БУ ВО «Сургутский государственный университет»

Политехнический институт

Кафедра автоматизированных систем обработки информации и управления

ОТЧЕТ

ПО ЛАБОРАТОРНОЙ РАБОТЕ №4

ПО ТЕМЕ «УПРАВЛЕНИЕ КАЧЕСТВОМ. КЕЙС»

ПО ДИСЦИПЛИНЕ «МЕТРОЛОГИЯ, СТАНДАРТИЗАЦИЯ И СЕРТИФИКАЦИЯ»

Выполнил: студент группы №606-12,

Речук Дмитрий Максимович

Принял: ст. преподаватель кафедры АиКС,

Гребенюк Елена Владимировна

Сургут 2024

**СОДЕРЖАНИЕ**

[ВВЕДЕНИЕ 3](#_Toc183279598)

[ОПИСАНИЕ КЕЙСА 4](#_Toc183279599)

[АНАЛИЗ ПРЕДЛОЖЕННЫХ МЕРОПРИЯТИЙ 7](#_Toc183279600)

[СОБСТВЕННЫЕ РЕШЕНИЯ 8](#_Toc183279601)

[ЗАКЛЮЧЕНИЕ 9](#_Toc183279602)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 10](#_Toc183279603)

# ВВЕДЕНИЕ

Цель: подобрать оптимальные методы управления качеством для данного кейса.

Задачи:

* Рассмотреть представленную ситуацию и разработанные мероприятия по устранению проблем в разработке.
* Проанализировать возможности на предмет реальности проведения всех намеченных мероприятий.
* Разработать свои варианты решения проблем и обосновать оптимальный вариант.
* Ответить на вопросы.

# ОПИСАНИЕ КЕЙСА

Борис работает ведущим IT-менеджером компании по разработке программных продуктов.

Вот его рассказ: «Когда-то я был тестировщиком и в те далекие времена порой был крайне недоволен программистами: вечные сомнительные доводы: «Это не баг, это фича» или «Если это и баг, то незначительный, пусть остается!».

Да как же остается, если система колом встает!? Потом я стал программистом. И всё изменилось – меня начали жутко бесить бесконечные возвраты на доработку: то им это не нравится, то тут не работает! Да зачем было вообще в этом окне контекстное меню вызывать и вставлять нечитабельные символы!? Как они вообще до этого додумались!? Бред же, в боевом режиме так ни один пользователь не сделает! Не буду править, пусть остается!

В общем, классика – вражда программистов и тестировщиков. А потом я стал IT-менеджером. И понял, что вражда эта губительна для общего дела. К счастью, я хорошо помнил себя и программистом и тестировщиком, что и помогло мне осознать истинную суть проблемы и изменить ситуацию».

Пять лет назад в компании был очень простой и понятный процесс разработки:

Задачи --> Программирование <--> Тестирование --> Релиз (результат).

Причем: тестировщики узнавали о задачах только в момент передачи их в тестирование, т.е. о начале разработки новой задачи они уведомлений не получали; программисты не дожидались завершения тестирования задач и приступали к новым задачам немедленно.

Все считали подобную ситуацию примером идеальной инкапсуляции!

У программистов на вход: новые задачи и возвраты из тестирования, на выход: передача задач в тестирование.

У тестировщиков: на вход - задачи от программистов, на выход - возврат задач программистам и официальный выпуск версии.

Собственно, процесс хорош – каждый живет в своем мире и занимается любимым делом!

Но ведь у этого процесса есть вполне конкретная конечная цель – выпуск качественного ПО с нужным функционалом в срок.

В этот момент и начинаются проблемы: приходит менеджер и начинает напоминать о конечной цели.

Типичная ситуация: Менеджер приходит к программистам и спрашивает «когда?», а они отвечают «не знаем, мы все сделали, спросите у тестировщиков». Менеджер идет к тестировщикам с тем же вопросом «когда?», а они ему отвечают: «Разработчики только утром выдали сборку, мы только сейчас приступили к тестированию, и точно будет возврат – тут багов уже много вылезло, поэтому мы не знаем когда будет выпуск – спросите у программистов».

И менеджер начинает ходить по кругу туда-сюда, а релиза все нет и нет.

В итоге, терпение менеджера кончается, он собирает программистов и тестировщиков вместе и пытается как-то решить вопрос. Собственно, все решения сводятся к выработке правил взаимодействия между программистами и тестировщиками.

И все дальнейшие усилия менеджера направлены на контроль за соблюдением этих правил.

Вот некоторые правила, которые сформировались в отделе спустя несколько месяцев изнурительной работы менеджера:

* Совместное планирование. Версия планируется заранее. На планировании присутствуют и программисты и тестировщики. Благодаря этому, все заранее знают, что им предстоит делать. Например, это позволяет тестировщикам заранее начать составлять планы тестирования и подготавливать необходимое тестовое окружение.
* Маленькие версии. Это попытка сократить разработку, ведь это накладные расходы на актуализацию кода и повторное тестирование. Если делать маленькие версии, то можно работать сразу в основной ветке.
* Ничегонеделанье. Ещё одна попытка сократить разработку. Когда накапливается много задач, и разработка сильно убегает вперед от тестирования, то программистам разрешается просто ничего не делать, чтобы ещё больше не убегать вперед.
* Раннее информирование. Например, тестировщик приступил к тестированию задачи. Задача большая, но он уже сразу нашел дефект. Он сообщает об этом программисту сразу при обнаружении, а не в конце, когда все тестирование завершено. Или, наоборот, программист ещё во время разработки сообщает тестировщику о нюансах реализации, чтобы тот заранее подготовил тест-план. Это позволяет развести работу программиста и тестировщика.

Все эти правила на самом деле хоть и помогли улучшить ситуацию, но кардинально её не изменили. Они как будто затыкали по одной маленькой дырочке. Чувствовалось, что есть какое-то другое решение, что что-то упущено из виду. Например, как только менеджер расслаблялся – почти сразу все договоренности между программистами и тестировщиками забывались, и все возвращалось к исходному состоянию.

Более того, взаимоотношения между программистами и тестировщиками не улучшались – как была вражда, так она и осталась.

Борис, как ведущий IT-менеджер, много размышлял о причинах происходящего, о поведении людей, об их эмоциях, потребностях и мотивах. А потом вдруг всё стало ясно! Да ведь сама структура такого подхода, когда «одни программируют – другие тестируют» порождает конфликт между программистами и тестировщиками. И вся суть этого конфликта в том, что у них разные цели!

У тестировщиков цель «протестировать». У разработчиков цель «разработать», т.е. «передать в тестирование». А цель «выпустить релиз» только у менеджера. И только пока он прикладывает к этому усилия – она достигается. А когда у людей разные цели – им не по пути, они не интересны друг другу. Как их не притягивай, они все равно будут идти своей дорогой, в разные стороны.

Собственно, решение проблемы и заключается в том, чтобы поставить и перед программистами, и перед тестировщиками одну общую цель. Причем цель очевидную – выпуск качественного программного продукта в срок. Ведь не достигая эту цель, локальные цели «разработать» и «протестировать» теряют всякий смысл. Изменить цели оказалось делом не простым. Но поскольку Борис полностью проникся логикой своих мыслей – то был готов сломить любое сопротивление изменениям!

Вот основное, что было сделано:

1. Изменена организационная структура отдела. Вместо отделов разработки и тестирования сформированы команды, в которых собраны и программисты и тестировщики.
2. Переезд подразделений. Вновь сформированные команды получили по отдельной комнате. Теперь не было ситуации, когда тестировщики и программисты сидят в отдельных комнатах и живут своей жизнью.
3. Пропаганда. Пришлось сказать немало пламенных речей о том, для чего и почему затеяли реорганизацию. И главное – донести до всех общую цель.
4. Увольнение. Увы, но кто-то не согласился с новыми идеями. Они уступили дорогу тем, кто теперь приносит куда больше пользы.

И все эти усилия того стоили! Эффект оказался просто поразительным! Напряженные отношения между программистами и тестировщиками исчезли, как будто и не было.

Положительные последствия этих решений:

* Появилась взаимная поддержка, продукты стали качественнее.
* Программисты стали помогать тестировщикам, указывая на узкие места, которые стоит дополнительно проверить.
* Изменилось общее отношения к обнаруживаемым дефектам. Никто никого не считает виноватым. Даже наоборот, разработчик благодарен тестировщику, что тот помог сделать продукт лучше.
* Все договоренности о взаимодействии программистов и тестировщиков стали выполняться сами собой – потому что все поняли их эффективность.
* В общем, все заработало как часы – регулярные релизы, качественный продукт. На глазах за полгода произошло настоящее преображение!

# АНАЛИЗ ПРЕДЛОЖЕННЫХ МЕРОПРИЯТИЙ

1. Изменение организационной структуры отдела. На реализацию такого решения потребуется время, так как появятся совершенно новые команды, состоящие из сотрудников разных должностей, а значит придется выработать новые правила и алгоритмы работы, к тому же не стоит забывать, что в командах существуют межличностные отношения, на формирование которых уходит определенное количество времени.
2. Переезд подразделений. Исходя из средних заработных плат программистов и тестировщиков можно сказать, что IT компании обладают достаточным количеством средств на аренду помещений и их обустройство. Такое мероприятие организовать несложно.
3. Пропаганда. В данном случае пропаганда состоит из двух частей: поднять мотивацию и донести общую цель. С первым могут возникнуть проблемы, так как даже самые яркие речи не будут гарантировать, что сотрудник проникнется духом компании или чем-то еще, а второе вполне реально и легко осуществимо, так как цель имеет свои объективные последствия.
4. Увольнение. Такое мероприятие провести несложно, но поиск новых сотрудников может стать проблемой. Спрос на программистов на рынке существует долгое время, значит проблемы с наймом есть у многих компаний и в борьбе за кадры выигрывают компании с достаточным бюджетом.

# СОБСТВЕННЫЕ РЕШЕНИЯ

1. Премии. Вознаграждение за высокую производительность может стимулировать сотрудников или целые отделы.
2. Повышение квалификации. Оплата курсов тимбилдинга для сотрудников компании. Эти знания могут улучшить внутреннюю коммуникацию в новых отделах.
3. Оптимизация бизнес-процессов. Внедрение методов для устранения неэффективности в работе отделов.
4. Изменение планирования. Переназначение дат сдачи промежуточных результатов и конечного результата для более комфортной работы сотрудников, но могут быть негативные последствия в виде финансовых потерь.

# ЗАКЛЮЧЕНИЕ

В результате была достигнута цель, выполнены поставленные задачи.

После анализа предложенных решений можно сказать, что все предложенные в кейсе решения при правильной реализации имеют свои положительные эффекты.

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Сургу. Moodle. «Лаб№4. Библиотека 15.03.2024».