



Технология непрерывной поставки ПО

Практики DevOps

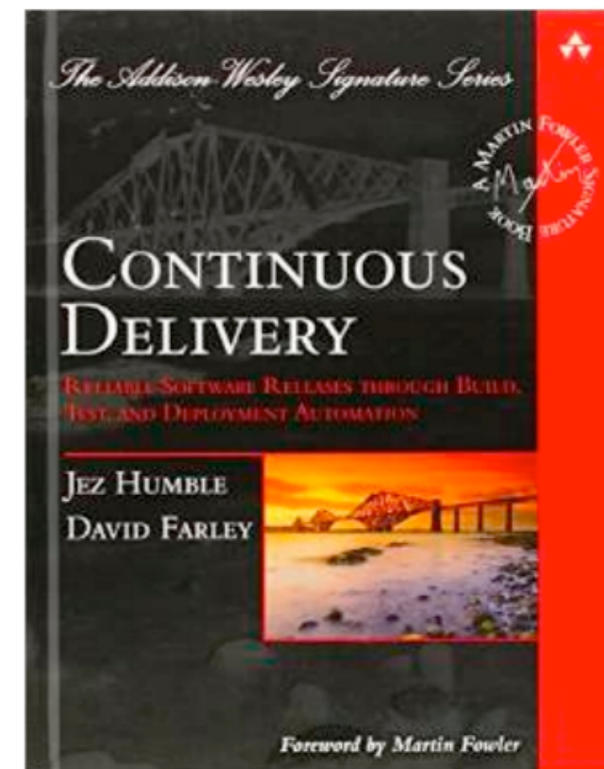
- Непрерывная интеграция (Continuous Integration)
- Непрерывная поставка (Continuous Delivery)
- Тестирование и Управление тестовыми данными
- Непрерывный мониторинг
- Управление версиями и релиз -менеджмент
- Управление конфигурацией и инфраструктура как код

Непрерывная интеграция

- проект забирается из СКВ
- проект собирается
- прогоняются тесты
- проходит выкатка на тестовый стенд (шоты)
- рассылаются оповещения

Непрерывная поставка

- Continuous Delivery (CD)
- Непрерывная поставка ПО — это процесс разработки и эксплуатации ПО, при котором каждое изменение может быть выкачено в боевую среду



Конвейер



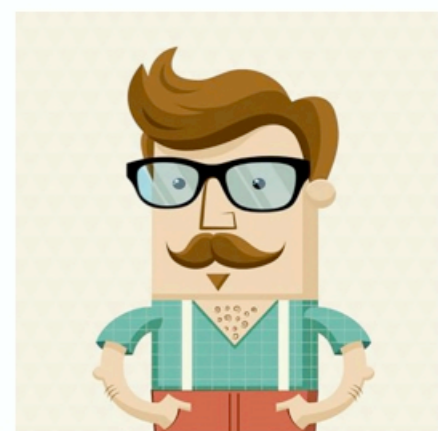
Среда разработки



Dev

CI

QA



Ops

Staging



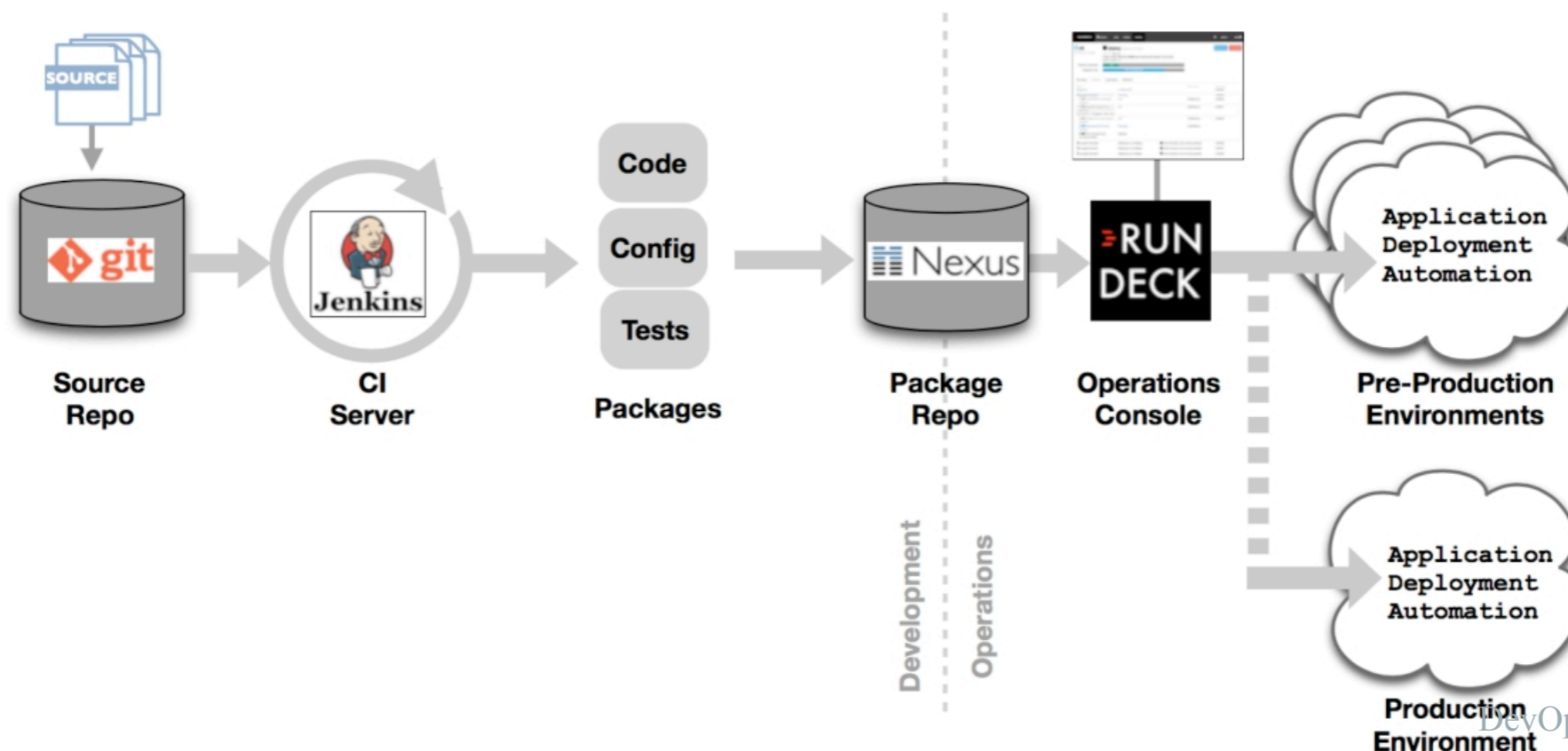
QA

Production



CI/CD

- Автоматизация процессов (сборки, деплой, инфраструктура)
- Стандартизация подходов к управлению окружениями
- Повышение прозрачности процессов
- Обеспечение командного взаимодействия



Проблематика

- Разработка и Тестирование нуждаются в независимых окружениях.
- Окружений становится много, непонятно кто за них отвечает
- Состояние окружений не синхронизировано

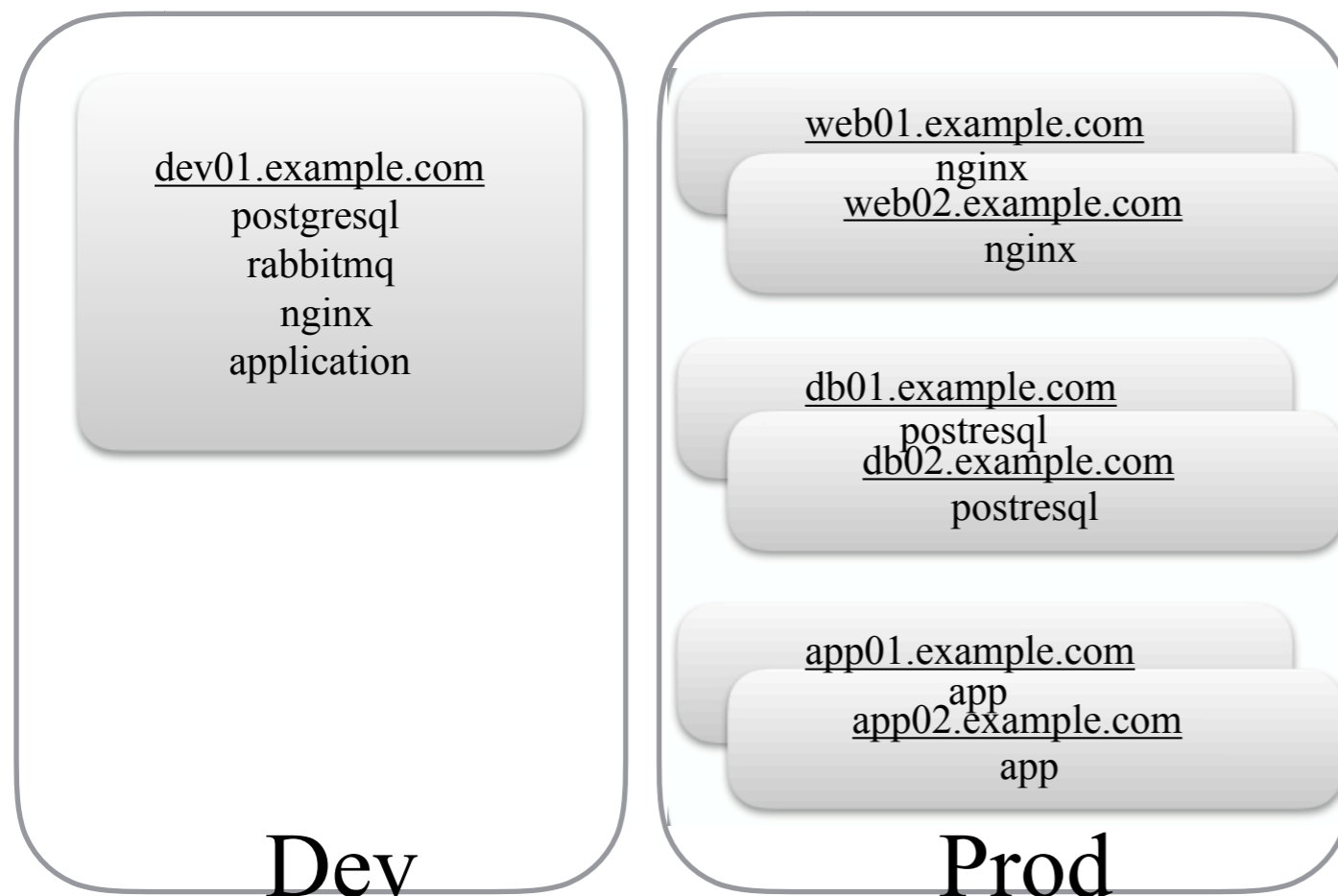


На test01 развернута новая фича с изменениями в структуре таблиц БД и мы не можем там тестировать багфикс .
На dev02 кто - то обновил openssl, а на остальных – забыл .

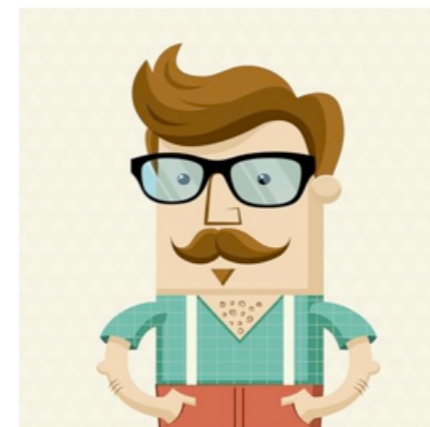


Проблематика

- Продукт может состоять из множества компонент
- Много компонент * много окружений = сложно
- Ограничения в ресурсах



Мы планируем запустить рекламную компанию и нужно провести нагрузочное тестирование на окружении аналогичном боевому



Процессы CI/CD

- Сборка приложения
- Контроль качества
- Публикация протестированных артефактов сборки в хранилище

- Создание нового окружения / Удаление более не нужного окружения
- Обновление окружения до состояния, описанного в виде кода
- Выкатка новой версии приложения / Откат до прошлой версии
- Применение изменений к базам данных этого окружения

Рассмотрим на примере

Веб приложение
Фронтенд : Angular2
Бекенд : Flask
База : PostgreSQL

web01.example.com
nginx, static files (html, js)

app01.example.com
python (api, flask app)

db01.example.com
postgresql

Please sign in

Sign in

Пример 1: новый функционал приложения

Задача : добавить кнопку “Запомнить меня”

Фронтенд

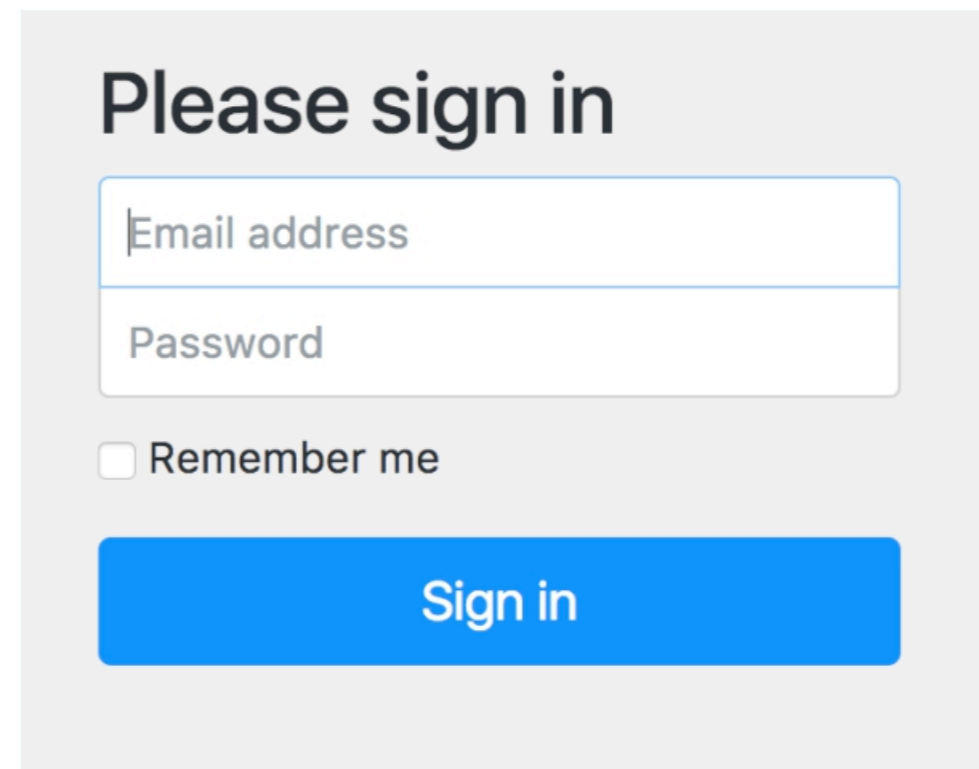
```
<form name="form" (ngSubmit)="f.form.valid && login()" #f="ngForm" novalidate>
  ...
  <div *ngIf="f.submitted && !password.valid" class="help-block">Password is required</div>
</div>
+ <div class="checkbox">
+   <label>
+     <input type="checkbox" value="{{remember_me}}"> Remember me
+
```

Бекенд

```
+ @app.route( "/session", methods=['POST'])
+ def session():
```

БД

```
+ CREATE TABLE session (
+   ...
+);
```



Please sign in

Email address

Password

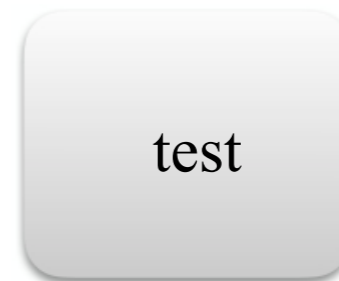
Remember me

Sign in

Пример 2: новая инфраструктура

Задача: добавление функционала рассылки

```
- hosts: email-servers
  roles:
    - role: postfix
      postfix_config_templates:
        -
          src: 'main.cf.j2'
          dest: '/etc/postfix/main.cf'
          owner: root
          group: root
          mode: '0644'
      postfix_tld: 'example.com'
```



Пример 3: Security disaster



Задача: Случился heartbleed. обновить openssl на всех серверах.

Как CI/CD помогает?

Контекст

Dev

Контекст '

Test

Контекст ''

Prod

Изменение

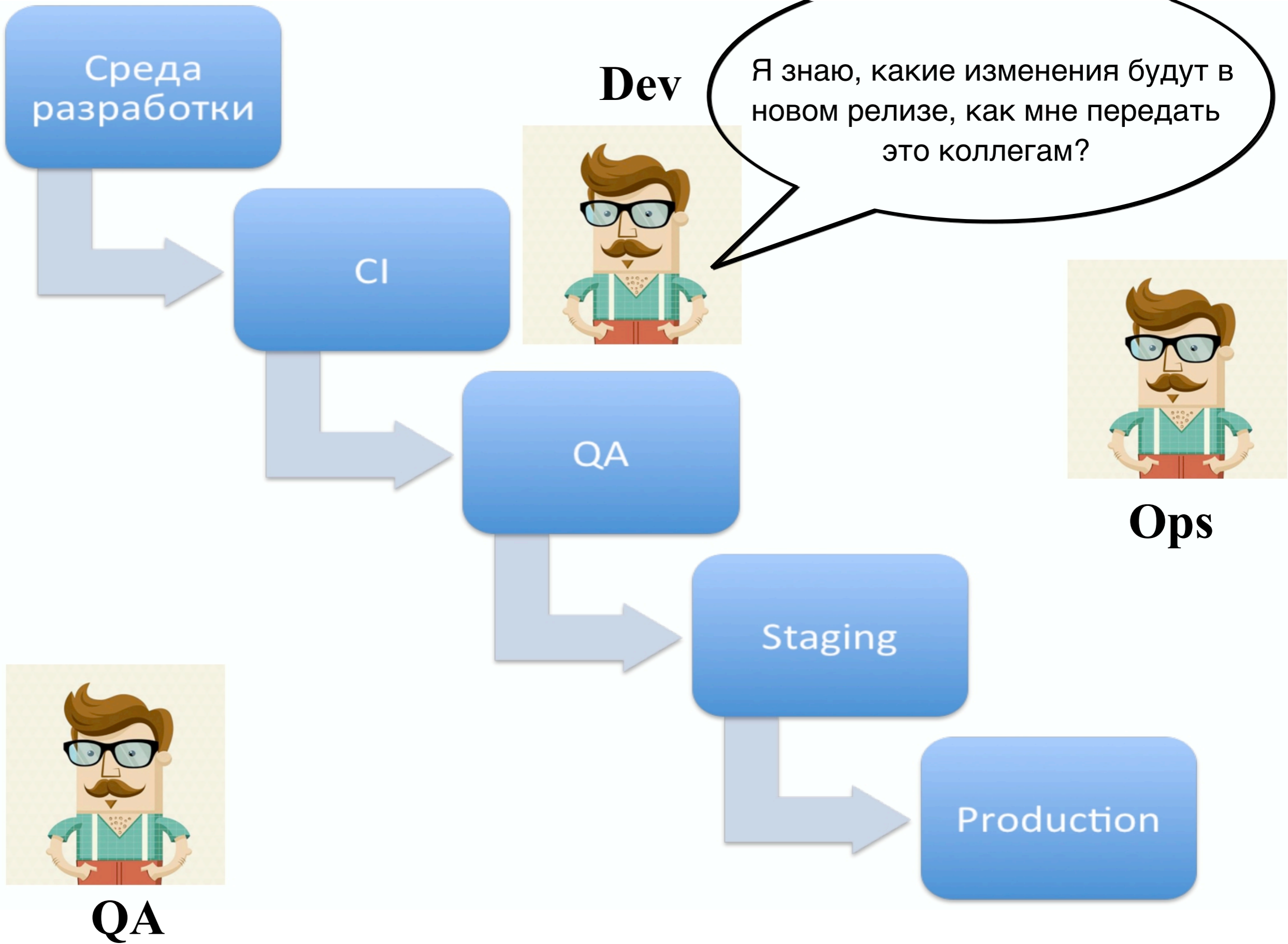
```
2 docker-compose.yml
@@ -8,4 +8,4 @@ services:
8     volumes:
9     - ./future-devops:/future-devops
10    - ./static:/static
11    - ./default.conf:/etc/nginx/conf.d/default.conf
+11  - ./nginx.conf:/etc/nginx/nginx.conf
```

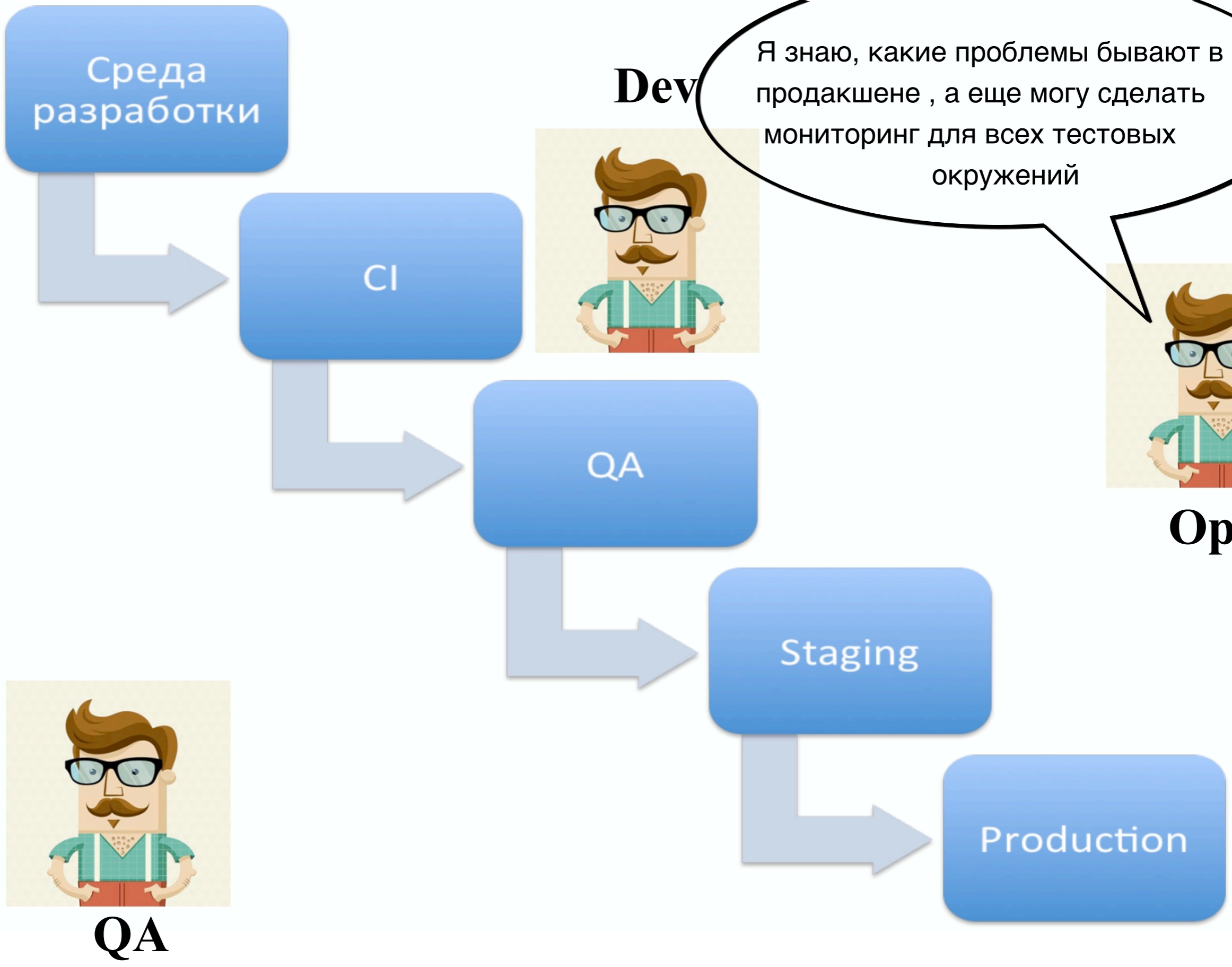
ПРИНЦИПЫ

- Создать повторяемый и надежный процесс поставки ПО
- Автоматизировать все , что можно . Интерфейс с кнопками
- Хранить все в системе контроля версий
- Получать раннюю обратную связь
- Совершенствоваться через повторения
- Встроенный контроль качества
- Выполнено , значит зарелизилось
- Релизим именно то , что было протестировано
- Каждый отвечает за процесс поставки ПО

ПРИНЦИПЫ

- Создать повторяемый и надежный процесс поставки ПО
- Автоматизировать все , что можно . Интерфейс с кнопками
- Хранить все в системе контроля версий
- Получать раннюю обратную связь
- Совершенствоваться через повторения
- Встроенный контроль качества
- Выполнено , значит зарелизилось
- Релизим именно то , что было протестировано
- **Каждый отвечает за процесс поставки ПО**





Dev

Я знаю, какие проблемы бывают в продакшене , а еще могу сделать мониторинг для всех тестовых окружений

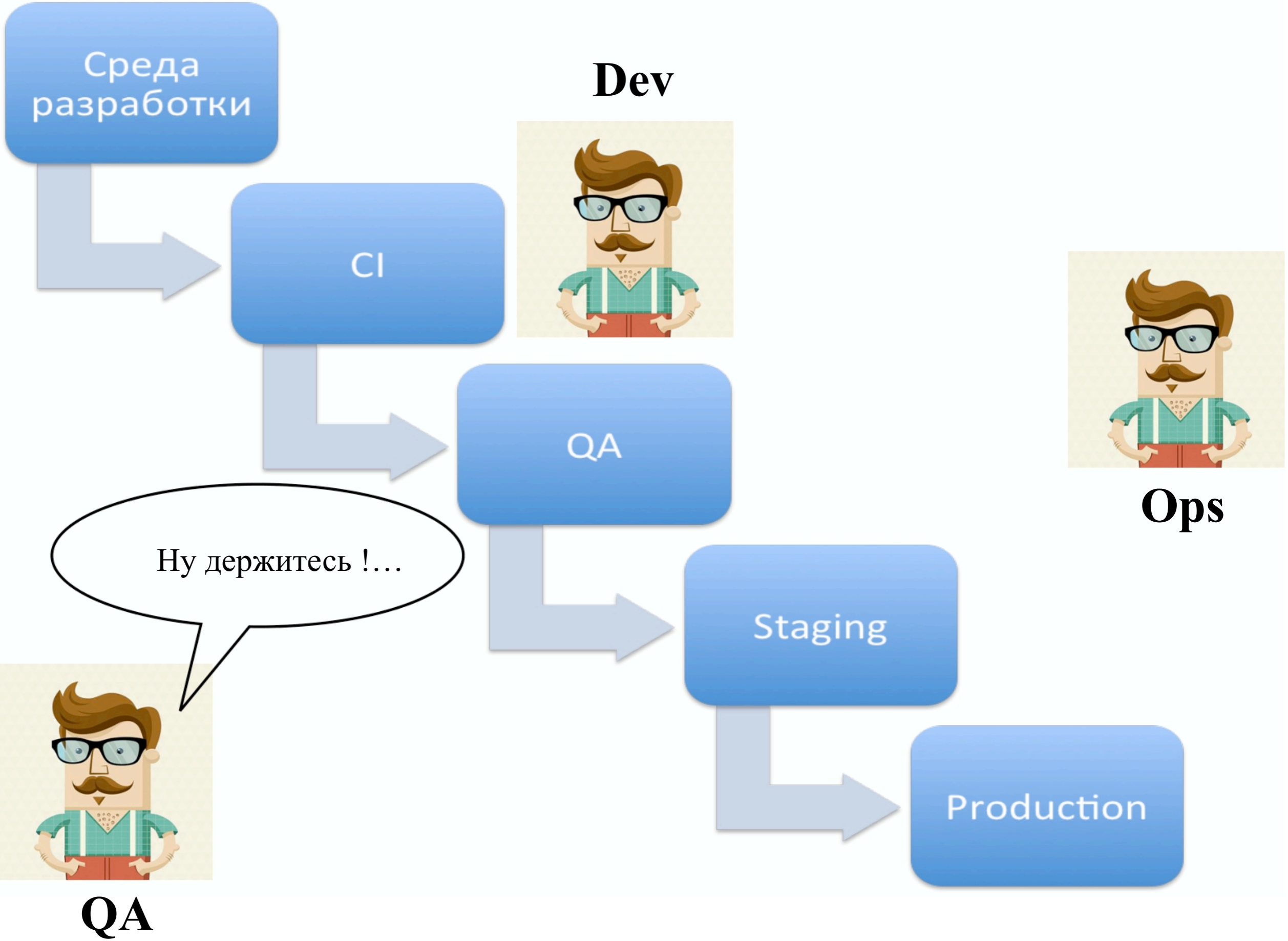


Ops



QA

Production



Обратная связь

- Светофор
- Отчеты о тестировании
- Контроль качества кода
- Уведомления о статусах в чат или почту
- Интеграция с таск-трекером

Контроль качества кода

- Измерения и визуализации
 - Покрытие тестами
 - Технический долг
 - Поиск потенциальных багов и уязвимостей
- Помогает найти потенциальные проблемы
- Помогает отслеживать тренды

SonarQube

The screenshot displays the SonarCloud web interface. At the top, there is a navigation bar with 'sonarcloud' logo, 'Projects', and 'Issues' tabs. A search bar on the right contains the text 'Search for projects, sub-projects and files...' and a 'Log in' button. Below the navigation bar, the main content area is titled 'Open Source' and includes sub-tabs for 'Projects', 'Issues', 'Quality Profiles', 'Rules', 'Quality Gates', and 'Members'. The 'Projects' tab is active, showing a list of projects. On the left side, there are filter sections for 'Quality Gate', 'Reliability', 'Security', and 'Maintainability', each with a list of status options and corresponding counts. The main project list shows four entries: 'AisLib application framework' (Passed), 'ajako-sample-angular2-spring-backend' (Failed), 'akcjamis' (Passed), and 'aksu-cloud' (Passed). Each project entry includes a 'Perspective' dropdown (set to 'Overall Status'), a 'Sort by' dropdown (set to 'Name'), and a search input field. The project details for each entry include a 'Passed' or 'Failed' status badge, a 'Last analysis' timestamp, and a row of five metrics: Reliability (E), Security (E), Maintainability (A), Coverage (38.3% for AisLib, - for others), and Duplications (0.6% for AisLib, 0.0% for others). A size indicator (M or XS) and a list of languages (Java, XML, JavaScript, etc.) are also shown for each project.

Project Name	Status	Last Analysis	Reliability	Security	Maintainability	Coverage	Duplications	Size	Languages
AisLib application framework	Passed	September 1, 2016, 10:42 PM	E	E	A	38.3%	0.6%	12k	Java, XML, ...
ajako-sample-angular2-spring-backend	Failed	October 9, 2016, 9:59 AM	D	E	A	-	0.0%	615	Java, XML, ...
akcjamis	Passed	August 30, 2016, 1:43 AM	D	E	A	-	7.6%	20k	Java, JavaScript, ...
aksu-cloud	Passed	March 9, 2017, 9:37 AM	-	-	-	-	-	-	-

SonarQube

The screenshot displays the SonarCloud web interface for a project named 'Activiti'. The top navigation bar includes 'sonarcloud', 'Projects', 'Issues', a search bar, and a 'Log in' button. The main content area is divided into three sections:

- Left Sidebar:** Shows a list of 5 issues out of 342. The top issue is highlighted: 'Cast one of the operands of this multiplication operation to a "double".' It is categorized as a 'Bug' with a 'Minor' severity. Below it, another identical issue is shown, followed by 'Equality tests should not be made with floating point values.' (Major severity).
- Code Editor:** Displays the source code for 'src/main/java/org/activiti/bpmn/BpmnAutoLayout.java'. The code includes a loop that iterates over points and calculates distances. A red box highlights a specific issue: 'NullPointerException might be thrown as 'closestPoint' is nullable here'. The issue is categorized as a 'Bug' with a 'Major' severity, is 'Reopened', and has a '10min effort' estimate. It was reported '2 years ago' and has a key 'L409'. The code snippet around the issue is:

```
402     for (mxPoint rhombusPoint : Arrays.asList(northPoint, southPoint, eastPoint, westPoint)) {
403         double distance = euclidianDistance(startPoint, rhombusPoint);
404         if (distance < closestDistance) {
405             closestDistance = distance;
406             closestPoint = rhombusPoint;
407         }
408     }
409     startPoint.setX(closestPoint.getX());
```
- Right Sidebar:** Shows the details of the highlighted issue, including its severity, status, and effort.

Хранение артефактов

Результатом билда иногда является артефакт
Например : jar, war, rpm, zip, docker image

Исходный код	Артефакт
Текст	Бинарник
Diffable	Not diffable
Версионирование по содержимому	Версионирование по имени
Перезаписывается	Не должен перезаписываться

Хранение артефактов

- Artifactory
- Nexus
- Archiva

- Прокси для внешних репозиториев
- Управление жизненным циклом артефакта
- Модель управления доступом
- Поиск
- REST API
- Single source of truth

Хранение артефактов

The screenshot displays the JFrog Artifactory web interface. The browser address bar shows the URL: `localhost:8081/artifactory/webapp/#/artifacts/browse/tree/General/libs-release-local/com/jeroenmols/awesomelibrary/awesomelibrary/1.0.0/awesomelibrary-1.0.0.aar`. The page title is "JFrog Artifactory".

The main content area is titled "Artifact Repository Browser" and shows a tree view of the repository structure. The selected artifact is `awesomelibrary-1.0.0.aar`. The right-hand pane displays the artifact's details:

- Info**
 - Name: `awesomelibrary-1.0.0.aar`
 - Repository Path: `libs-release-local/com/jeroenmols/awesomelibrary/awesomelibrary/1.0.0/awesomelibrary-1.0.0.aar`
 - Module ID: `com.jeroenmols.awesomelibrary:awesomelibrary:1.0.0`
 - Deployed by: `admin`
 - Size: `21.56 KB`
 - Created: `06-08-15 23:40:55 CEST`
 - Last Modified: `06-08-15 23:40:55 CEST`
 - Downloaded: `0`
- Dependency Declaration**
 - Build Tool: `Maven`
 - Code:

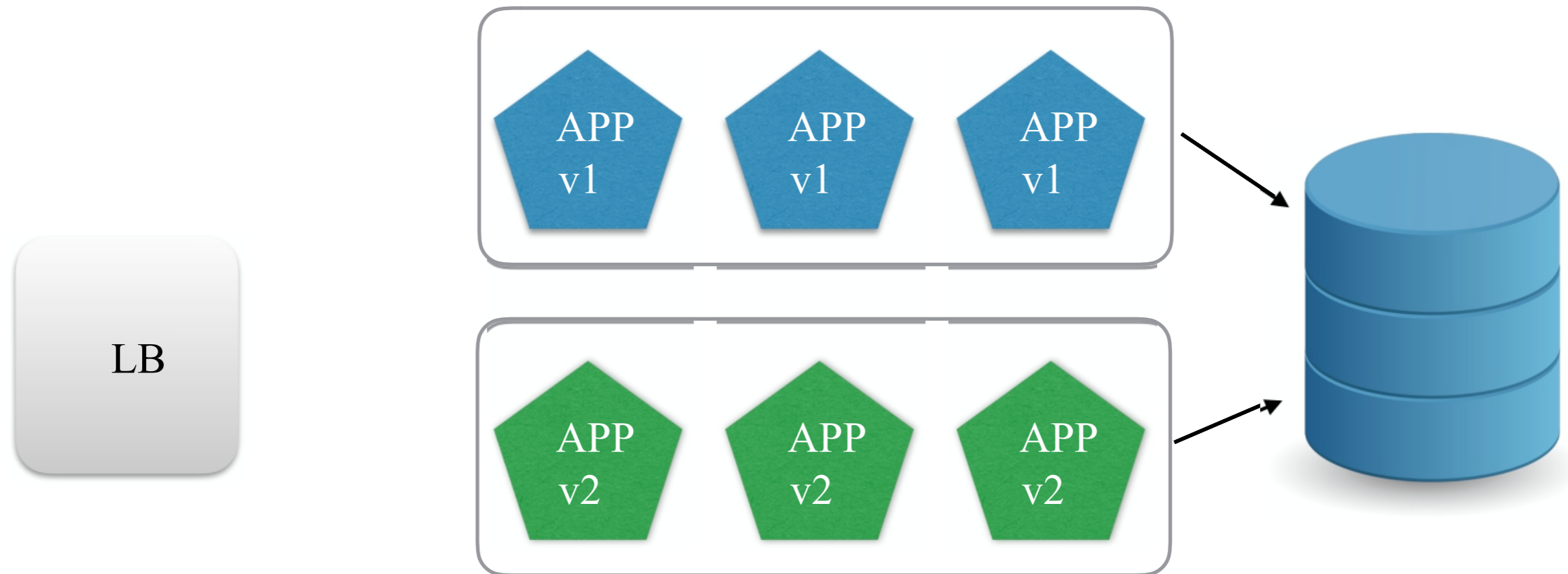
```
1 <dependency>
2   <groupId>com.jeroenmols.awesomelibrary</groupId>
3   <artifactId>awesomelibrary</artifactId>
4   <version>1.0.0</version>
5   <type>aar</type>
6 </dependency>
```
- Virtual Repository Associations**
 - `libs-release`
- Checksums**

Стратегии выкатки

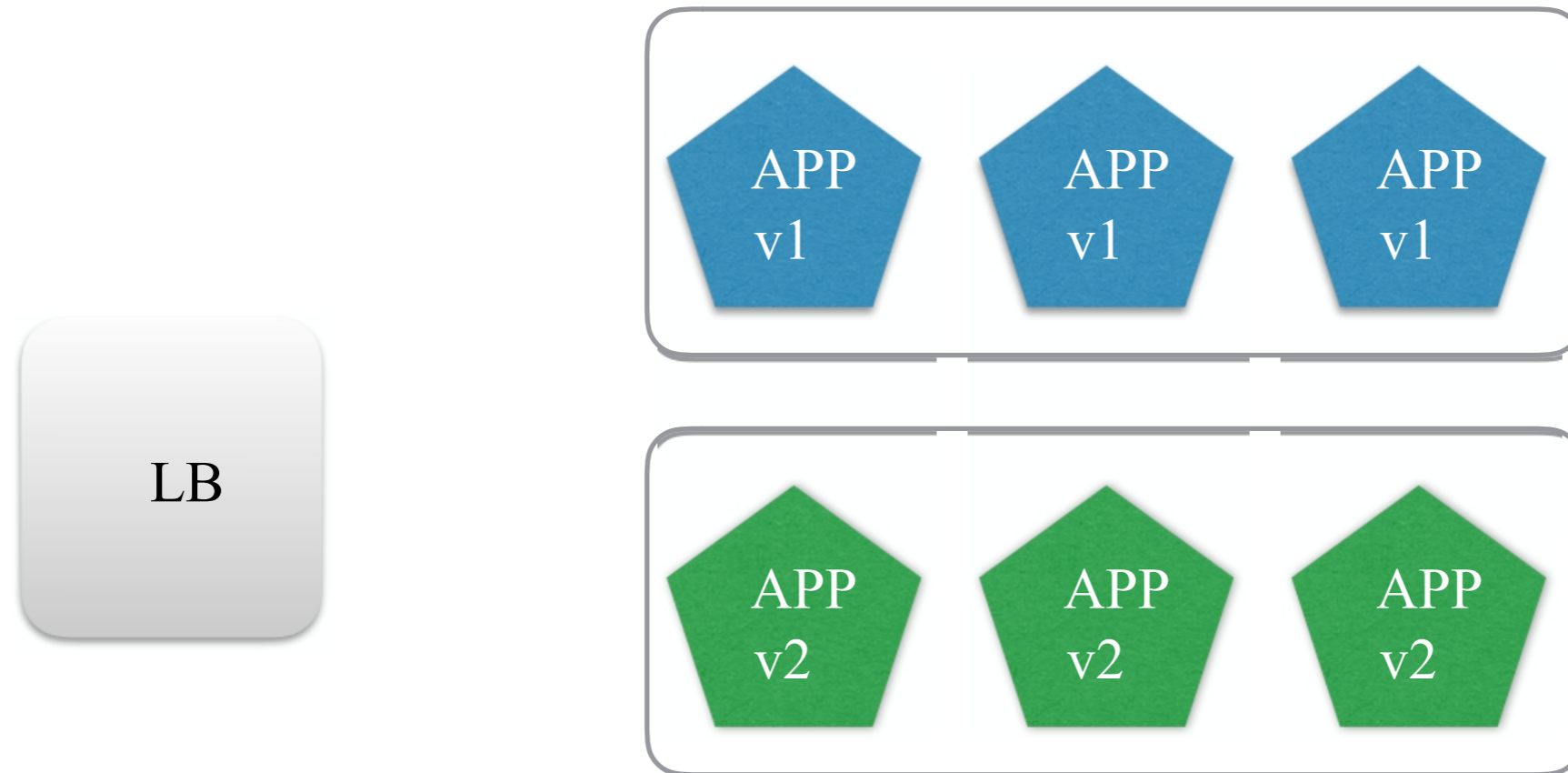
Для того , чтобы нам автоматически выкатывать изменения нужно реализовать обновление без простоев (zero-downtime deploy)

- Blue-green
- Canary
- Rolling

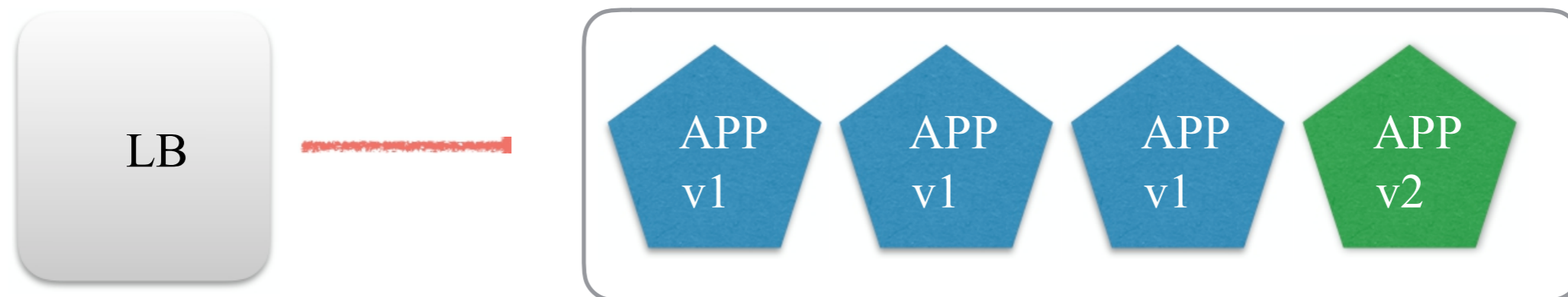
Blue-green deployment



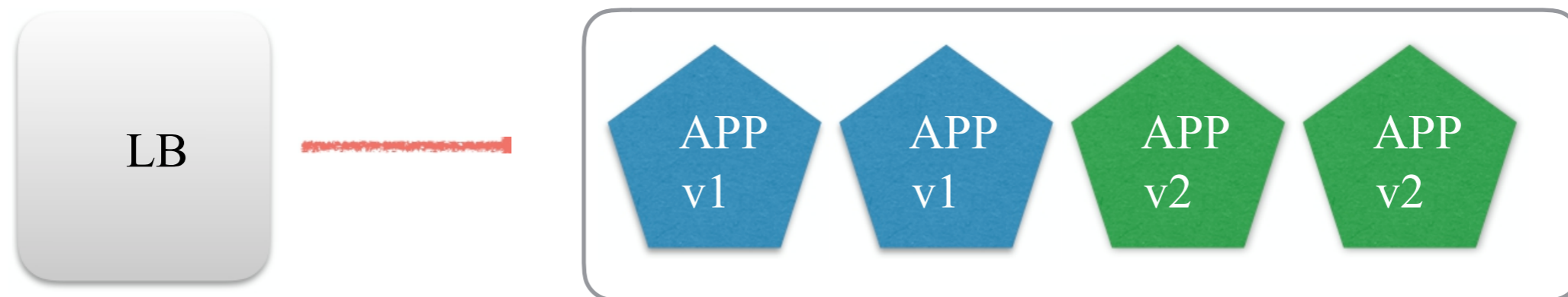
Blue-green deployment



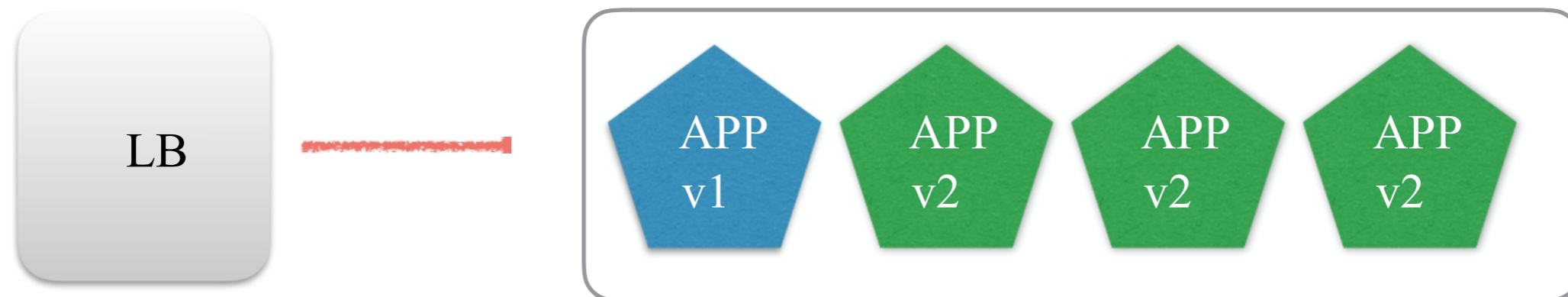
Canary deployment



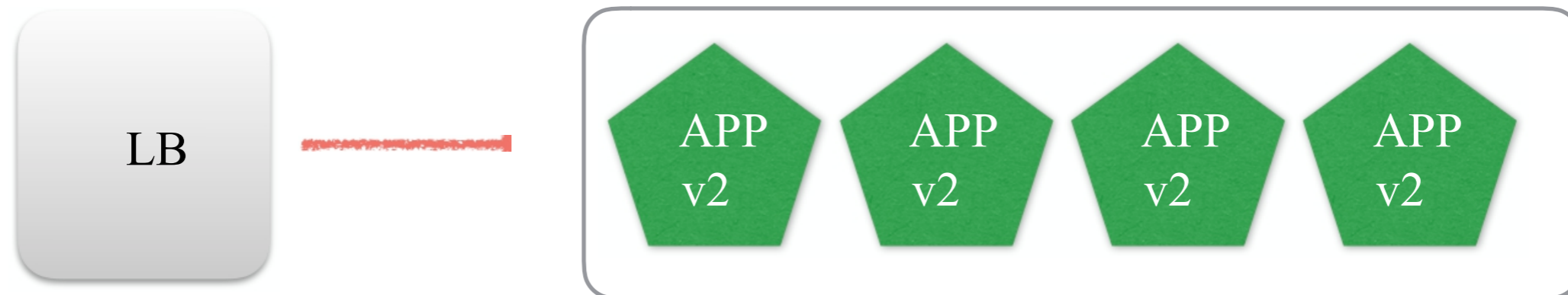
Rolling deployment



Rolling deployment



Rolling deployment



Нюансы

- Интеграция инструментов в процессы CI/CD
 - Как по созданию ветки в git создать окружение ?
 - Как будет запускаться ansible, настраивающий окружения?
 - Как подключать новые окружения к мониторингу ?
- CI-сервис в больших организациях
 - Windows / Linux / AIX / Docker
 - Много продуктов со своей спецификой
 - Состояние сборочных агентов уплывает
- CI/CD для компонент, поставляемых вендором

Метрики

- Скорость
- Качество

Метрики скорости

- **Mean Lead Time** – время на билд, тесты, деплой
- **Daily Change Rate** – успешных изменений в день
- **Mean Time To Environment** – время на подготовку нового окружения
- **Mean Time To Detect** – как долго баг может быть необнаруженным

Метрики качества

- **Build Failure Rate** – процент неуспешных сборок
- **Deployment Failure Rate** – процент неуспешных выкаток
- **Infra-Related Failure Rate** – процент неуспешных выкаток / сборок из -за проблем инфраструктуры
- **Automated Detection Rate** – процент проблем, выявленных на этапах тестирования

Инструменты

- Jenkins
- Bamboo от Atlassian
- TeamCity от JetBrains
- GitLab CI
- TravisCI
- Concourse CI
- Go CD
- Rundeck

Почему Gitlab CI

- Уже интегрирован в экосистему Gitlab (VCS, Issues, Docker registry!)
- Отдельный CI процесс под репозиторий (компонент)
- Быстрая обратная связь
- Визуализация процесса
- Достойная поддержка Docker
 - Тестирование внутри контейнеров
 - Тестирование с учетом других сервисов в контейнерах
- Обратная связь из коробки (много метрик , мониторинг)

Gitlab VCS

The screenshot shows the GitLab web interface for a project named 'example-shell-app'. The top navigation bar includes 'GitLab', 'Projects', 'Groups', 'Activity', 'Milestones', and 'Snippets'. The main content area displays the project details, including the repository name, a star/fork button, and the SSH URL. A commit history table is visible at the bottom, showing a recent commit 'add README' by 'Yury Ignatov'.

Navigation: Projects ▾ Groups Activity Milestones Snippets

Project: example-shell-app

Overview Details Activity Cycle Analytics

Repository Issues 0 Merge Requests 0 CI / CD Wiki Snippets Settings

Files (72 KB) Commit (1) Branch (1) Tags (0) Readme Add Changelog Add License Add Contribution guide Set up CI

master example-shell-app / + History Find file

add README f48b91cc
Yury Ignatov committed less than a minute ago

Name	Last commit	Last Update
README.md	add README	less than a minute ago

Gitlab Issues

The screenshot shows the GitLab interface for a project named 'example-js-app'. The top navigation bar includes 'Projects', 'Groups', 'Activity', 'Milestones', and 'Snippets'. The main content area displays an issue titled 'Add more tests' with the description 'Our application has very low code coverage. Someone has to deal with it.' The issue is marked as 'Open' and was created by user 'yignatov'. Below the description are reaction buttons for thumbs up (0), thumbs down (0), and a smiley face. A 'Create a merge request' button is visible. A comment box is open, showing a 'Write' tab and a rich text editor with options for bold, italic, quote, code, list, and link. The right sidebar contains a 'Todo' section with a 'Mark done' button, and fields for 'Assignee' (yignatov), 'Milestone' (None), 'Time tracking' (No estimate or time spent), 'Due date' (Oct 14, 2017), 'Confidentiality' (Not confidential), and '1 participant'. A 'Notifications' section has an 'Unsubscribe' button. The bottom left of the sidebar has a 'Collapse sidebar' button.

Gitlab Pipelines

GitLab Community Edition

Overview
Repository
Issues 9,227
Merge Requests 461
CI / CD
Pipelines
Jobs
Schedules
Charts
Snippets
Members

GitLab.org > GitLab Community E... > Pipelines > #12682265

running Pipeline #12682265 triggered 8 minutes ago by Grzegorz Bizon

Extract ensure stage service from commit status class

85 jobs from `fix/gb/ensure-that-job-belongs-to-stage` (queued for 13 seconds)

501a19f4

Pipeline Jobs 85

Build	Prepare	Test	Post-test
build-package	retrieve-tests-... (checked)	codequality	coverage
review-docs-d...	setup-test-env	db:check-sch...	flaky-example...
		db:migrate:res...	lint:javascript:r...
		db:migrate:res...	
		db:rollback-m...	
		db:rollback-pg	

<< Collapse sidebar

Gitlab Pipelines

The screenshot displays the GitLab web interface. The top navigation bar includes 'GitLab', 'Projects', 'Groups', 'Snippets', and 'Help'. On the right, there are links for 'This project', 'Search', and 'Sign in / Register'. The left sidebar shows the 'CI / CD' section with options for Pipelines, Jobs, Schedules, and Charts. The main content area shows a log for a job named 'setup-test-env'. The log text is as follows:

```
Showing last 500.3056640625 KiB of log - Complete Raw  
Fixing public/assets/notify-abfe30822d82207505a29881cc27c46d0cf2f8bf68481bd421ba1ce23ca7c508.css  
Fixing public/assets/notify-abfe30822d82207505a29881cc27c46d0cf2f8bf68481bd421ba1ce23ca7c508.css.gz  
$ bundle exec ruby -Ispec -e 'require "spec_helper" ; TestEnv.init'  
Knapsack time offset warning enabled!  
  
==> Setting up GitLab Shell...  
Warning  
You are running as user root, we hope you know what you are doing.  
Things may work/fail for the wrong reasons.  
For correct results you should run this as user git.  
  
mkdir -p /builds/gitlab-org/gitlab-ce/tmp/tests/.ssh: OK  
chmod 700 /builds/gitlab-org/gitlab-ce/tmp/tests/.ssh: OK  
mkdir -p /builds/gitlab-org/gitlab-ce/tmp/tests/repositories: OK  
chmod ug+rwX,o-rwx /builds/gitlab-org/gitlab-ce/tmp/tests/repositories: OK  
mkdir -p /builds/gitlab-org/gitlab-ce/tmp/tests/non-existent-repositories: OK  
chmod ug+rwX,o-rwx /builds/gitlab-org/gitlab-ce/tmp/tests/non-existent-repositories: OK  
Creating/Repairing hooks symlinks for all repositories  
done  
GitLab Shell setup in 40.522782655 seconds...  
  
==> Setting up Gitaly...  
Warning  
You are running as user root, we hope you know what you are doing.  
Things may work/fail for the wrong reasons.  
For correct results you should run this as user git.  
  
Gitaly setup in 28.153000821 seconds...  
$ scripts/gitaly-test-build  
if test -z "/usr/local/bundle" ; then cd ruby && bundle config --local path vendor/bundle; fi  
cd ruby && bundle install  
Warning: the running version of Bundler (1.14.6) is older than the version that created the lockfile (1.15.4). We suggest you upgrade to the latest version of Bundler by running `gem install bundler`.  
Fetching gem metadata from https://rubygems.org/.....  
Fetching version metadata from https://rubygems.org/.
```

On the right side of the interface, the job details for 'setup-test-env' are shown:

- Merge Request: !14724
- Duration: 9 minutes 37 seconds
- Runner: #37398
- Tags: [gitlab-org](#)
- Commit [501a19f4](#) in [!14724](#)
- Extract ensure stage service from commit status class
- Pipeline #12682265 from [fix/gb/ensure-that-job-belongs-to-stage](#)
- Dropdown menu: prepare
- Job list: [setup-test-env](#) (running), [retrieve-tests-metadata](#) (completed)