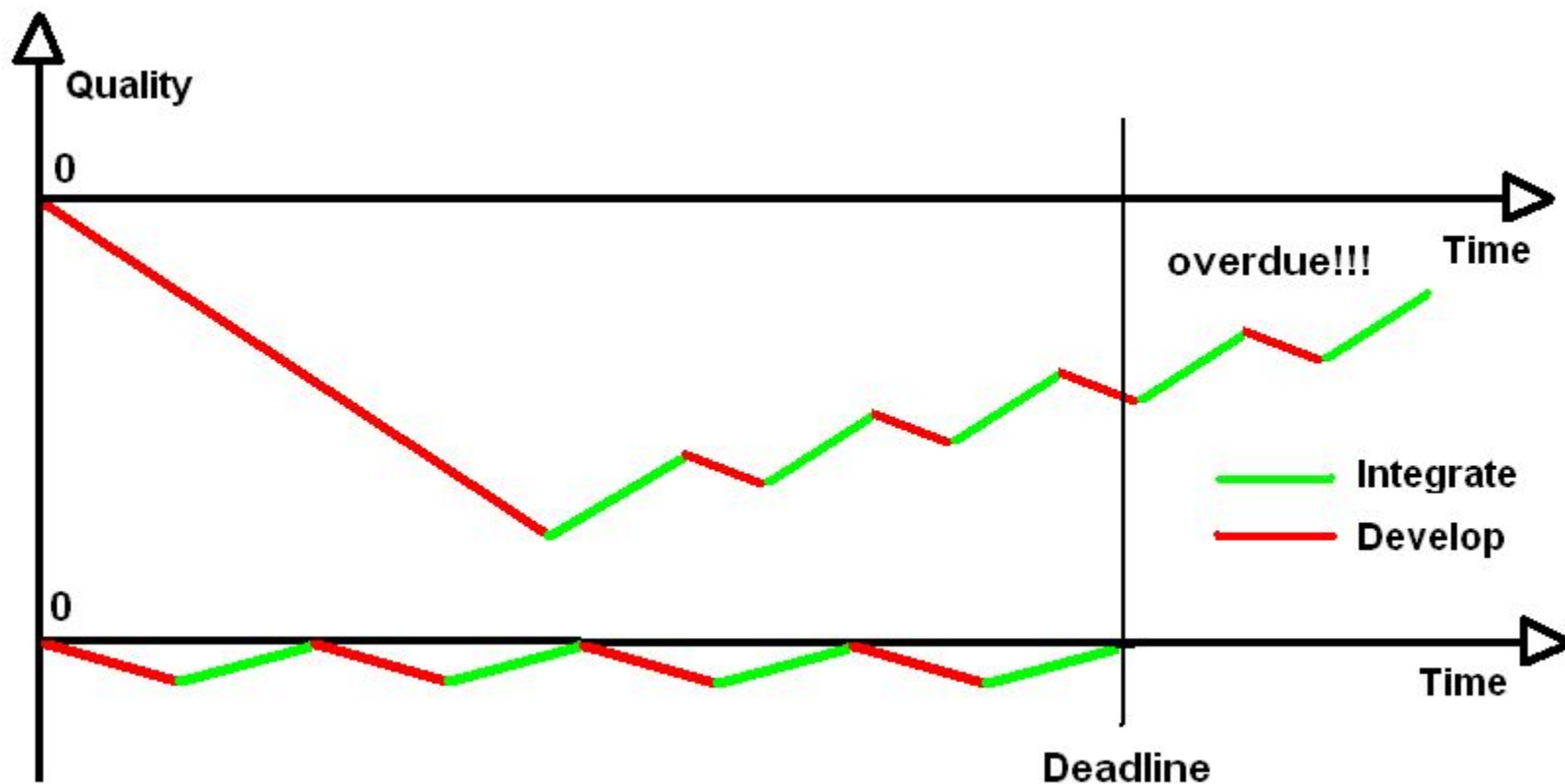
# Что такое Continuous Integration?

- «Process of applying quality control during development» (с)wikipedia
  - *Стратегия разработки,*
  - *связанная с регулярной интеграцией,*
  - *проводимой в автоматическом режиме*

**Производство программных  
продуктов –  
рискованное дело**



# Риск 1: Поздняя интеграция стоит дорого



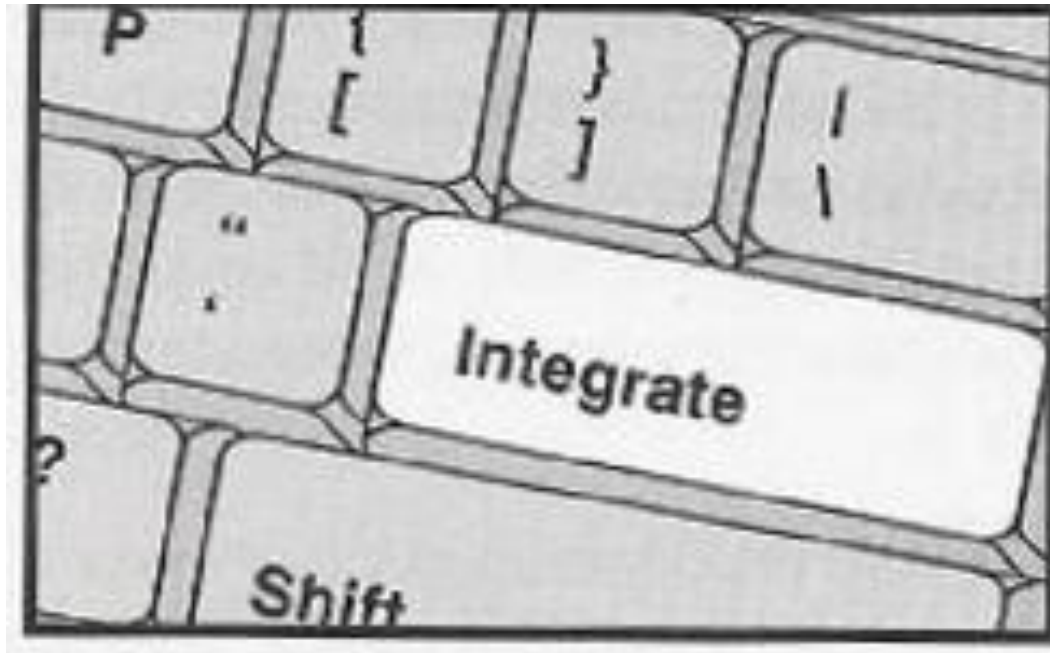
- *Атомарные изменения*
- *Интегрируемся чаще!*

## Риск 2: Отсутствие регулярных сборок

- Продукт работает только локально
  - *Локально всё работает!*
  - *Завтра придет босс, показывать нечего!*
- Процесс непрозрачен
  - *Какой сейчас статус проекта?*
  - *Что мы такого сделали в версии 1.0.1?*

# Интеграция необходима

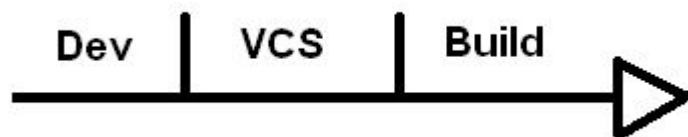
- Хотелось бы иметь что-то вроде



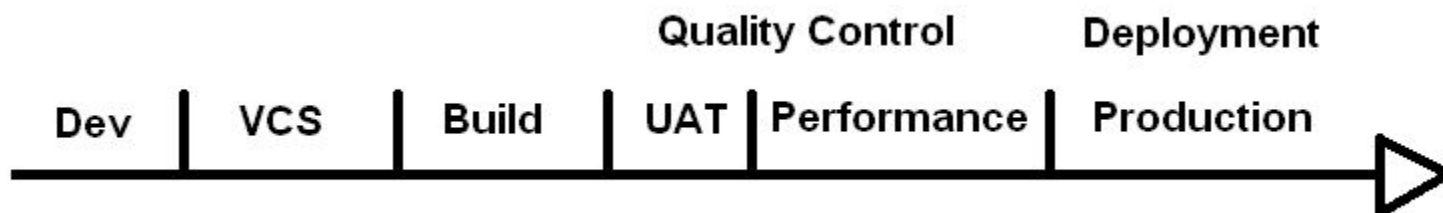
- Что дальше?

# От «Continuous Integration» к «Build Pipeline»

- «Automated manifestation of your process for getting software from version control into the hands of your users»
- Continuous Integration flow



- Build Pipeline flow





# Build Pipeline: Как это организовать?

- Специальные инструменты для поддержки процесса



ThoughtWorks®

anthillpro  
by urban{code}

- *... сегодня про это не говорим*

# Build Pipeline: Quality Control

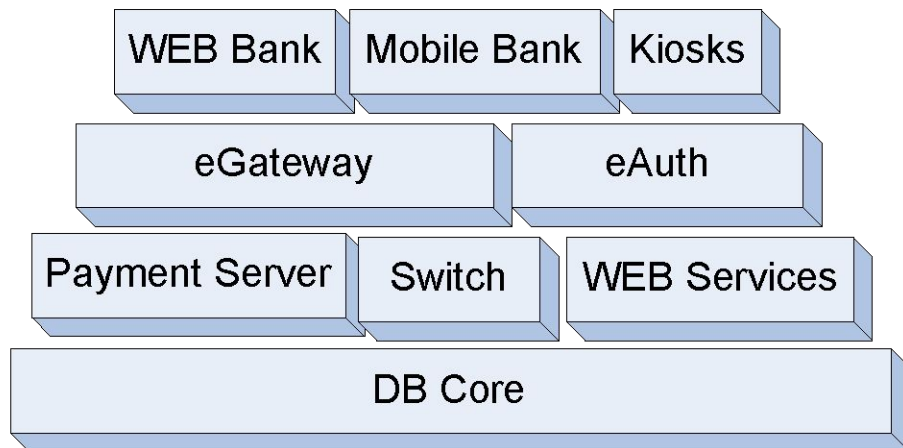
- Сконцентрируемся на QC
- Какие для этого предпосылки?
  - *К фазе QC доступны все необходимые артефакты (binaries)*
  - *Билд готов к тестированию!*
- Как его организовать?
  - *Есть проблемы*
  - *и есть решения*

# Проблема 1: Слишком много продуктов

- Много продуктов -> слишком большая энтропия
- Все продукты разные -> разные инструменты
- Нужны:
  - *Правила организации тестов*
  - *Единая система управления разнородными тестами*



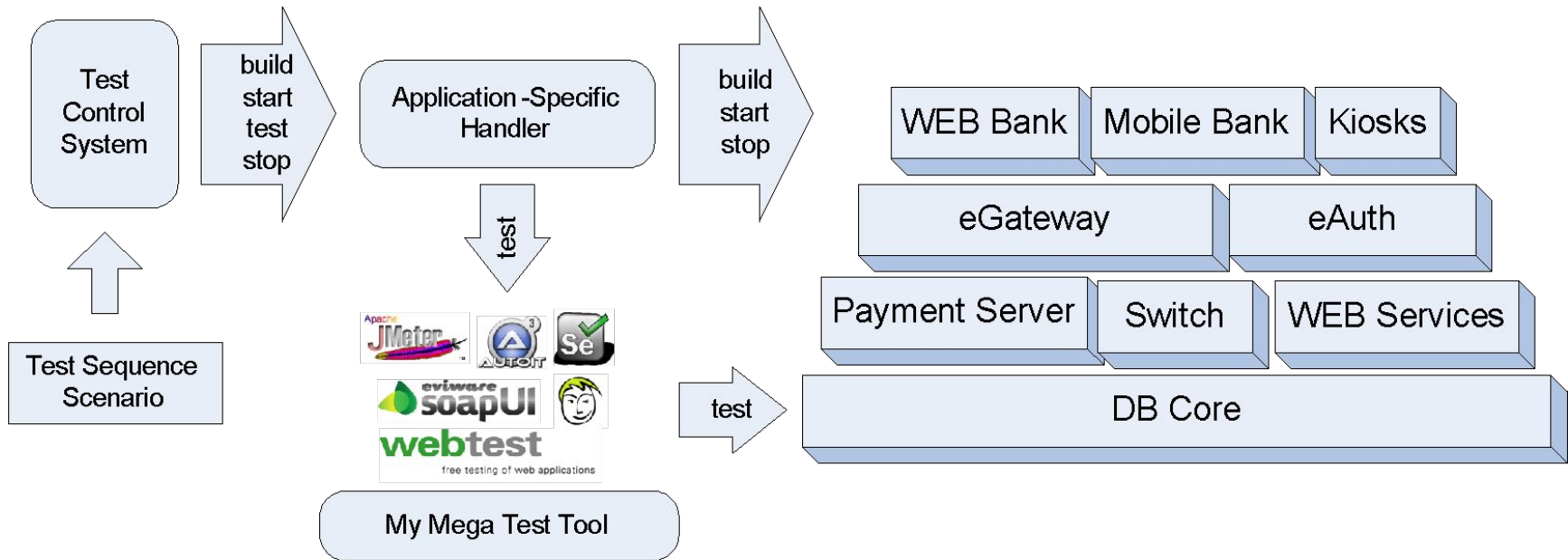
VS



# Решение 1: Слишком много продуктов

- Основные шаги – общие для всех
- Идём от алгоритма
- Что нужно?
  - *Собрать конфигурацию приложения (build)*
  - *Запустить конфигурацию (deploy + start)*
  - *Запустить тесты (test)*
  - *Собрать отчёт (collect logs)*
  - *Остановить конфигурацию (stop + undeploy)*
- Отделяем управление от реализации

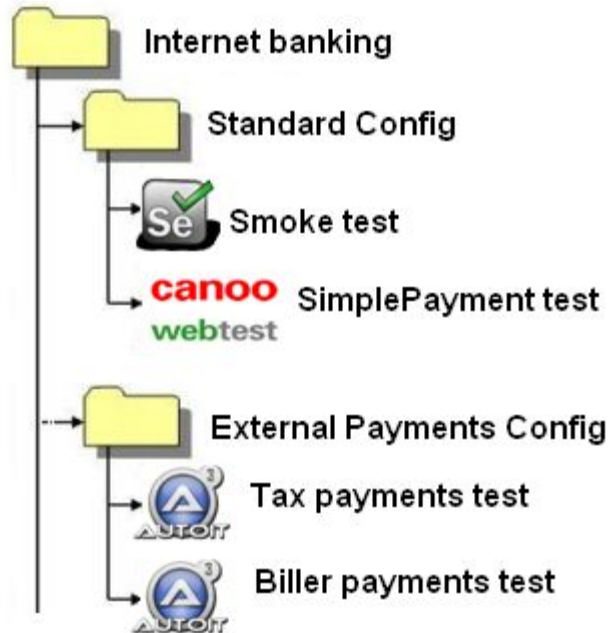
# Система управления тестами: Алгоритм



- *Вся специфика тестируемых приложений спрятана в специальном handler-е*

# Система управления тестами: Последовательность тестов

- Основа – файловое дерево
  - Алгоритм обхода – итерирование на одном уровне



- Простота!

# Система управления тестами: Мониторинг

- Единая WEB консоль CI сервера

HEAD/wb2 build at Tue Nov 29 21:35:07 MSK 2011

Overview

Init

Logs

st

Test

Deploy

Post Build

Changelog

## Last 20 builds

Time	Version	Caller	Result
29 ноя 2011 21:35:07 (26 minutes)	2.4.6-0001-SNAPSHOT	CINT	failure
28 ноя 2011 14:03:06 (25 minutes 38 seconds)	2.4.6-0001-SNAPSHOT		failure
25 ноя 2011 20:05:39 (23 minutes 31 seconds)	2.4.6-0001-SNAPSHOT	CINT	failure
25 ноя 2011 18:20:45 (44 minutes 23 seconds)	2.4.6-0001-SNAPSHOT	kzhukov	clear
25 ноя 2011 16:27:18	2.4.6-0001-SNAPSHOT		failure

## Build history

## Build phases

Name	Result	Duration
Init	clear	38 seconds
Prepare	clear	11 seconds
Dependencies		52 seconds
Build		1 minute 56 seconds
Dist	clear	46 seconds
Test	failure	22 minutes 4 seconds
Deploy	skipped	0 seconds
Post Build	skipped	0 seconds

## Pipeline phases

## Projects

Project	Checkout	Build	Test	Validation
wbcommon			unavailable	skipped
wbproto			unavailable	unavailable

## Dependent project status

## Test named actions

Test	Result
wbtests.Typical.WB2.IWBErollUser.w4c.03.33.30	clear ...
wbtests.Typical.WB2.Answerl	clear ...
wbtests.Typical.WB2.BadBro	clear ...
wbtests.Typical.WB2.CPAcc0	failure.
wbtests.Typical.WB2.CPAppiChangeClientAddress.w4c.03.33.30	failure.

## Tests status

# Build Pipeline: Бесплатное приложение

- Ручной redeploy дорог
  - *Можно ли его автоматизировать?*
- Можно
  - *Меняем местами шаги start <-> stop*
  - *Создаём технический сценарий без тестов*
- В итоге:
  - *Приложение всегда up-to-date и готово к тестированию*
  - *... и к показу боссу 😊*



## Проблема 2: Продукт слишком сложен

- Продукт слишком сложный, но нужны интеграционные тесты
- Разбираем его на части, сохраняя общую структуру,
  - *Собираем каждый компонент в режиме redeploy*
  - *Запускаем интеграционные тесты из общей точки входа*

- Что мы добились
  - *Разрешение всех рисков*
  - *Покрытие всех продуктов компании*
  - *+ Помощь в ручном тестировании*
- С небольшими недостатками
  - *Поддержка непростой инфраструктуры*
  - *Необходимость отдельных серверов*
  - *Зачем тестировать самому, если CI всё проверит сам? 😊*

Thanks

Thank you!



[kzhlukov@icctol.ru](mailto:kzhlukov@icctol.ru)  
[kost.zhukov@gmail.com](mailto:kost.zhukov@gmail.com)