

Применение статического анализа кода в преподавании и в разработке свободного ПО



Докладчик:
Георгий Грибков

Содержание

1. Статический анализ: краткий экскурс
2. Применение статического анализа в высшей школе
3. Использование в студенческих и открытых проектах

Статический анализ: краткий экскурс

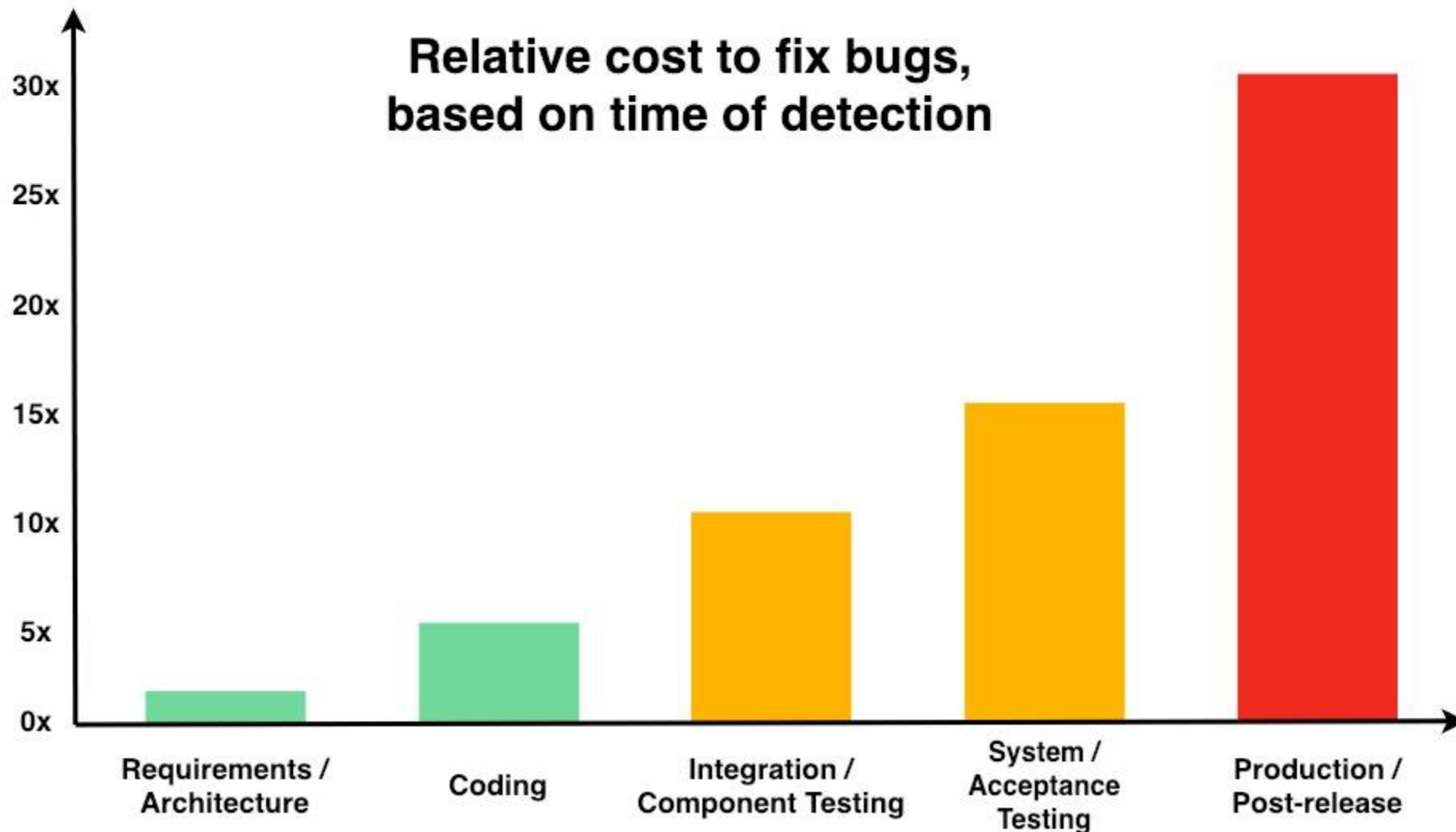
Как улучшить качество кода

- Делать сразу правильно
- Юнит-тесты
- Регрессионное тестирование
- Code review
- ...а можно ли как-нибудь еще?
- Можно! Например – автоматические средства анализа

Автоматические средства анализа кода

- Статические анализаторы: код проверяется без его выполнения
- Динамические анализаторы: код проверяется во время его выполнения
- Обе методологии отлично дополняют друг друга

Цена исправления ошибки



Недостатки статического анализа

- Имеет ложные срабатывания
- Сложно с многопоточностью
- Полностью не избавляет от code review

Достоинства статического анализа

- Полностью покрывает код
- Значительно быстрее, чем динамический анализ
- Более удобен для больших проектов

Достоинства статического анализа

- Может проверить code style или соответствие стандарту кодирования (MISRA, AUTOSAR C++)
- *Прост в использовании*
- Помогает программистам обучаться и обучать

Применение статического анализа в высшей школе

Преподавателям

- Проверка домашних заданий
- Проверка курсовых работ
- Экономия времени преподавателя

Студентам

Младший разработчик Java

от 60 000 до 90 000 руб. на руки

000 Серебряная Пуля ✓

● Павелецкая, ● Павелецкая, ● Пролетарская, Москва, Дербеневская набережная, 11



Откликнуться



Требуемый опыт работы: 1–3 года

Полная занятость, гибкий график

В связи с развитием продуктовой линейки мы ищем junior-разработчика, студента или молодого специалиста в команду Java разработки.

Обязанности:

- Разработка расширений для платформы SonarQube в составе команды
- Создание логики проверки исходного кода на основе имеющегося детального списка требований
- Удобная и красивая разработка на Java с использованием методик TDD, CI/CD, статического анализа кода и т.п.

Требования:

- Знание Java Core
- Минимальный опыт в Java, результаты прохождения курсов

списка требований
статического анализа кода

DevOps инженер (CMake)

от 150 000 руб.

Вяртсиля, Центр цифровых технологий ✓

● Василеостровская, ● Приморская, ● Спортивная, Санкт-Петербург, Малый пр.ВО, 54
к.4



Откликнуться



Требуемый опыт работы: 3–6 лет
Полная занятость, полный день

Компания Вяртсиля Цифровые Технологии ищет **DevOps инженера** на проекты разработки нового поколения судового навигационного приложения.

Задачи DevOps инженера:

Поддержка и развитие системы непрерывной интеграции кросс-платформенного десктопного продукта в частности:

- Повышение стабильности прохождения интеграционных тестов.
- Повышение скорости сборки и интеграционных тестов.
- Написание groovy скриптов для Jenkins
- Перевод системы сборки на новый стек технологий. Пример: перевод сборки библиотеки со scops на CMake, перевод сборочной системы с Visual Studio 2015 на Visual Studio 2019/
- Внедрение статического анализа кода на базе PVS-Studio и SonarQube.
- Выпуск пресобранных пакетов сторонних библиотек и внутренних продуктов. Перевод пакетной системы с самописной реализации на Conan.
- Выпуск нескольких взаимосвязанных продуктов из общей кодовой базы.

...е стабильности пр
...ышение скорости сборки и ин.
...аписание groovy скриптов для Jenk.
...Перевод системы сборки на новый сте
...перевод сборочной системы с Visual Stu
— Внедрение статического анализа кода н
— Выпуск пресобранных пакетов сторонни
...самописной реализации на Conan.
...Выпуск нескольких взаимосвязанных

Студентам

- Обучение развивающейся технологии
- Самостоятельная проработка проблем
- Облегчение процесса разработки
- Знакомство с паттернами ошибок

Примеры паттернов (Vangers)

```
void aciPackFile(....)
{
    int sz,sz1;
    char *p,*p1;
    ....
    p = new char[sz];
    p1 = new char[sz1];
    ....
    delete p;
    delete p1;
}
```



Примеры паттернов (Vangers)

```
void aciPackFile(....)
{
    int sz,sz1;
    char *p,*p1;
    ....
    p = new char[sz];
    p1 = new char[sz1];
    ....
    delete p;    // <=
    delete p1;   // <=
}
```



Примеры паттернов (Vangers)

```
void aciPackFile(....)
{
    int sz,sz1;
    char *p,*p1;
    ....
    p = new char[sz];
    p1 = new char[sz1];
    ....
    delete [] p;
    delete [] p1;
}
```



Примеры паттернов (Apache HTTP Server)

```
static void MD4Transform(  
    apr_uint32_t state[4],  
    const unsigned char block[64])  
{  
    apr_uint32_t a = state[0], b = state[1],  
                c = state[2], d = state[3],  
                x[APR_MD4_DIGESTSIZE];  
  
    ....  
    /* Zeroize sensitive information. */  
    memset(x, 0, sizeof(x));  
}
```



Примеры паттернов (Apache HTTP Server)

```
static void MD4Transform(  
    apr_uint32_t state[4],  
    const unsigned char block[64])  
{  
    apr_uint32_t a = state[0], b = state[1],  
                c = state[2], d = state[3],  
                x[APR_MD4_DIGESTSIZE];  
  
    ....  
    /* Zeroize sensitive information. */  
    memset(x, 0, sizeof(x));    // <=  
}
```



Примеры паттернов (Apache HTTP Server)

```
static void MD4Transform(  
    apr_uint32_t state[4],  
    const unsigned char block[64])  
{  
    apr_uint32_t a = state[0], b = state[1],  
                c = state[2], d = state[3],  
                x[APR_MD4_DIGESTSIZE];  
  
    ....  
    /* Zeroize sensitive information. */  
    memset_s(x, 0, sizeof(x));  
}
```

*Или используйте флаг `-fno-builtin-memset!`



Студентам

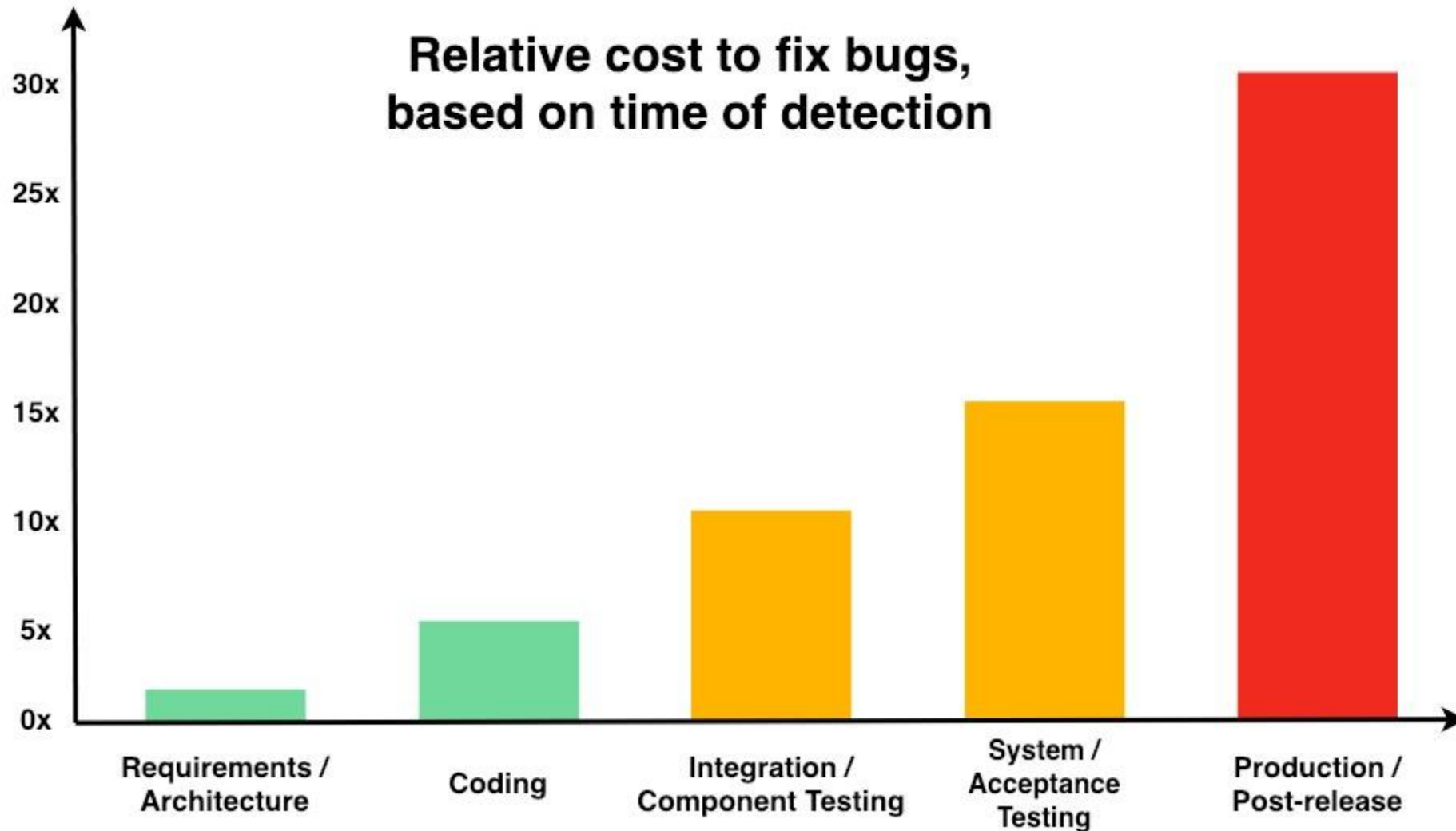
- Обучение развивающейся технологии
- Самостоятельная проработка проблем
- Облегчение процесса разработки
- Знакомство с паттернами ошибок

Использование в студенческих и открытых проектах

Главное – регулярность

- Максимальная польза от статического анализа достигается только при регулярном использовании!

Главное-регулярность



Подходящие статические анализаторы

- PVS-Studio
 - Clang Static Analyzer
 - Cppcheck
 - Infer
 - IntelliJ IDEA
 - FindBugs
 - ...
- Большой список статических анализаторов:

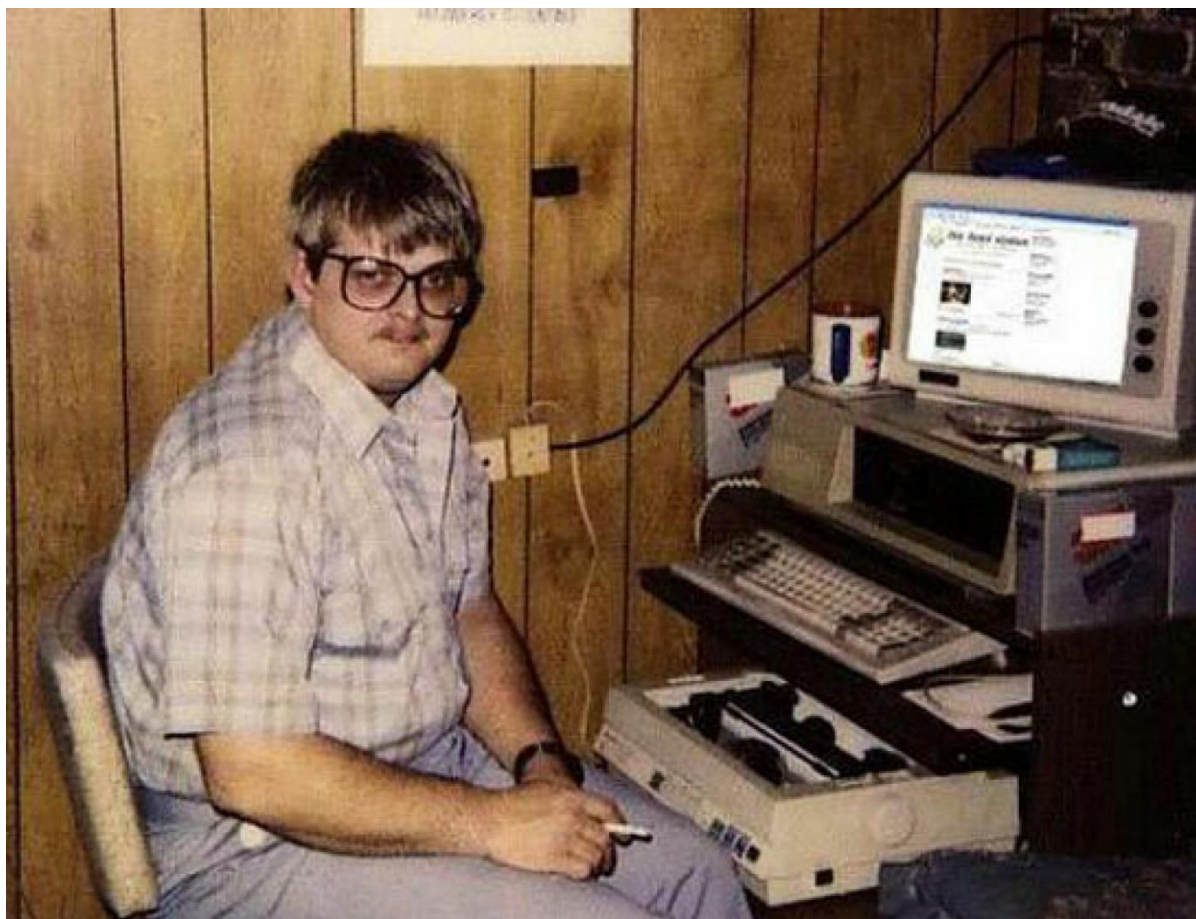


Внедрение анализа

1. Классический сценарий разработки
(в офисе)
2. Разработка студенческих и открытых проектов

Классический сценарий

- Локально на компьютерах разработчиков (плагины для IDE, системы мониторинга компиляции)

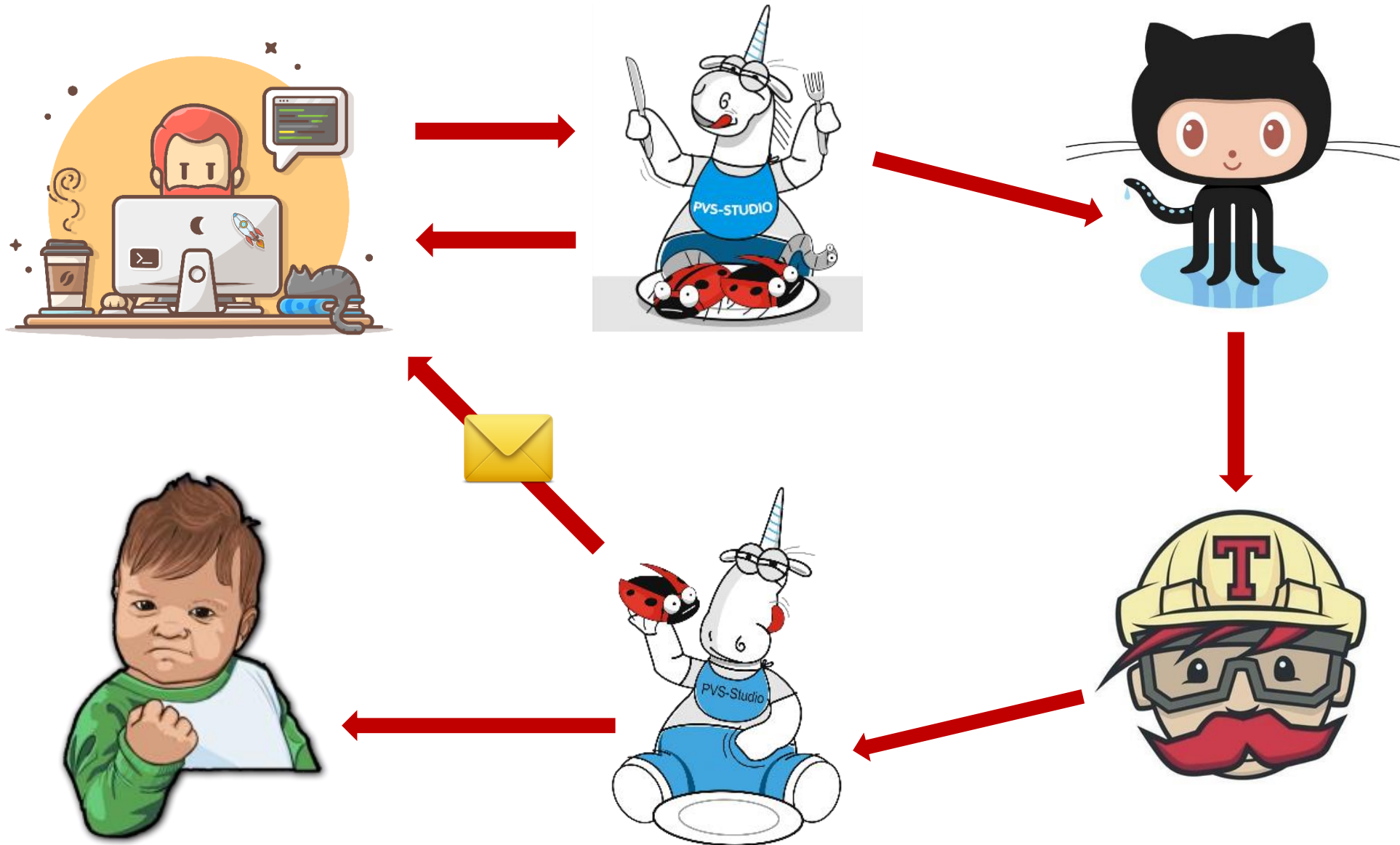


Классический сценарий

- В системах непрерывной интеграции (command-line утилиты, плагины для CI-систем, системы мониторинга)

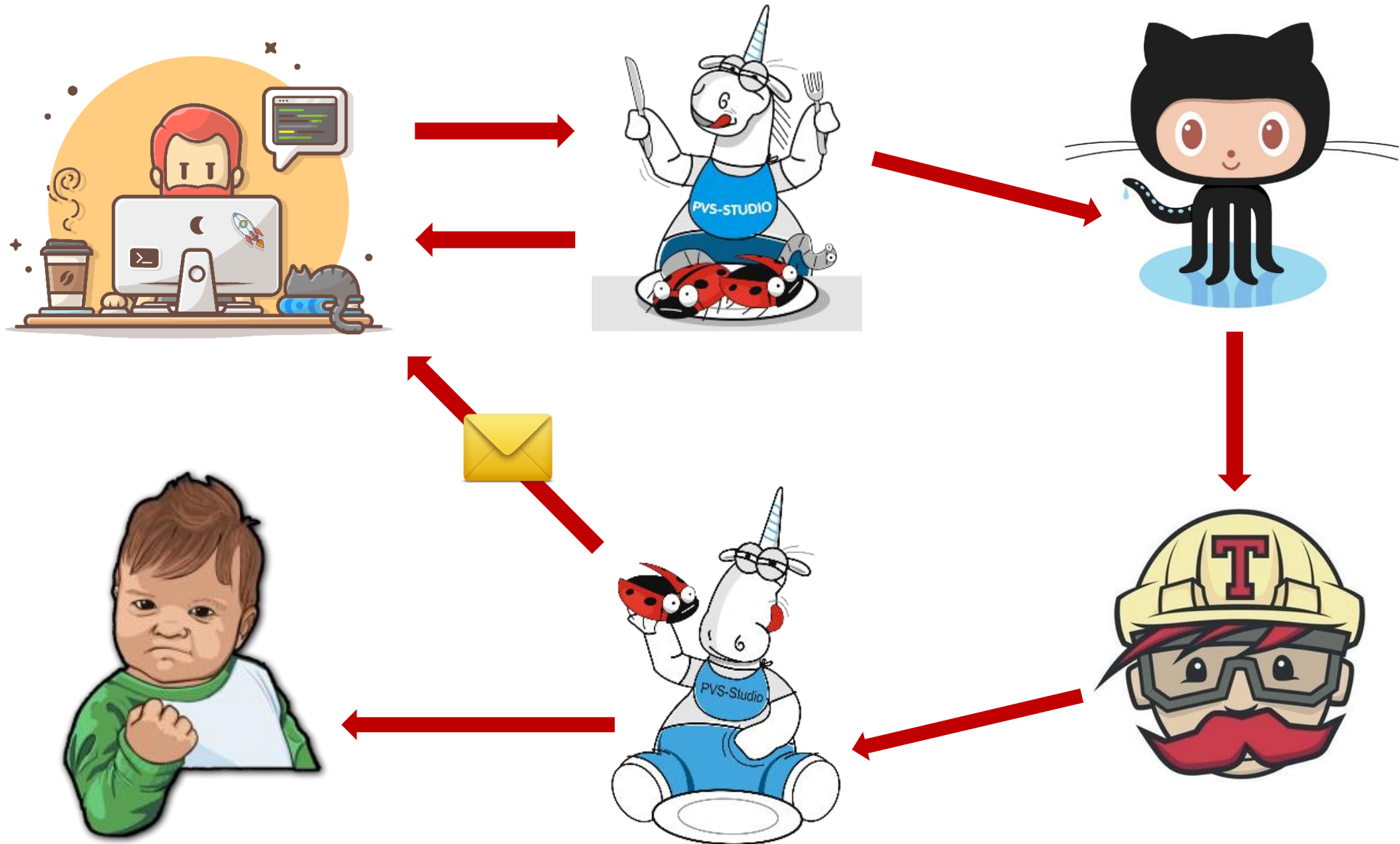


Классический сценарий



В ЧЕМ ОТЛИЧИЕ?

Классический сценарий



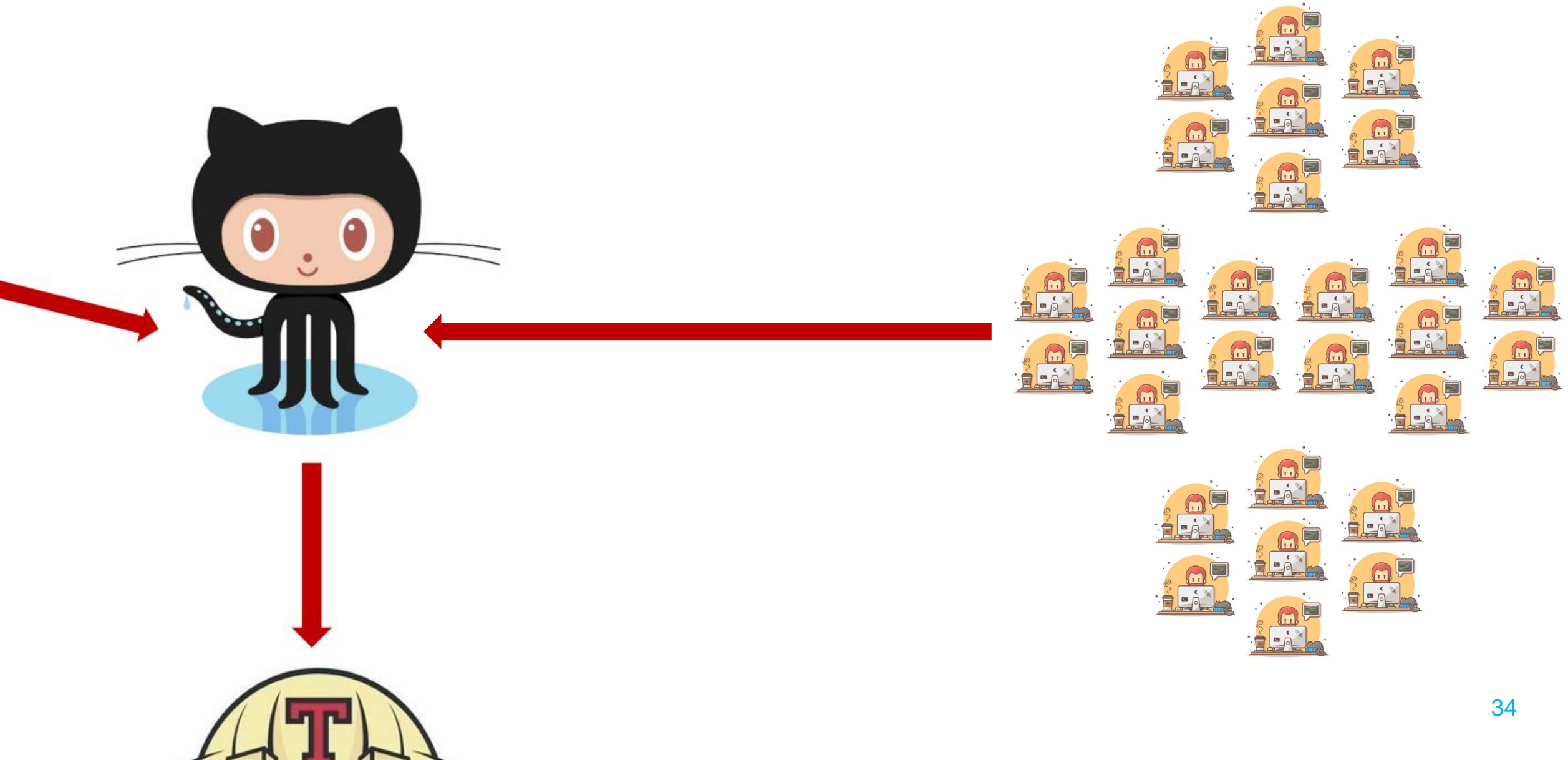
Студенческие и открытые проекты



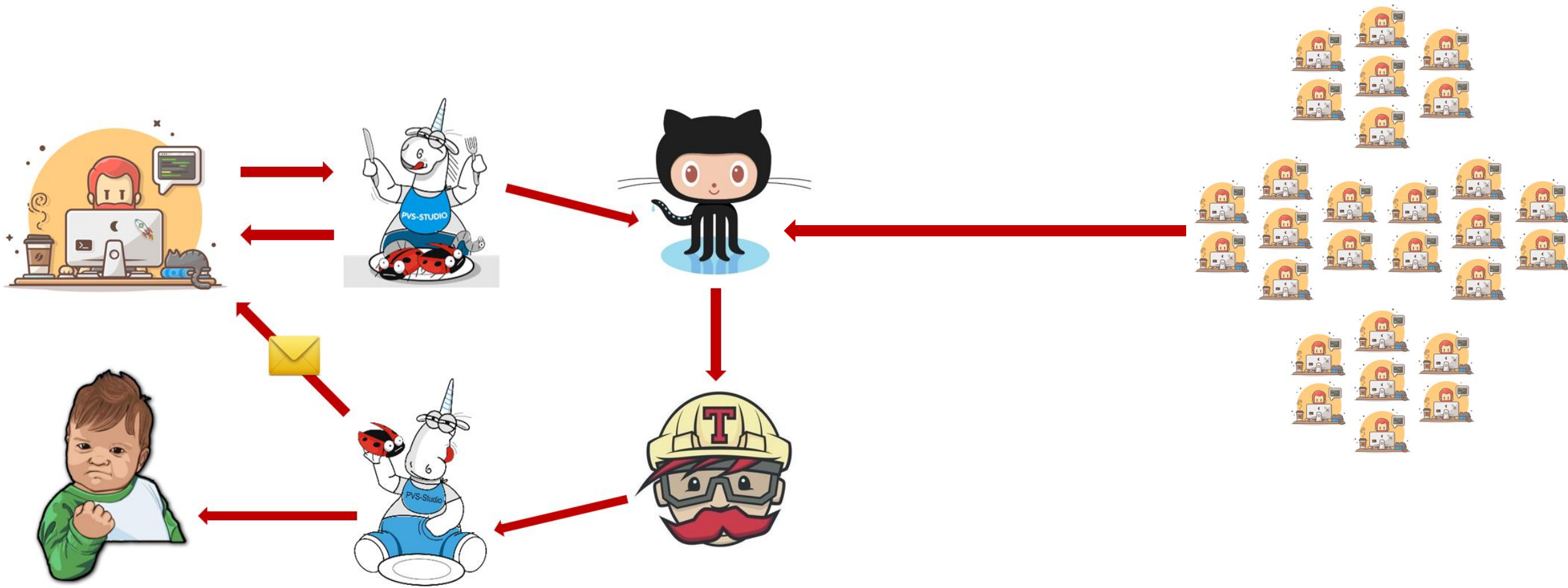
Студенческие и открытые проекты



Студенческие и открытые проекты



Студенческие и открытые проекты

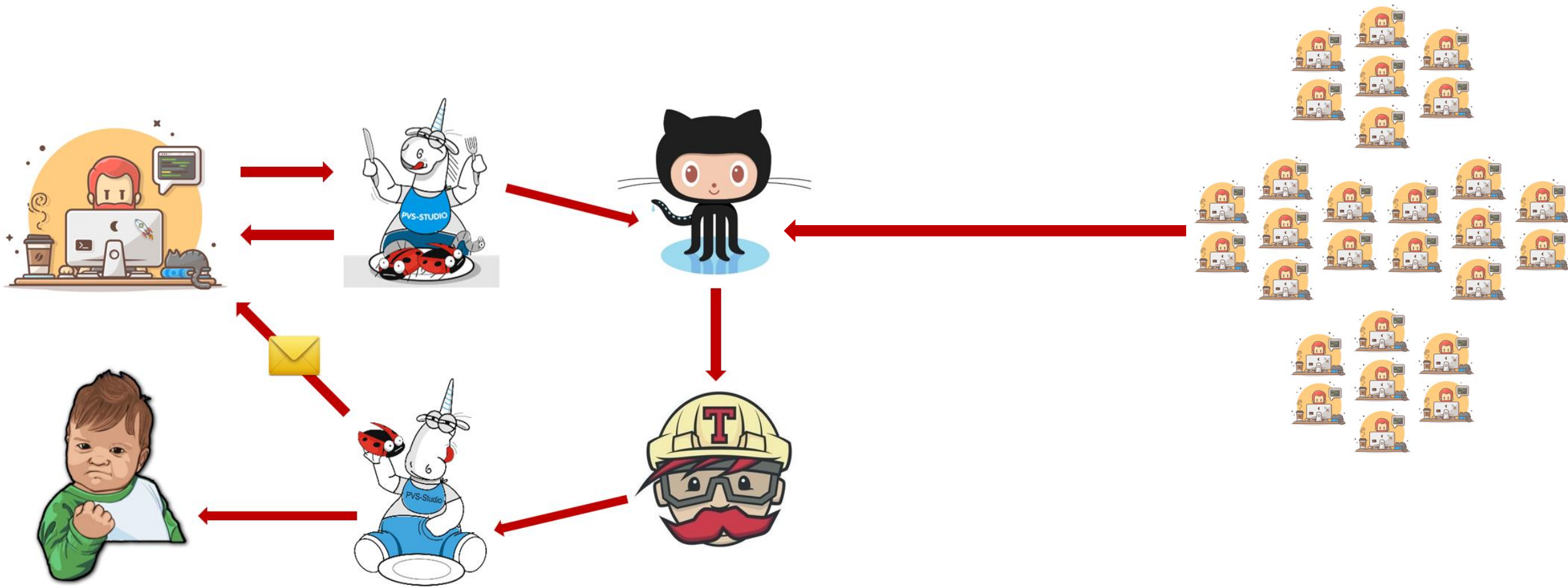


Внедрение анализатора в открытые проекты

Внедрение анализатора в открытые проекты



Как анализировать вклад сообщества?



Что делать после первой проверки?

Analyzer Output			
☰ Fails: 6 ▲ ▼ High: 1256 Medium: 3069 Low: 3250			
General Optimization 64-bit Custom MISRA FA:0 ▼			
★	Code	MISRA	Message
★	V730		Not all members of a class are initialized inside the constructor. Consider inspecting: L.
★	V611		The memory was allocated using 'new T[]' operator but was released using the 'delete' operator. Consider inspecting this code.
★	V512		A call of the 'memcpy' function will lead to underflow of the buffer 'ubuf'.
★	V547		Expression 'oslev.dwMajorVersion < 50' is always true.
★	V512		A call of the 'memset' function will lead to underflow of the buffer 'psDec->sLPC_Q14'.
★	V571		Recurring check. The 'if (limit)' condition was already verified in line 50.
★	V694		The condition (mode + 5) is only false if there is pointer overflow which is undefined behavior anyway.
★	V512		A call of the 'memset' function will lead to underflow of the buffer 'psDec->outBuf'.
★	V610		Unspecified behavior. Check the shift operator '>>'. The left operand is negative ('...' = [-2828544..0]).
★	V610		Unspecified behavior. Check the shift operator '>>'. The left operand is negative ('...' = [-2147483648..2147483647]).

Внедрение анализатора в открытые проекты



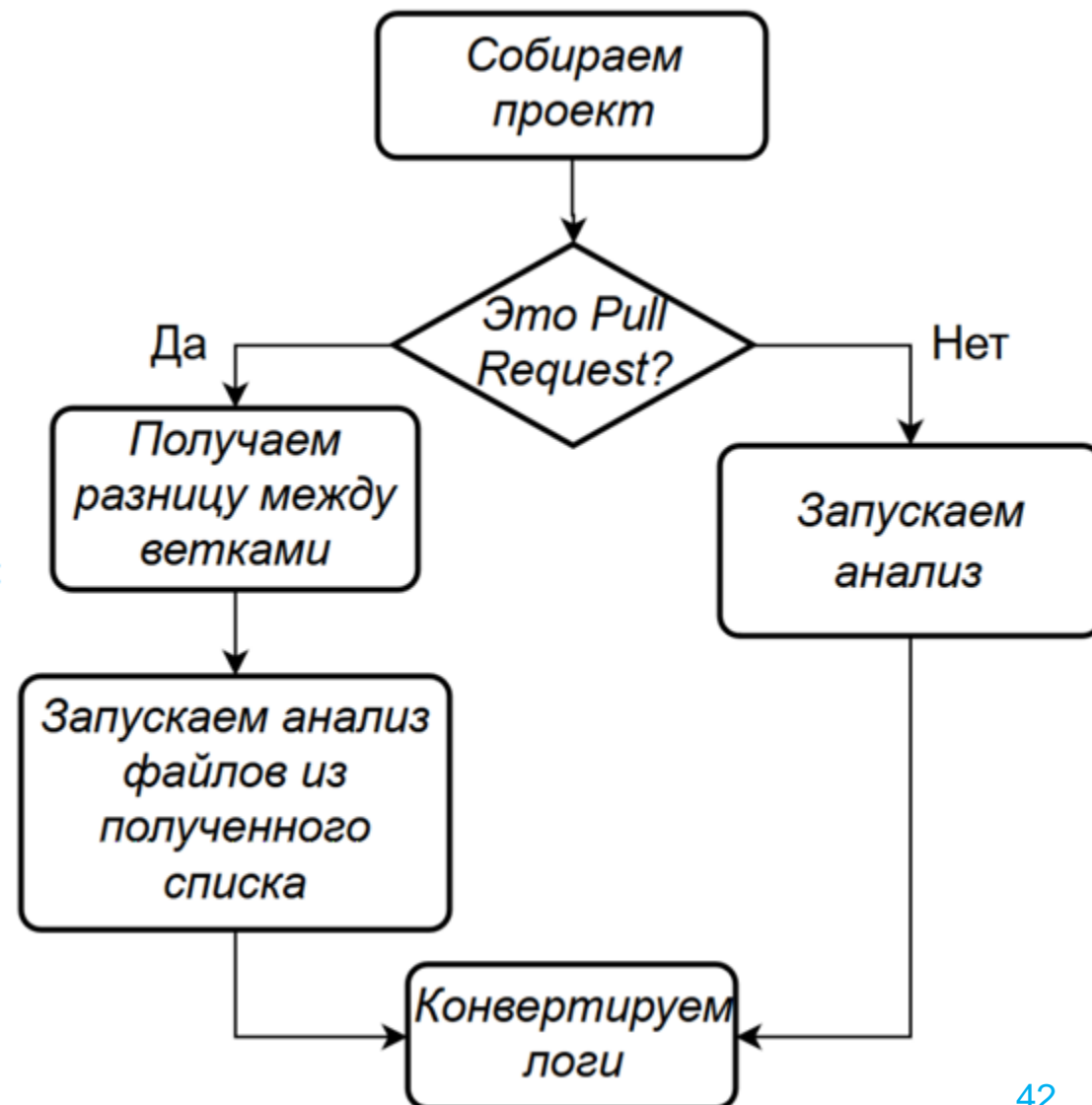
Внедрение анализатора в открытые проекты



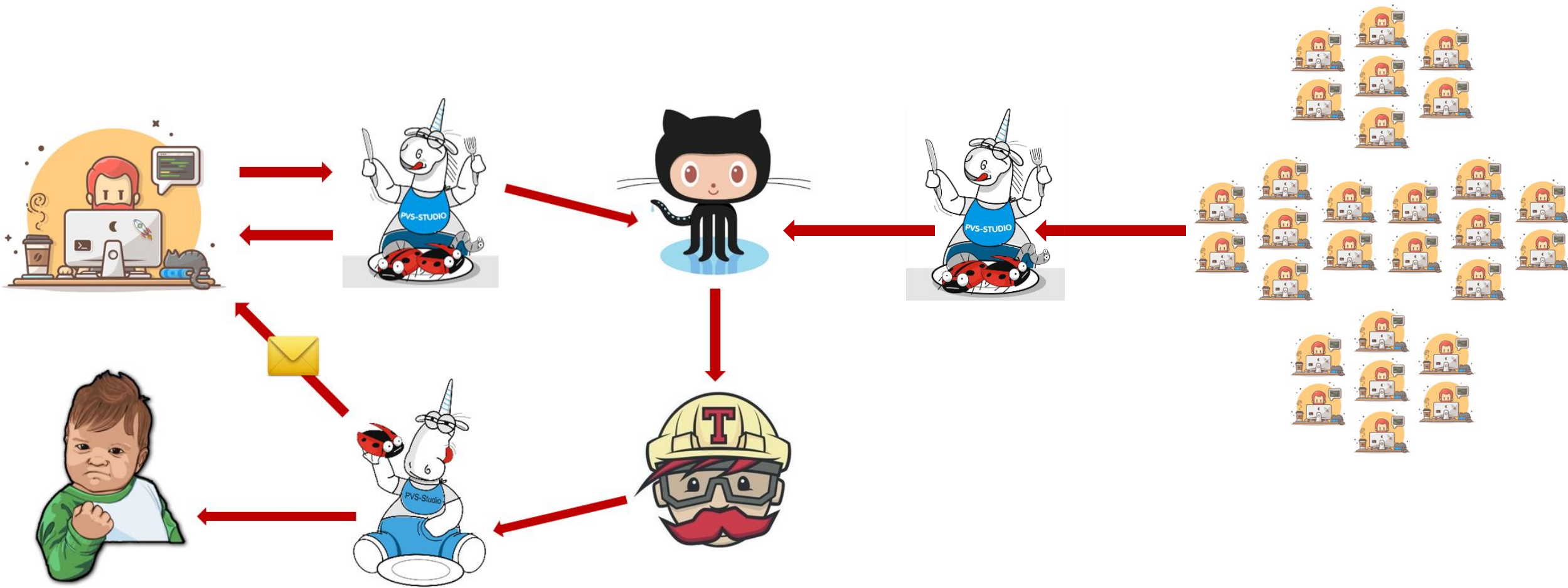
Анализ pull-request'ов

```
git diff --name-only origin/HEAD > .pvs-pr.list
```

```
pvs-studio-analyzer analyze -o PVS-Studio.log -S .pvs-pr.list
```

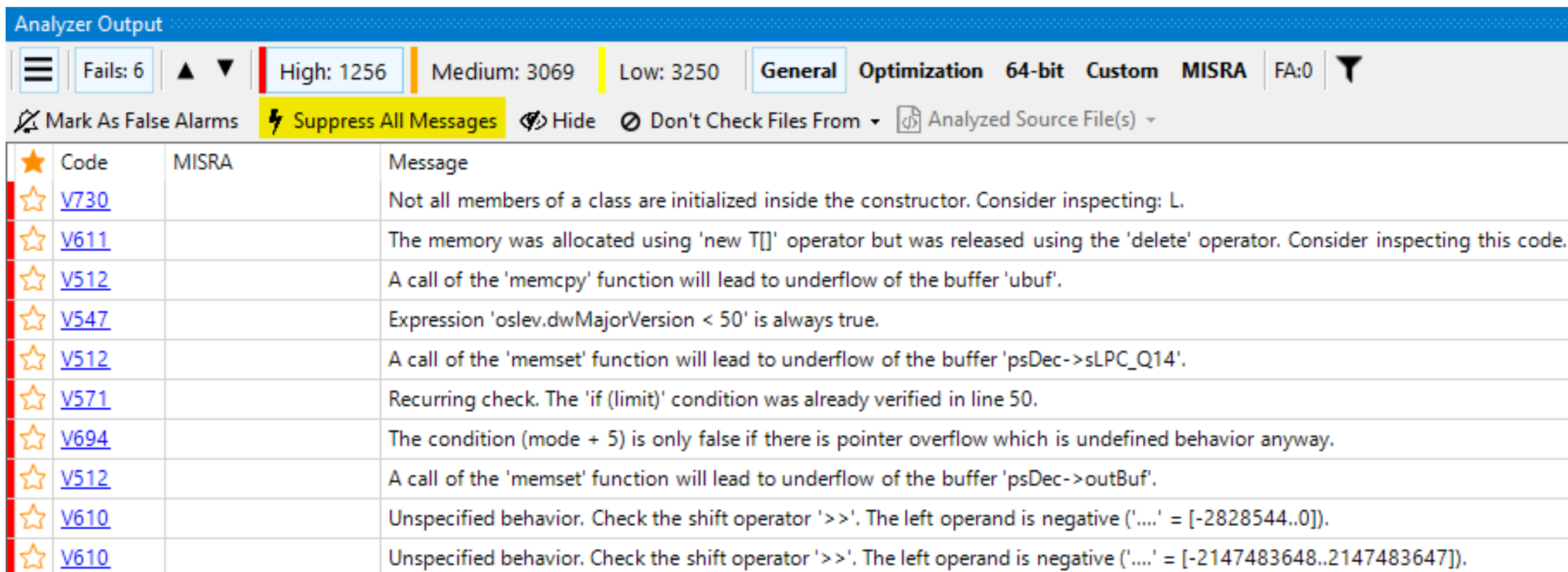


Как анализировать вклад сообщества?



После первой проверки

- Suppress-базы – это механизм массового подавления сообщений анализатора

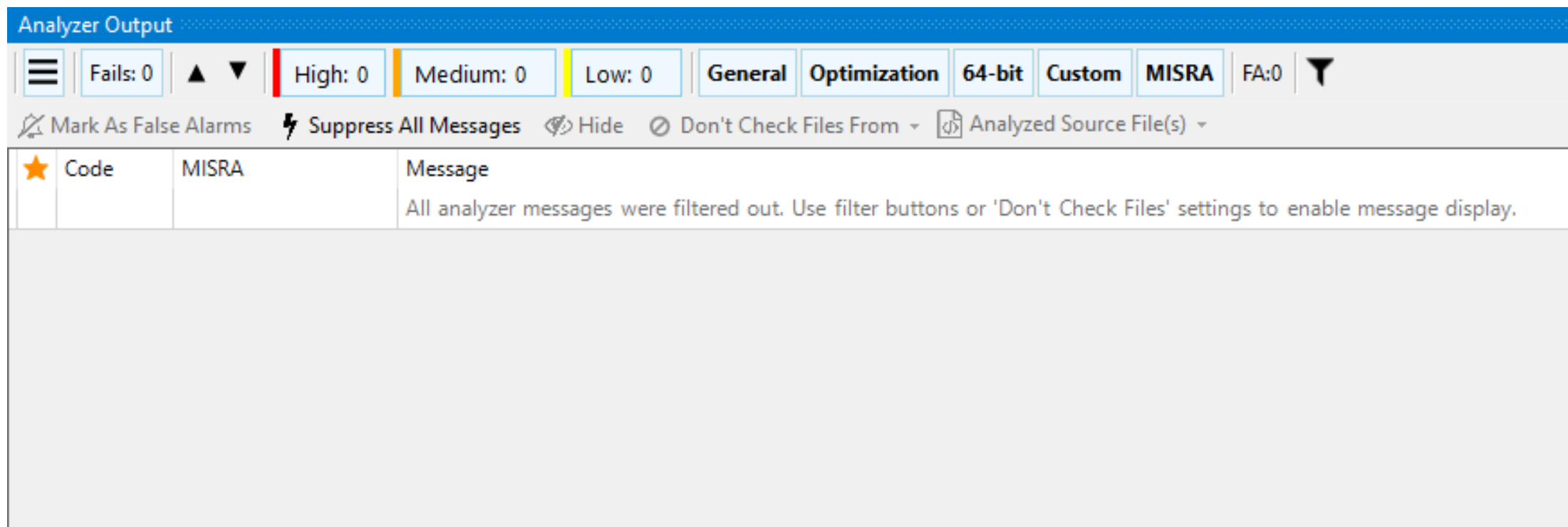


The screenshot shows the 'Analyzer Output' window in Visual Studio. The interface includes a toolbar with 'Fails: 6', severity filters for 'High: 1256', 'Medium: 3069', and 'Low: 3250', and tabs for 'General', 'Optimization', '64-bit', 'Custom', 'MISRA', and 'FA:0'. Below the toolbar, there are options to 'Mark As False Alarms', 'Suppress All Messages', 'Hide', and 'Don't Check Files From'. The main area displays a table of MISRA code violations.

★	Code	MISRA	Message
☆	V730		Not all members of a class are initialized inside the constructor. Consider inspecting: L.
☆	V611		The memory was allocated using 'new T[]' operator but was released using the 'delete' operator. Consider inspecting this code.
☆	V512		A call of the 'memcpy' function will lead to underflow of the buffer 'ubuf'.
☆	V547		Expression 'oslev.dwMajorVersion < 50' is always true.
☆	V512		A call of the 'memset' function will lead to underflow of the buffer 'psDec->sLPC_Q14'.
☆	V571		Recurring check. The 'if (limit)' condition was already verified in line 50.
☆	V694		The condition (mode + 5) is only false if there is pointer overflow which is undefined behavior anyway.
☆	V512		A call of the 'memset' function will lead to underflow of the buffer 'psDec->outBuf'.
☆	V610		Unspecified behavior. Check the shift operator '>>'. The left operand is negative ('...' = [-2828544..0]).
☆	V610		Unspecified behavior. Check the shift operator '>>'. The left operand is negative ('...' = [-2147483648..2147483647]).

После первой проверки

- Suppress-базы – это механизм массового подавления сообщений анализатора



The screenshot shows the 'Analyzer Output' window. At the top, there are several filter buttons: 'Fails: 0', 'High: 0', 'Medium: 0', 'Low: 0', 'General', 'Optimization', '64-bit', 'Custom', 'MISRA', and 'FA:0'. Below these are action buttons: 'Mark As False Alarms', 'Suppress All Messages', 'Hide', 'Don't Check Files From', and 'Analyzed Source File(s)'. The main area contains a table with the following content:

★	Code	MISRA	Message
			All analyzer messages were filtered out. Use filter buttons or 'Don't Check Files' settings to enable message display.

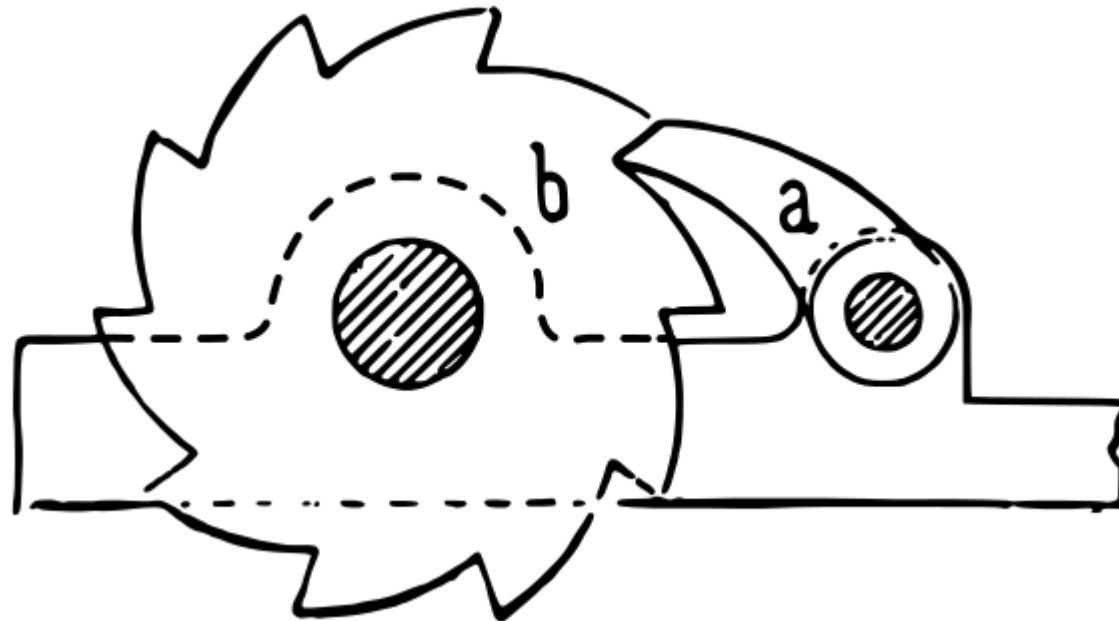
Зачем нужны suppress-базы

- Прячем старые ошибки – работаем в привычном темпе
- С этого момента видим только новые предупреждения
- Получаем пользу от анализатора СРАЗУ

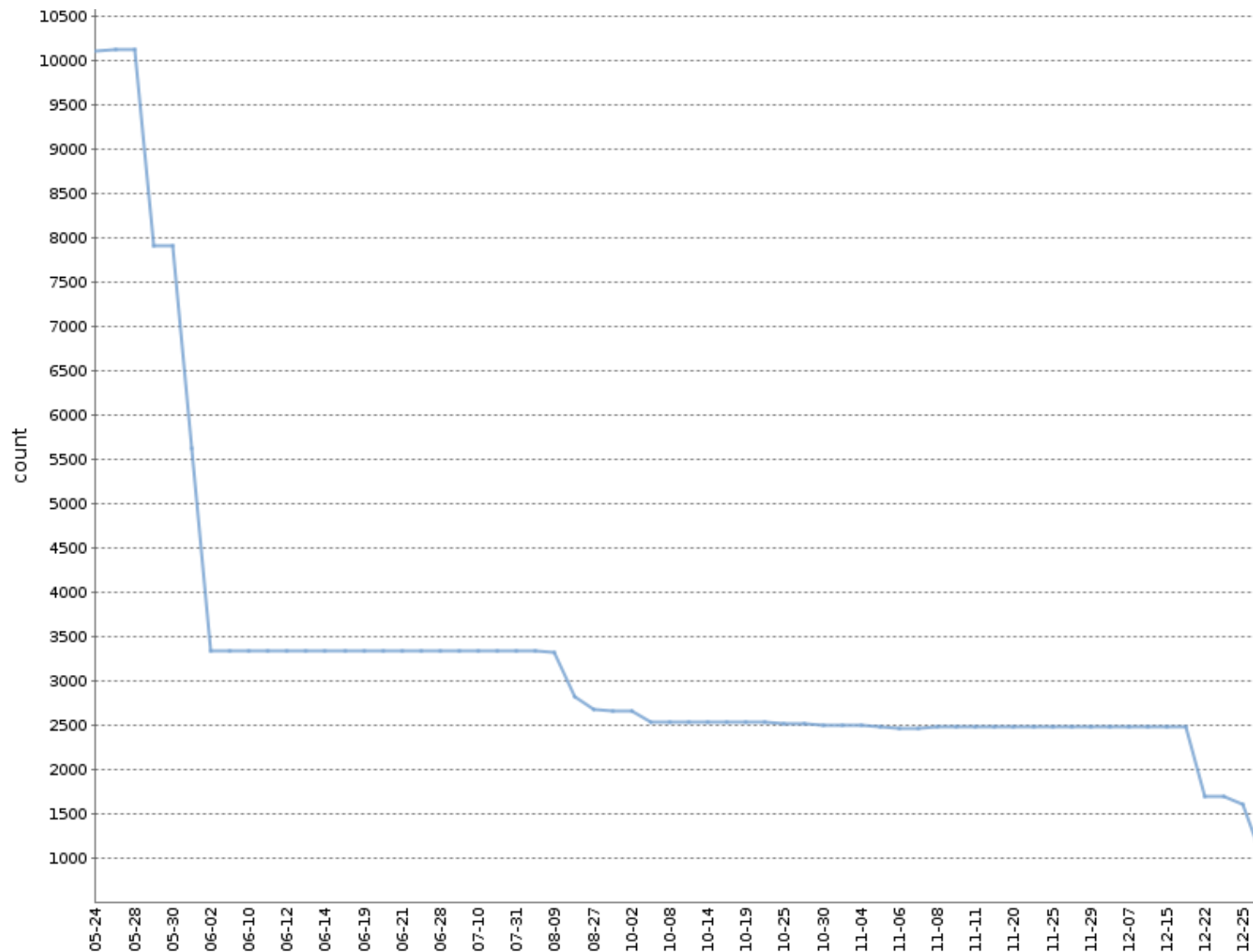
- Спрятанные ошибки не забываем! Возвращаемся и потихоньку правим.

Как работать с suppress-базой

- Очень удобный вариант: метод храповика («ratcheting»)
- Количество ошибок в базе закладывается в репозиторий
- Допускаются только те изменения, которые не увеличивают общее количество предупреждений



Как работать с suppress-базой



Полезный доклад по теме

- Иван Пономарёв — Непрерывный статический анализ кода



Иван Пономарёв
КУРС

Непрерывный
статический анализ
кода



Заключение

Выводы

- Статический анализ помогает обучаться программированию
- Важно использовать его регулярно
- Внедрять статический анализ в открытые проекты **можно!**

Бесплатная лицензия PVS-Studio для разработчиков открытых проектов



END

Q&A

