

ЛЕКЦИЯ 5

Системы
отслеживания ошибок

СИСТЕМЫ ОТСЛЕЖИВАНИЯ ОШИБОК

Система отслеживания ошибок (англ. bug tracking system) — прикладная программа, разработанная с целью помочь разработчикам программного обеспечения (программистам, тестировщикам и прочим) учитывать и контролировать ошибки (баги), найденные в программах, пожелания пользователей, а также следить за процессом устранения этих ошибок и выполнением или невыполнением пожеланий.

ОШИБКА (БАГ)

Баг (англ. bug — жук) — жаргонное слово, обычно обозначающее ошибку в программе или системе, которая выдаёт неожиданный или неправильный результат.

«Баги» могут быть обнаружены:

- в процессе тестирования программы (тестировщиками);
- в процессе отладки программы (разработчиками);
- сторонними пользователями приложения.

ИСТОРИЯ ВОЗНИКНОВЕНИЯ ТЕРМИНА «БАГ»

По легенде, **9 сентября 1947 года** учёные Гарвардского университета, тестиовавшие вычислительную машину Mark II Aiken Relay Calculator, нашли мотылька, застрявшего между контактами электромеханического реле и Грейс Хоппер произнесла этот термин.

Извлечённое насекомое было вклеено скотчем в технический дневник, с сопроводительной надписью: «*First actual case of bug being found*» (англ. «Первый случай обнаружения бага»). Этот забавный факт положил начало использованию слова «**debugging**» в значении «отладка программы».

МЕТОДОЛОГИЯ AGILE

Для понимания того, как команда работает над ошибками, рассмотрим методологию разработки Agile (читается «аджайл»), так как системы отслеживания ошибок используют схожие принципы.

Agile-методы – это методы разработки программного обеспечения, ориентированные на разработку по итерациям.

Суть методологии заключается в том, что разработчики от итерации к итерации выполняют требования заказчика, постоянно улучшая свой продукт.

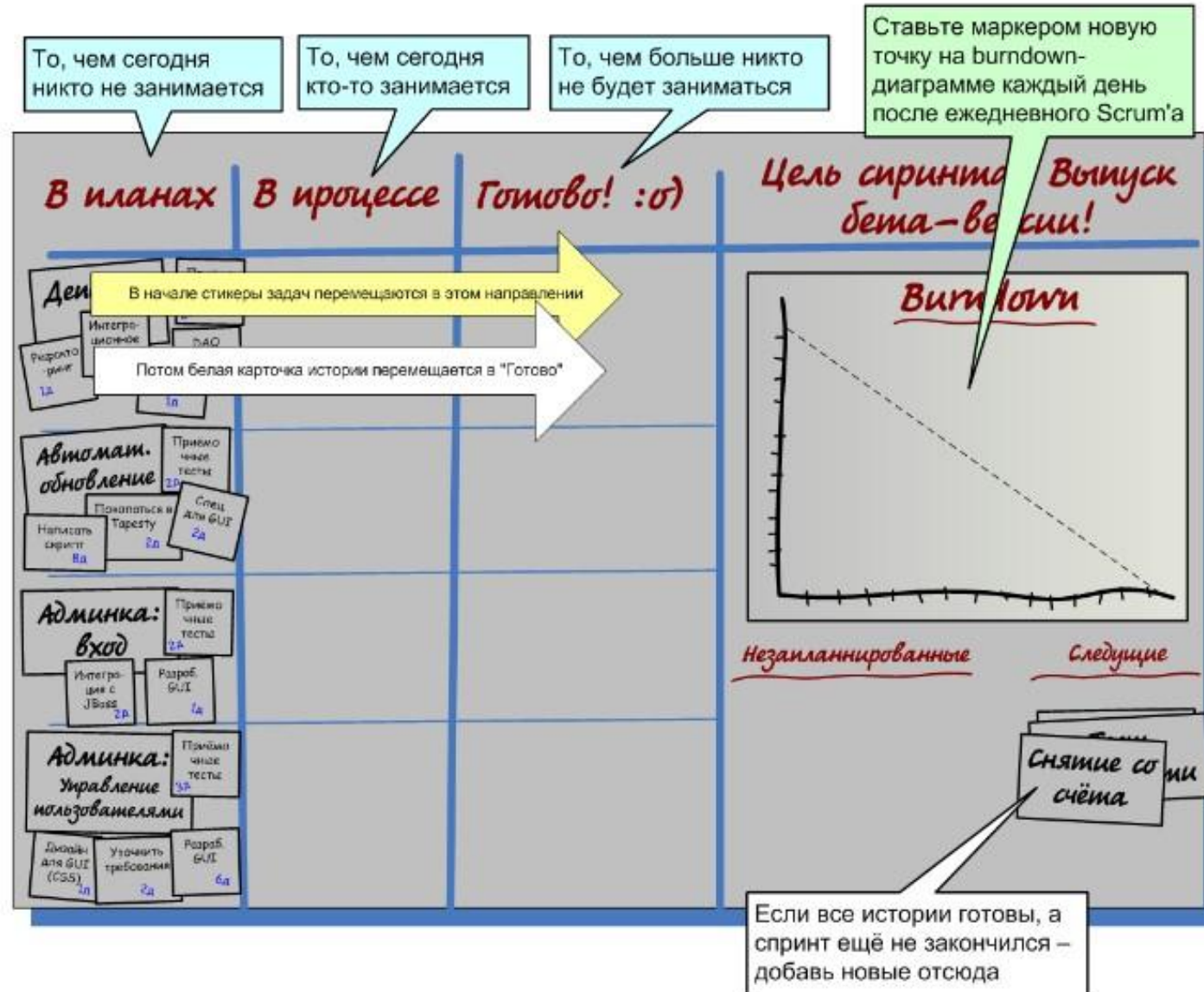


МЕТОДОЛОГИЯ AGILE

Для контроля выполнения задач в Agile используется доска, по которой можно отслеживать процесс выполнения задач. Основные состояния:

- задачи, которые еще не выполняются, но планируются на эту итерацию;
- задачи, которые сейчас разрабатываются;
- задачи, которые уже выполнены и будут выпущены в конце итерации.

AGILE - ДОСКА



ОТЧЁТЫ ОБ ОШИБКАХ

Для повышения качества программного обеспечения пользуются специальными программами, цель которых — отловить ошибку в целевом приложении, собрать необходимую информацию об её симптомах и отправить отчёт по интернету к разработчикам данного ПО.

Например, в операционную систему Windows встроена утилита **Dr. Watson**, которая по умолчанию отлавливает ошибки в приложениях пользователя, и отправляет отчёт на специальный сервер компании Microsoft. Также в качестве примера можно привести аналогичные библиотеки **Breakpad** и **CrashRpt**.

СОСТАВ ИНФОРМАЦИИ О ДЕФЕКТЕ

Главный компонент системы отслеживания ошибок - база данных, содержащая сведения об обнаруженных дефектах. Эти сведения могут включать в себя:

- номер (идентификатор) дефекта;
- кто сообщил о дефекте;
- дата и время, когда был обнаружен дефект;
- версия продукта, в которой обнаружен дефект;
- серьёзность (критичность) дефекта и приоритет решения;
- описание шагов для выявления дефекта;
- кто ответственен за устранение дефекта;
- обсуждение возможных решений и их последствий;
- текущее состояние (статус) дефекта;
- версия продукта, в которой дефект исправлен.

ЖИЗНЕННЫЙ ЦИКЛ ДЕФЕКТА

Начало

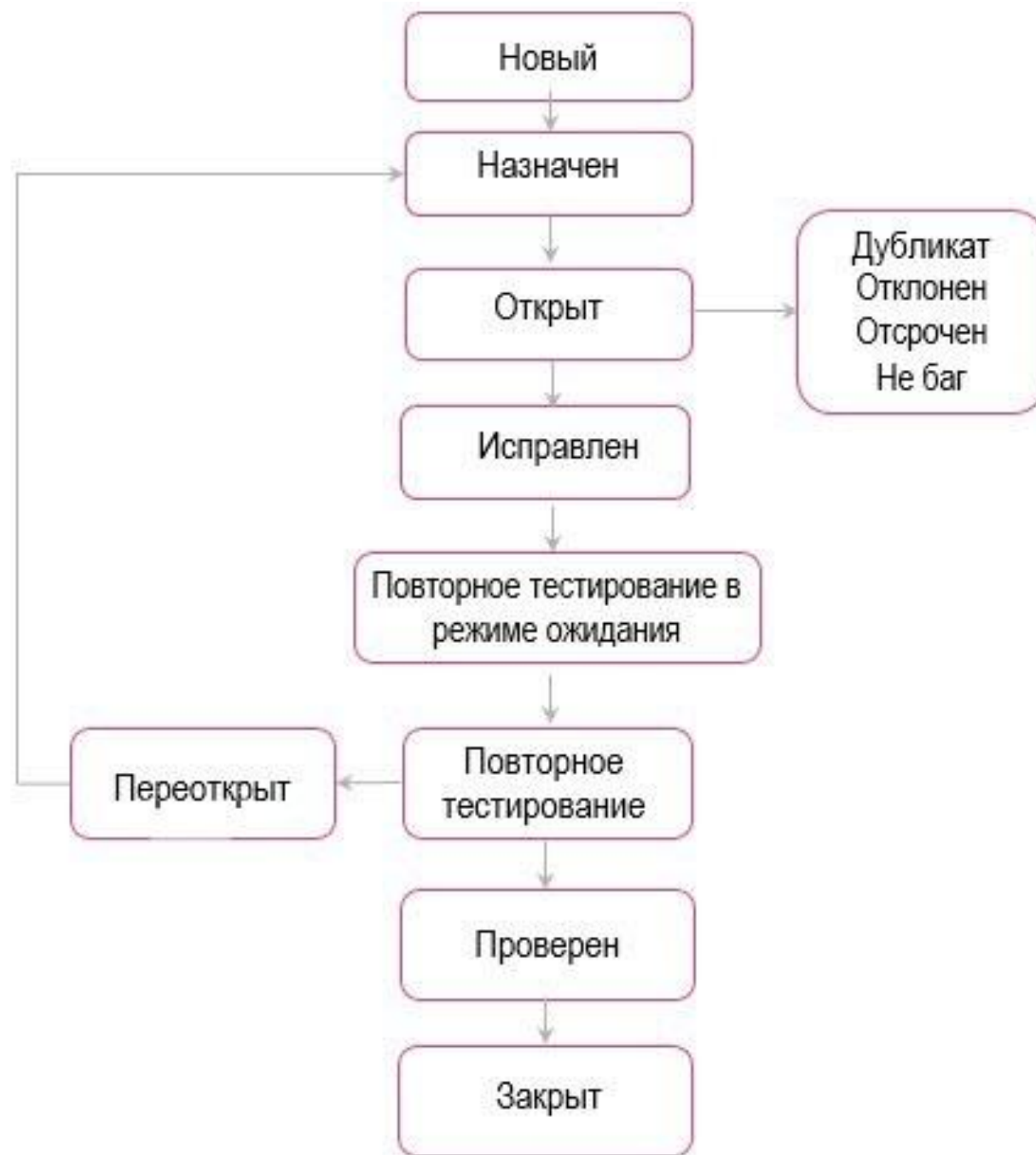
Типичный жизненный цикл дефекта:

1. **Новый** — дефект зарегистрирован тестировщиком
2. **Назначен** — назначен ответственный за исправление дефекта
3. **Разрешён** — дефект переходит обратно в сферу ответственности тестировщика. Как правило, сопровождается разрешением ошибки, например:
 - **Исправлено** (*исправления включены в версию такую-то*);
 - **Дубль** (*повторяет дефект, уже находящийся в работе*);
 - **Не исправлено** (*работает в соответствии со спецификацией, имеет слишком низкий приоритет, исправление отложено до следующей версии и т.п.*);
 - **«У меня всё работает»** (*запрос дополнительной информации об условиях, в которых дефект проявляется*);

ЖИЗНЕННЫЙ ЦИКЛ ДЕФЕКТА

Окончание

4. Тестировщик проводит проверку исправления, в зависимости от чего дефект либо снова переходит в статус “**Назначен**” (если он описан как исправленный, но не исправлен), либо в статус “**Закрит**”.
5. **Открыт повторно** — дефект вновь найден в другой версии.



СИСТЕМЫ КОНТРОЛЯ ОШИБОК

Наиболее известные из бесплатные:

- The Bug Genie
- Bugzilla
- eTraxis
- MantisBT
- Trac
- EmForge
- Redmine

BUGZILLA

Bugzilla — свободная система отслеживания ошибок с веб-интерфейсом.

В 1998 году Bugzilla была выпущена как открытое программное обеспечение компанией Netscape. В настоящее время система разрабатывается «**Mozilla Foundation**». С одной стороны, Bugzilla довольно проста, с другой стороны, там есть всё, что нужно для ведения ошибок в типичном проекте.

По функциональности Bugzilla сейчас отстаёт от многих современных систем отслеживания ошибок. Разработчики считают, что одна из причин этого — выбор Perl в качестве языка реализации Bugzilla, рассматривается возможность переписать её на каком-нибудь другом языке программирования.

Для работы Bugzilla требуются:

веб-сервер с поддержкой CGI (рекомендуется Apache);

поддержка языка Perl база данных MySQL, PostgreSQL, или Oracle;

ETRAXIS

eTraxis — это бесплатная веб-система для отслеживания записей, распространяемая под лицензией GPL v2. Наиболее популярное использование системы — контроль ошибок.

Базовые возможности:

- гибкая настройка жизненного цикла записей (ошибок, улучшений, любых запросов);
- управление правами групп — вплоть до отдельного поля; также есть авторизация через LDAP;
- зависимости между записями;
- сохранение полной истории работы с записями — запоминается любое изменение в любом поле;
- большое количество локализаций;
- нотификации/подписки/напоминания;

New record

Project: eTraxis

Template: Defect report

Subject: [required] ⓘ

— New

Priority: [required] ⓘ Medium

DESCRIPTION

Critical - must resolve in the specified milestone

High - strongly want to resolve in the specified milestone

Medium - normal priority

Low - might slip to later milestone

Found in version: ⓘ

Description: ⓘ

[b]What steps will reproduce the problem?[/b]

- 1.
- 2.
- 3.

[b]What is the expected output? What do you see instead?[/b]

ⓘ All fields marked as required should be filled in.

ⓘ You can insert link to another record by specifying "rec#" and its number (e.g. "rec#305").

OK Cancel

REDMINE








Redmine — открытое серверное веб-приложение для управления проектами и отслеживания ошибок. Redmine написан на Ruby и представляет собой приложение на основе широко известного веб-фреймворка Ruby on Rails. Распространяется согласно GNU GPL.

Функциональные возможности:

- ведение нескольких проектов;
- гибкая система доступа, основанная на ролях;
- система отслеживания ошибок;
- ведение новостей проекта, документов и управление файлами;
- оповещение об изменениях с помощью RSS-потоков и электронной почты;
- учёт временных затрат;
- лёгкая интеграция с репозиториями (SVN, CVS, Git, Mercurial, Bazaar и Darcs);
- создание записей об ошибках на основе полученных писем;
- многоязыковой интерфейс (в том числе русский);
- поддержка СУБД MySQL, PostgreSQL, SQLite, Oracle.

REDMINE

Новая задача

Трекер *	Bug
Тема *	Test
Описание	<div><div>B I U S C H1 H2 H3     pre   </div><div>Test</div></div>
Статус *	New
Приоритет *	Normal
Назначена	Andreas Bagattini
Версия	Test
Начата	2018-11-24
Дата выполнения	2018-11-30
Оценка времени	15 час(а,ов)
Готовность	50 %
Файлы	<div>Выбрать файлы</div> Файл не выбран (Максимальный размер: 100 КБ)

[Создать](#) [Создать и продолжить](#) [Предпросмотр](#)

JIRA

Atlassian JIRA — система отслеживания ошибок, предназначена для организации общения с пользователями. Разработана компанией Atlassian Software Systems. Имеет веб-интерфейс.

JIRA создавался в качестве замены Bugzilla и во-многом повторяет архитектуру Bugzilla. Система позволяет работать с несколькими проектами. Для каждого из проектов создаёт и ведёт схемы безопасности и схемы оповещения.

JIRA

Базовые возможности:

- понятный пользовательский интерфейс;
- хорошая расширяемость;
- управление ошибками, проблемами, задачами.
- высокий уровень безопасности;
- возможность составить план работы; планировать своё время на решение рабочих задач;
- отличная система для управления задачами в мультиязычном сегменте;
- наличие API и множества плагинов, по этому возможности расширений не ограничены. Существует более 100 готовых бесплатных расширений, есть возможность написания собственных;
- есть возможность настройки для любой сферы деятельности проекта или всей компании.

JIRA

Summary*

Priority

Component/s

Start typing to get a list of possible matches or press down to select.

Affects Version/s

Start typing to get a list of possible matches or press down to select.

Environment

For example operating system, software platform and/or hardware specifications (include as appropriate for the issue).

MongoDB Version

Topology

Description

Style **B** **I** U

Hello, Dear developers!

Driver reads array elements less than two incorrectly.

1. I have this code (which selects array of TEST_CASES from collection):

```
auto collectionName = CollectionCnst::VARIANT;
auto collection = db.collection(collectionName);
auto doc = bsoncxx::v_noabi::builder::basic::document{};
doc = bsoncxx::v_noabi::builder::basic::document{};
doc.append(kvp(ID, bsoncxx::oid(ourVariantId)));
auto cursor = collection.find_one({doc});
if (cursor->view().length() == 0) {
    // not found
    throw std::exception();
}
// select testCase's
auto testCases = cursor->view()(FieldCnst::TEST_CASES).get_array();
return testCases.value;
```

2. I want to iterate through it (foreach loop):

```
for (auto &testCase: testCases) {
    // do smth
}
```

3. If I have 1 element in collection, loop don't invokes no one time. If I have two elements, it invokes 1 time. If I have three or more array elements, it invokes exactly through all elements.

Attachment

Drop files to attach, or browse.

- Backwards Compatibility ☐ None ☒ Fully Compatible ☐ Minor Change ☐ Major Change

If this change could break old code

Labels

Begin typing to find and create labels or press down to select a suggested label.

Story Points

Measurement of complexity and/or size of a requirement.

Epic Link