Управление конфликтами

Описание

Конфликт – наиболее острый способ разрешения противоречий в интересах, целях и взглядах, возникающий в процессе социального взаимодействия. Соответственно, управление ими решает задачу минимизации потерь и не повторения конфликтов в будущем.

Почему ветка важна?

Появление конфликтов неизбежно в силу человеческой природы, но важно научиться их разрешать и обеспечивать условия для неповторения.

В некоторых моделях наличие конфликтов является необходимым этапом на пути развития команды. Этот этап называют штормингом. Автор модели исходит из того, что шторминг является следствием чрезмерных ожиданий друг от друга. Когда же команда определяет нормы, роли и ожидания друг от друга, команда нормализуется.

Важным качеством лида также является принятие возможности появления конфликтов. В обратном случае боязнь конфликтов может парализовать работу.

Что будет, если её не делать?

Избегание конфликтов приводит к тому, что теряется возможность контролировать исполнение. Тут важно отличать «идти на конфликт» и «принимать возможность его появления». Так, первое является неконструктивным поведением, когда второе необходимое умение.

Неумение разрешать конфликты приводит к следующим негативным сценариям:

- Избеганию конфликтов на дистанции.
- Выбору вариантов невыгодных для себя, в то время как конечной целью является снижение издержек и поиск компромиссного варианта.

На кого может быть делегирована?

Эти навыки являются неотъемлемыми для эффективной работы с людьми. Поэтому делегирование в рамках профессии тимлида не возможно.

Примеры поведения

Примеры плохого поведения

- Избегание возможных конфликтов.
- Неконструктивные эмоции: от гнева до страха.

Примеры хорошего поведения

- Перевод конфликта из эмоциональной плоскости в плоскость аргументов и ожиданий.

Способы прокачки

Практика

Ключ к разрешению конфликта – переход от эмоций к явному обозначению желаемого решения проблемы каждой из сторон. Хорошим приёмом является на эмоциональный выпад ответить что-то в духе: «я услышал, что ты чем-то недоволен, можешь рассказать как бы ты хотел, чтобы разрешилась ситуация?».

В некоторых случаях внезапно оказывается, что сами решения не противоречат друг другу, а конфликт появился из-за недопонимания, или что людям не нравятся незначительные детали.

Если конфликт в противоречии интересов, то необходимо сделать явным, откуда у каждой из сторон такие ожидания. Далее можно переходить к торгам. Даже если вы не сможете договориться, спор будет зафиксирован на уровне аргументов и взвешивания «за и против».

Отдельный момент – эмоциональная реакция на конфликты. Она в полной мере не контролируется сознательной частью мозга. Из практик можно применить десенсибилизацию, работу с психотерапевтом.

Консультации

- Чат TeamLead Bootcamp

Теория

Видео

- Конструктивная конфронтация: как решать конфликты по схемам

Сотрудничество

Описание

Понимание ценности и умение выстраивать совместную работу для достижения взаимовыгодного результата.

Почему ветка важна?

Взаимовыгодное сотрудничество – основа любых договорённостей, от постановки текущих задач до обсуждения зарплаты или полугодовых целей.

Здоровые отношения между двумя людьми дают:

- Понятные обязательства с обеих сторон.
- Мотивацию закрывать чужие ожидания, для получения закрытия собственных.

Что будет, если её не делать?

Нежелание сотрудничать обычно приводит к двум сценариям:

- Ожидания превращаются в требования и опускаются на человека сверху. Что часто приводит или к демотивации, или к невыполненным обязательствам.
- Отсутствие или неявные ожидания людей друг от друга. Первый не понимает что от него ждёт второй, а второй в свою очередь недоволен, что не закрыты «очевидные» ожидания.

В других видах отношений, не обязательно трудовых, нежелание договариваться и сотрудничать обычно приводит к конфликтам.

На кого может быть делегирована?

Эти навыки являются неотъемлемыми для эффективной работы с людьми. Поэтому делегирование в рамках профессии тимлида не возможно.

Примеры поведения

Примеры плохого поведения

- Человек не считает важными и не слышит чужие ожидания.
- Учитываются только чужие потребности.

Примеры хорошего поведения

- Конечной целью договорённостей становится нахождение взаимовыгодного варианта.
- Уважение чужих ожиданий на равне с собственными.

Способы прокачки

Практика

1. Выслушайте ожидания человека, с которым хотите договориться.
2. Расскажите свою мотивацию и своё идеальное виденье разрешения ситуации. Во многих случаях оказывается, что конечные цели не противоречили друг другу, находится оптимальное решение для обоих сторон.
3. В случае, если цели противоречат друг другу: имеет смысл перейти от желаний к понятию справедливости. Тут может помочь честное объяснение причин собственных ожиданий, аналогичные примеры.

Консультации

- Чат TeamLead Bootcamp

Теория

Книги

- Семь навыков высокоэффективных людей. Мощные инструменты развития личности (глава 4), Стивен Кови

Фасилитация (Управление групповой дискуссией)

Описание

Фасилитация (от англ. «facilitate» – облегчать) – это специальные действия, направленные на организацию групповой работы.

Фасилитатор – человек, организующий работу группы таким образом, чтобы она достигла стоящие перед ней цели. Фасилитатор в идеальном случае не имеет мнения по обсуждаемому вопросу и/или не озвучивает его, чтобы не оказать влияния на мнение группы.

Ключевой особенностью данной техники является уход от диалога "двое разговаривают, остальные слушают" к работам в малых группах, схождение и расхождение от/к общей дискуссии. Комбинация разных техник, форматов участия, а также их ограниченность в времени позволяют участникам не "залипать", а наоборот быть в тонусе и работать продуктивно над решением вопроса всю сессию.

Почему ветка важна?

Владение искусством фасилитации позволяет помочь большой группе людей прийти к единому решению или решениям. Такие решения отражают мнения всей группы, а не самых активных участников. Позволяет значительно экономить время, даже при рассмотрении сложных вопросов, таких как "архитектура предприятия" или "выбор нового JS-фреймворка".

Что будет, если её не делать?

Встречи начинают занимать значительное время команды, но как правило, не имеют конечного результат, каждый участник остаётся при своём мнении.

На кого может быть делегирована?

Scrum-мастер, внешний фасилитатор, тимлид ниже уровнем, разработчик.

Способы прокачки

- Тренинги (например от Дмитрия Лазарева).
- Практика работы с группами и отработка обратной связи.

Практика

Функции фасилитатора

- **Побуждать к полноценному участию**, т.е. давать сделать полноценный вклад в достижение цели всем участникам.
- **Способствовать взаимопониманию**, чтобы участники могли понять даже противоположные точки зрения.
- **Стимулировать принятие взаимоприемлемых решений**, чтобы учитывать мнения всех и чаще получать ситуацию win-win.
- **Культивировать чувство общей ответственности**, чтобы принятые решения могли быть воплощены в жизнь, а не остались лишь в рамках сессии.

Процесс

- Определение целей сессии с заказчиком.
- Создание фасилитационного дизайна (план, тайминг, участники, ограничения, подготовительные материалы, ...).
- Предварительная работа с участниками до сессии (5 мин 1-1 по ожиданиям, валидация плана, сбор информации, ...).
- Проведение сессии.
- Ретроспектива сессии.
- Оповещение участников о результатах и принятых решениях (распространение материалов, фото и видеозаписей, слайдов, ...).

Инструменты и техники

- **Активное слушание** - создание комфортной атмосферы с одним или несколькими собеседниками, чтобы дать им возможность высказать своё мнение.
- **Чартрайтинг** - фиксация предложений от группы, например во время мозгового штурма.
- **Бизнес-визуализация** - простые рисунки, пиктограммы и надписи на флипчарте, которые позволяют лучше понять выступающего, не стоит путать со скрайбингом.
- **Открытая дискуссия** - самый тяжёлый вид групповой работы, диалог 2/3-человек при молчании всей группы, которая начинает "залипать" после нескольких минут.
- **Альтернативы открытой дискуссии** - форматы участия, чтобы уйти от групповой дискуссии и упростить групповую динамику
- **Мозговой штурм** - классический мозговой штурм.
- **Управление длинными списками** - классификация, приоритизация, голосование, выстраивание топ X.

- **Сложная динамика группы** - разбор сложных ситуаций, таких как "тролли", "затягивание времени", неуместное обсуждение деталей, ...
- **Шкала принятия решений** - шкала, например от 1 до 7, где 1 - категорически не согласен, а 7 - полностью поддерживаю, которая позволяет по рисунку голосования понять "теплоту" принятия конкретного решения.
- ...

Альтернативы открытой дискуссии

- **Индивидуальные задания** - что-то описать или подумать самостоятельно.
- **Круговой опрос** - опрос участников по очереди, давая возможность ответить на один вопрос. Можно использовать токен, например мягкую игрушку или куш бол.
- **Работы в малых группах** - задачи для малых групп до 5 человек, чтобы сделать обсуждение общего вопроса параллельно.
- **2-4-8-все** (пример малых групп) - обсуждение общего вопроса в двойках, далее слияние в четвёрки, потом в восьмёрки, далее на всю группу. Позволяет убрать одинаковые предложения и сформировать уникальные список.
- **Многозадачные группы** - подгруппы обсуждающие разные аспекты общей темы.
- **Аквариум** - обсуждение острого вопроса, когда, например есть два противоположных мнения. Одна группа обсуждает, вторая группа слушает без права эфира, потом они меняются.
- **Шифратор** - может комбинироваться с ролевыми сценариями, когда роли одной группы после 1 раунда переходят в другие группы для получения нового опыта.
- **Ролевые сценарии** - отработка практик по ролям.
- **Доклады, презентации, питчи** - представление результат работы, например после работы в мини группах.
- **"Трейд-шоу", "Мировое кафе"** - одновременная презентация работы нескольких групп, участники перемещаются между станциями и имеют возможность услышать решения всех подгрупп.
- ...

Литература

- Facilitator's Guide to Participatory Decision-Making, Sam Kaner, 2014
- Руководство фасилитатора. Как привести группу к принятию совместного решения, Сэм Кейнер, 2016 г.

Консультации

- Чат TeamLead Bootcamp

Дача и получение обратной связи

Описание

Навык получения обратной связи от других людей, а также умение её давать. Имеет смысл разделить кейсы по направлению обратной связи. Цели получения обратной связи:

- Получить дополнительную информацию для рефлексии и развития.
- Узнать об ожиданиях и потребностях других людей от нас.

Цели дачи обратной связи аналогичны, но в обратную сторону:

- Поблагодарить человека.
- Высказать собственные ожидания от человека.
- Обратить внимание на факты, дать информацию к размышлению.

С обратной связью на работе мы встречаемся в случаях: code review, one-to-one, всевозможные опросы, например 360.

Почему ветка важна?

Обратная связь один из немногих инструментов, который позволяет доносить собственные ожидания людям и помогать формировать точки роста. Обратная связь от других людей в свою очередь даёт материал для рефлексии.

В команде умение лида давать обратную связь помогает показывать проблемы и точки роста коллегам. А развитие культуры дачи обратной связи в команде даёт членам команды лучше расти и избегать конфликтов даже без участия лида.

Что будет, если её не делать?

Обратную связь давать и получать сложнее чем кажется из-за того, что легко пересечь грань перехода на личности и начать неконструктивный диалог.

Некоторые явления

- Обратная связь не даётся.
  - У лида или членов команды есть невысказанные незакрытые ожидания. Это приводит к обидам и конфликтам.

- На качество дачи обратной связи не обращают внимание.
  - Это может приводить к культуре токсичности внутри команды, где остаётся мнение людей, готовых к переходу на личности. Люди считают, что помогают развиваться.

На кого может быть делегирована?

В идеальном случае как позитивную, так и негативную обратную связь даёт сам человек, так как он максимально погружён в конкретный кейс.

В особенно сложных случаях дачи негативной обратной связи, можно прибегнуть к третьей стороне, но это не рекомендуется, так как возможен эффект «сломанного телефона», а в случае начала конфликта этот человек будет также вовлечён.

Примеры поведения

Примеры плохого поведения

- В обратной связи участники команды, или даже сам лид переходят на личности.
- Негативная обратная связь даётся публично.
- Обратная связь как позитивная, так и негативная не даётся.
- Участники команд или лид не обращают внимание на полученную обратную связь.

Примеры хорошего поведения

- В обратной связи преобладают факты и ожидания.
- Ожидания не замалчиваются, а даются напрямую участниками команды.
- Лид принимает обратную связь от сотрудников.

Способы прокачки

Практика

Дача обратной связи

- Говорите о конкретной ситуации и конкретных фактах. Это важно как для негативной, так и для позитивной обратной связи. Факты дают вес словам.
- Не стесняйтесь говорить о том, что вам было неприятно в той или иной ситуации. Если вы этого ещё не делаете, то вы приятно удивитесь, насколько это лучше работает, чем отсылки к правилам и нормам.
- Поговорите с людьми, которые не считаются с чужими личными границами. Некоторые на это парируют, что готовы получать в свой адрес такие же «шутки». Но если это рассматривать в

контексте границ, то может получиться, что у человека в шутках и переходах на личности границ может почти не быть. В этом случае можно привести гипотетическую ситуацию там, где они есть, например code review: что будет, если Вася будет сливать код в trunk не обращая внимание на твои комментарии? Ему не нравятся твои шутки – ты с этим ничего не делаешь, ты ему пишешь комментарии – он их игнорирует. Команда без уважения не работает.

Будет полезно выстроить процесс дачи обратной связи сотрудникам. One-to-one встречи - наиболее популярный инструмент для этого. Также полезно периодически собирать обратную связь от коллег непосредственно перед выбором новых целей развития сотрудника. При этом позитивную обратную связь лучше давать публично.

Получение обратной связи

- Дослушайте собеседника до конца, даже если то, что вы слышите, вам неприятно. Частой ошибкой является включение со словами «да, но».
- Помогите вспомнить конкретные кейсы даже если считаете это свидетельствованием против себя.
- Искренне поблагодарите за обратную связь, часто для человека это даётся непросто.
- Задайте уточняющие вопросы, чтобы очистить историю от эмоциональных оценок и получить факты и ожидания.
- Обсудите ожидания, которые не готовы закрыть. Объясните почему. Вместе подумайте как можно было бы выйти из ситуации в состоянии win-win.
- В явном виде проговорите ожидания, которые закроете в будущем.

Для развития лида необходимо получать обратную связь от непосредственного руководителя, а также от сотрудников. Тут также можно посоветовать one-to-one и анкеты 360.

Консультации

- Чат TeamLead Bootcamp

Теория

Статьи

- Как давать и получать обратную связь, если ты воробушек-социофобушек
- Как давать обратную связь: 9 правил
- Метод оценки персонала «360 градусов»

Книги

- Спасибо за отзыв. Как правильно реагировать на обратную связь, Дуглас Стоун, Шейла Хин, 2014

Нетворкинг

Описание

Нетворкинг – это построение и поддержка сети социальных связей как внутри компании, так и вне её. Само понятие предполагает как минимум поверхностное знакомство с людьми и понимание того, чем вы можете быть друг другу полезны.

Почему ветка важна?

Для связей внутри компании:

- Знание о том, кто в компании за что отвечает и как принимает решения.
- Возможность попросить ресурсы или задать вопросы правильным людям.
- Выстраивание отношений, полезных для работы.
- Доступ к информации и новостям ещё до того, как они станут известны официально.

Для связей вне компании:

- Проще нанимать к себе в команду, во многих случаях получится приглашать своих знакомых вместо открытого найма.
- Тебе самому могут предложить интересную вакансию.
- Благодаря тому, что будешь знать, в каких компаниях с какими проблемами сталкивались, сможешь обращаться к ним за помощью в схожих ситуациях.

Что будет, если её не делать?

- Проекты могут тормозить из-за того, что ты не знаешь, к кому правильно обратиться за помощью.
- Из-за отсутствия прочных связей труднее будет получать помощь от коллег.
- Будешь изобретать велосипеды для решения уже давно известных сообществу проблем.

На кого может быть делегирована?

Ветка не делегируется.

Примеры поведения

Примеры плохого поведения

- Не знаешь по имени коллег из других команд и отделов.
- Не можешь сходу сказать, кто отвечает за интересующий тебя вопрос.
- Не знаешь, в каких компаниях решают схожие задачи.
- Не посещаешь митапы и конференции.
- Не участвуешь в тематических сообществах.

Примеры хорошего поведения

- До официального открытия вакансии в первую очередь приглашаешь на неё своих знакомых с подходящим профилем.
- Используешь любой шанс для расширения и укрепления круга знакомств (например, позвать кого-либо на обед).
- Поддерживаешь тесные связи с коллегами внутри компании, с каждым можешь найти тему для разговора.
- Регулярно участвуешь в жизни различных сообществ.
- Не стесняешься обращаться за помощью внутри компании и вне её.
- Часто проводишь гостевые визиты в другие компании, чтобы изучить их процессы и подходы к разработке.
- Приглашаешь коллег из других компаний в гости для обмена опытом.

Способы прокачки

Практика

Построение нетворкинга внутри компании

1. Составь список людей внутри компании, которые могут быть тебе полезны. Это руководители соседних команд, отделов и департаментов, продакт-менеджеры, функциональные лидеры, работники-"старички".
2. Договорись с каждым из них о совместном кофе или обеде.
3. Расскажи о себе, и о том, чем ты можешь быть им полезен. Получи от них ту же информацию. Стоит обсудить, чем человек занимается в компании сейчас, что делал до того, как попасть сюда, чем интересуется, с какими проблемами сталкивается. Если можешь предложить свою помощь – предложи.

4. После встречи запиши куда-нибудь основную полученную информацию. Она может пригодиться в будущем.

Построение нетворкинга вне компании

Найди онлайновые или оффлайновые сообщества по интересующим тебя темам – управлению разработкой, архитектуре, проектному менеджменту, технологиям. Смело подключайся к их жизни – ходи на мероприятия, участвуй в обсуждениях, реализуй совместные проекты. По возможности – знакомься с людьми вокруг, узнавай, чем они занимаются и рассказывай о себе. Гораздо проще такие знакомства заводятся, если ты начинаешь принимать активную роль в жизни сообщества – скажем, выступаешь с докладом или сам организуешь мероприятие.

Консультации

- Чат TeamLead Bootcamp

Теория

Статьи

- Networking Tips for Software Developers
- 39 questions to make small talk with anyone

Книги

- "Никогда не ешьте в одиночку", Кейт Феррацци

Публичные выступления

Описание

Развитие навыка публичных выступлений позволит:

- Структурировать ход своих мыслей и речи
- Делиться с сообществом результатами своих разработок и исследований
- Привлечь большее количество людей к своим идеям и предложениям в рамках рабочих встреч
- Начать выступать на различных площадках в рамках конференций/хакатонов/воркшопов
- Посещать большее количество отраслевых конференций (для спикеров чаще всего участие бесплатное)

Почему ветка важна?

Публичные выступления в разных форматах являются одним из условий для:

- Карьерного роста (проще доверять большие проекты людям, которые говорят уверенно и логично)
- Роста размера возглавляемой команды (чем больше команда, тем чаще перед большим количеством людей придётся выступать)

Что будет, если её не делать?

Если не развивать данный навык, то это:

- Усложнит карьерный путь
- Затруднит продажу и отстаивание своих идей
- Замедлит развитие остальных soft skills, таких как фасилитация и других

На кого может быть делегирована?

Не может быть делегирована

Примеры поведения

Примеры плохого поведения

- Категорический отказ от выступлений, делегация на коллег
- Выступление без подготовки
- Игнорирование обратной связи по выступлению

Примеры хорошего поведения

- Частая практика выступлений
- Проведение нескольких репетиций
- Отработка обратной связи после выступления

Способы прокачки

Практика

- Курсы и тренинги, например Эффективные публичные выступления и презентации
- Помощь опытных коллег, их обратная связь и советы
- Выступление на внутренних митапах в компании
- Выступления в профессиональных сообществах и на конференциях

Консультации

- Чат TeamLead Bootcamp

Теория

Что нужно сделать при подготовке?

- Определить цель и конфликт выступления: что должно произойти после вашего выступления, чтобы вы могли оценить его, как успешное? Почему аудитории должно быть интересно ваше выступление?
- Составьте план выступления: Какими шагами будете идти к цели? Как будет выглядеть ваша история разрешения конфликта?
- Расскажите историю без слайдов: Что в ней стоит добавить/изменить/сократить?
- Накидайте слайды и дополнительные материалы, которые помогут визуализировать вашу историю: Без каких материалов ваша история будет не полной?

- Проведите несколько репетиций: Понятен ли окружающим смысл выступления? Укладываетесь ли вы в установленное время?

На что стоить обратить внимание во время выступления?

- Внешний вид: Должен соответствовать времени, месту и вашей роли, в которой выступаете.
- Вступление: На данном этапе важно расположить аудиторию к себе, выстроить доверительные отношения.
- Зрительный контакт: При постоянном зрительном контакте есть возможность следить за реакцией слушателей и управлять их вниманием. Если почувствуется непонимание или равнодушие, вы можете пояснить мысль или задать вопрос залу.
- Перемещение по сцене: Приближение и отдаление от слушателей во время выступления так же прибавляет динамики, особенно когда это подчёркивает какие-либо моменты рассказа.
- Интерактив с аудиторией: Хорошей практикой будут уместные шутки, истории из жизни, вопросы в зал, комплименты аудитории, обращении по имени.
- Заключение: В заключении вашего выступления вы должны сформировать намерения и побудить аудиторию к действию, чтобы достичь вашей изначальной цели.

Видео

- Публичная речь с Р.Гандапас ч.1: На раз, два, три
- Выступайте – Егор Толстой

Подкасты

- Podlodka #64: Публичные выступления

Статьи

- Выступать на конференции
- Как выступать
- Выступить лучше всех: 63 совета начинающему IT-спикеру
- Легко ли выступать на конференции в первый раз?
- Искусство презентации, или Ни слова о PowerPoint. Ольга Скрипка
- Tips for public speaking
- Tech Talks — You Do Have Something To Say!
- From Proposal to Applause: How I do Public Speaking

Работа с текстом

Описание

Выражать свои мысли менеджеру часто приходится не только устно, но и письменно. Письменная коммуникация может пригодиться в следующих случаях:

- Рабочая переписка в почте или мессенджерах.
- Работа со встречами – повестка и подведение итогов.
- Описание задач.
- Подготовка документации.
- Написание статей.
- Наполнение "Роадмапа Тимлида".

В любом из этих случаев нужно придерживаться ряда базовых правил:

- Базовая грамматическая и синтаксическая грамотность.
- Использование информационного стиля.
- Использование привычного команде предметно-ориентированного языка.

Почему ветка важна?

Для менеджера:

- Гарантия того, что твои мысли и идеи будут правильно поняты.
- Хороший текст повышает вероятность того, что он вообще будет прочитан.

Что будет, если её не делать?

- Тексты будут содержать много воды.
  - Их не будут читать, либо будут просматривать по диагонали, упуская при этом их суть.
- Тексты будут содержать ошибки.
  - Создаётся впечатление неряшливости. Если менеджер пишет с ошибками, возможно, он и в работе будет не слишком хорош.
- Тексты будут не конкретными, не убедительными и не содержать чёткого посыла.
  - Не получится договориться или убедить кого-то в своей точке зрения.

На кого может быть делегирована?

Ветка не делегируется.

Примеры поведения

Примеры плохого поведения

- Менеджер избегает письменных коммуникаций, заменяя их устными.
- Текст содержит грамматические и пунктуационные ошибки.
- Тексты наполнены водой, конкретика размыта.
- За красивыми оборотами теряется смысл.
- Текст слишком формален.
- Текст содержит штампы.
- Автор не думает о читателе.

Примеры хорошего поведения

- Тексты написаны грамотно, с фокусом на ключевую мысль.
- В команде есть культура письменной коммуникации – все важные решения и итоги встреч фиксируются в командной вики.
- Автор текста фокусируется на читателе и его удобстве.

Способы прокачки

Практика

В работе с текстами стоит придерживаться нескольких правил:

- Готовя текст, **думай о том, для кого он пишется**. Почему этот человек будет читать текст? Что он должен в нем увидеть, чтобы решить свою проблему? Каким максимально простым способом эту мысль до него донести? Такие вопросы помогут выкинуть из текста лишнее и структурировать его с ориентацией на конечного читателя.
- **Ищи хороший образец** и придерживайся его. Это особенно верно для подготовки описаний задач, повесток и итогов встреч. Если в компании есть какие-то общепринятые стандарты, то хорошо их придерживаться.
- **Прогоняй текст через** сервис Главред. Он выявляет частые ошибки, сушит воду и делает текст простым для восприятия.

- **Перечитывай перед отправкой.** Всегда.

Консультации

- Чат TeamLead Bootcamp

Теория

Статьи

- Набор статей от glvrd.ru

Книги

- «Пиши, сокращай», Максим Ильяхов
- «Слово живое и мёртвое», Нора Галь

Личный бренд

Описание

Личный бренд – это узнаваемость тимлида в профессиональном сообществе. Личный бренд может носить как позитивную, так и негативную окраску, смешиваться с брендом компании, технологии или какого-то сообщества.

Почему ветка важна?

Для тимлида:

- Упрощает найм команды в плане поиска людей и их мотивации прийти тебе в команду.
- Упрощает удержание существующей команды. Часто встречаемый мотивационный фактор – работа в кругу профессионалов, а личный бренд часто как раз и ассоциируется с профессионализмом.
- Даёт дополнительный бонус к нетворкингу, знакомиться и заводить связи становится существенно легче.
- Открывает больше возможностей для поиска новой работы.

Что будет, если её не делать?

- Уменьшается граф социальных связей, из-за чего сложнее получить помощь или информацию по индустрии.
- Теряется отличный канал найма людей и поиска работы.

На кого может быть делегирована?

Не делегируется.

Примеры поведения

Примеры плохого поведения

- Не выступаешь на конференциях сам, а всегда посылаешь вместо себя кого-то из команды.
- Не делишься опытом в блогах или Twitter.

- Строишь личный бренд, не имея под этим какой-то конкретной основы, например, только на эпатажных высказываниях.
- Считаешь, что сила личного бренда других людей всегда коррелирует с их профессионализмом.

Примеры хорошего поведения

- Регулярно выступаешь на профильных мероприятиях для разработчиков или тимлидов.
- Активно ведёшь Twitter и делишься там своим опытом.
- Ищешь и используешь разные способы усилить свой личный бренд – участие в подкастах, ведение коллективных Twitter аккаунтов.
- Регулярно участвуешь в жизни различных сообществ.
- Общаешься с коллегами из других компаний.

Способы прокачки

Консультации

- Чат TeamLead Bootcamp

Теория

Видео

- Барух Садогурский — Как правильно продать себя ради фана и профита
- Выступайте – Егор Толстой

Подкасты

- Podlodka #93 – Личный бренд разработчика

Статьи

- Зачем айтишнику личный бренд

Понимание ценности различий

Описание

Понимание ценности различий означает найм людей, которые могут отличаться друг от друга и не происходить из одного и того же окружения. Это могут быть различия по национальному происхождению, социальному статусу, внешнему виду, религии, образованию, возрасту, полу или сексуальной ориентации.

Почему ветка важна?

Люди с отличным от нас бэкграундом могут иметь видение и предлагать решения, о которых мы бы никогда не задумались. За счёт отличий между людьми у команды появляется возможность более многогранного разбора проблемы.

- Команда, состоящая из отличающихся людей, меньше подвержена когнитивным искажениям, лучше работает с фактами и более инновационна. (Rock and Grant 2016)
- Разнообразие команды увеличивает производительность. (Deloitte 2013)

Тимлид получает возможность выбирать людей из большего числа соискателей по сравнению с тимлидом, который не может работать с людьми отличными от себя.

Что будет, если её не делать?

- Уменьшение входной воронки кандидатов.
  - Подбор усложняется и затягивается. Конкуренты будут иметь преимущество на этапе найма.
- При прочих равных команда людей с одинаковым бэкграундом перформит хуже.

На кого может быть делегирована?

Первые шаги и настройка процессов могут быть делегированы на HR департамент.

Примеры поведения

Примеры плохого поведения

- Отказ в приёме на работу по половому признаку.
- Команда подбирается исключительно из выпускников технических вузов.

Примеры хорошего поведения

- Члены команды принимают и ценят различия друг друга.
- Бэкграунд и сильные стороны разных людей гибко используются в работе.

Консультации

- Чат TeamLead Bootcamp

Теория

Статьи

Раскрывают тему:

- Про плюсы команды, представленной отличающимися людьми https://hbr.org/2016/11/why-diverse-teams-are-smarter
- Разнообразие команды увеличивает производительность: https://www2.deloitte.com/content/dam/Deloitte/au/Documents/human-capital/deloitte-au-hc-diversity-inclusion-soup-0513.pdf
- Правила diversity на конференциях Ontico: https://www.facebook.com/photo.php?fbid=3098165436891793&set=a.170919312949768&type=3
- Андрей Бреслав — Это выгодно: почему нам нужно больше женщин-программисток: https://youtu.be/iYVVx5ZRm-0?t=26679

Эмоциональный интеллект

Описание

Эмоциональный интеллект – это способность осознавать, правильно различать свои и чужие эмоции. Умение сдерживать, стимулировать и выражать их подходящим способом. А также навык использования эмоций для корректировки поведения, мотивации, мыслительного процесса.

Почему ветка важна?

- Людьми движут эмоции больше, чем мышление. Эмоциональный интеллект позволяет понимать мотивы, как свои так и чужие.
- Правильное использование эмоций может положительно влиять на когнитивные способности: обучаемость, работоспособность.
- Умение выражать и читать эмоции облегчает коммуникации.

Что будет, если её не делать?

- Мотивы людей будут скрыты от вас, поведение плохо предсказуемым.
- При непонимании и/или не правильном выражении своих эмоций повышается вероятность неврозов.
- Часто эмоции сигнализируют о потребностях, и, если их не учитывать, растёт вероятность стресса.
- Сильные эмоции и не способность работать с ними мешают мышлению и принятию решений.

На кого может быть делегирована?

- Эти навыки являются неотъемлемыми для эффективной работы с людьми. Поэтому делегирование в рамках профессии тимлида не возможно.

Примеры поведения

Примеры плохого поведения

- Игнорирование своих и чужих эмоций, отрицание факта их наличия, безразличие к эмоциональному состоянию.
- Не умение сдержать эмоции или выразить их подходящим для ситуации способом.
- Неправильная идентификация эмоций.

Примеры хорошего поведения

- Использование эмоций для управления работоспособностью, мотивацией, улучшения коммуникаций.
- Пытаться понять связь между эмоциями и поведением.

Способы прокачки

Практика

- Ведение дневника эмоций.
- Проговаривание и выражение эмоций.
- Наблюдение за другими людьми.
- Чтение художественной литературы.
- Творчество.

Теория

Книги

- Emotional Intelligence - Daniel Goleman

Работа с привычками

Описание

Привычка - это паттерн автоматического поведения человека. Когда человек не задумывается над тем, что ему делать - он действует по привычке. Поскольку окно осознанности человека крайне мало, то можно сказать, что большая часть решений человек принимает автоматически, то есть руководствуясь привычками.

Работа с привычками нужна, чтобы создавать и поддерживать хорошие автоматические модели поведения, а также чтобы избавляться от плохих. Качество работы с привычками зависит от навыка рефлексии.

Навык работы с привычками необходим для:

- Оптимизации рутинных задач
- Достижения целей с отложенным результатом
- Избавления от ненужных привычек
- Формирования полезных привычек
- Обнаружения и оценки привычек

Почему ветка важна?

Для любого человека:

- Некоторые привычки могут напрямую или косвенно приносить вред человеку.
- Полезные привычки могут принести пользу в долгосрочной перспективе.
- Оптимизация привычек позволит освободить время.

Для тимлида:

- Многие полезные практики в IT являются работой с отложенным вознаграждением и не приносят желаемый эффект если не повторять их регулярно.

Что будет, если её не делать?

- Автоматическое поведение человека не будет меняться или будет меняться медленно

    - Поскольку мы чаще действуем автоматически, чем осознанно, то большинство наших действий будут неправильными или неэффективными
- Цели с отложенным результатом достигаться не будут
    - Большинство наших целей, как в быту, так и в работе, приносят результат не сразу. Чтобы похудеть - нужно заниматься спортом или сидеть на диете хотя бы месяц. Если человек не внедрит соответствующую привычку, то ему придётся полагаться на силу воли. К сожалению, практика показывает, что в таком случае шансы на успех предприятия существенно ниже.
- Человек не будет замечать тривиальные неудобства, которые мешают ему делать бытовые вещи быстрее, а следовательно, не будет работать над их устранением.
    - Это может привести к тому, что мы просто "утонем" в бытовухе: очень сложно работать над крупными проектами когда беспокоишься о грязной посуде или протекающей стиралке.
    - Тривиальные неудобства регулярно отвлекают наше внимание от целей, над которыми мы работаем, тем самым вырывая нас из контекста задачи.
- Человек будет считать, что он ничего не успевает, хотя 90% времени он тратит на привычки, которые больше не отвечают его целям и потребностям.

На кого может быть делегирована?

Развитие этого навыка не может быть делегировано. Однако есть практики, которые требуют поддержки от других людей.

Примеры поведения

Примеры плохого поведения

- "Начну бегать с понедельника" (отложенное внедрение привычки).
- Человек переедает, но поступает по завету бабушки "ешь до конца".
- Человек не видит причинно-следственной связи между привычкой доедать до конца и избыточным весом.
- Человек начинает бегать, но бросает спустя неделю.
- Человек тратит на утренние водные процедуры больше часа.
- Человек пользуется сломанной бытовой техникой, каждый раз совершая "танцы с бубном".
- Человек не изучает свои инструменты. Например, не учит шорткаты в любимой IDE.
- Человек живёт "от зарплаты до зарплаты", хотя получает очень много. Однако он не может начать вести бюджет.
- Возможно разочарование в не до конца внедрённой практике: "у нас были дейлики, но мы на них только время зря теряли, так что мы от них отказались".

Примеры хорошего поведения

- "Начну бегать с понедельника, так что поставлю себе напоминалку на вечер воскресенья, чтобы подготовить вещи".
- Человек ест ровно половину купленной порции, а спустя некоторое время начинает прислушиваться к себе, наелся он или нет.
- Человек пытается серией минимальных экспериментов найти самый подходящий для него способ борьбы с лишним весом и внедряет необходимые привычки согласно результатам экспериментов.
- Человек начинает бегать вместе с другом и благодаря возникшим социальным обязательствам успешно внедряет эту привычку.
- Человек оптимизирует время на водные процедуры за счёт ускорения рутинных действий.
- Человек последовательно устраняет тривиальные неудобства, усложняющие его быт.
- Человек изучает свои инструменты, чтобы сделать свою работу более эффективной.

Способы прокачки

Практика

Изучение имеющихся привычек и среды

Для начала вам необходимо составить список нежелательных моделей поведения. Записывайте всё, что придёт в голову по ходу чтения этой части.

Осознанное прохождение

Мысленно пройдите через все повседневные дела. Ищите вещи, которые можно оптимизировать, от которых нужно избавиться или которые нужно добавить. Например:

- Я не всегда чищу зубы по утрам, нужно всегда.
- Я часто просыпаю, нужно вставать раньше.
- Я часто опаздываю на работу, можно попробовать оптимизировать маршрут и/или вставать пораньше.
- Я часто не завтракаю, нужно завтракать всегда, даже если опаздываю.
- Я часто не успеваю приготовить еду с собой, нужно готовить с вечера.
- Я часто езжу на такси, это лишние траты. Если я буду вставать раньше, то мне удастся этого избежать.
- Я часто прихожу на работу в дурацкой одежде, нужно обновить гардероб и готовить вещи с вечера.
- Я часто мямлю на стендапах, потому что не могу вспомнить, что делал вчера. Нужно планировать дейлик вечером.
- Я часто провожу стендапы в случайное время. Нужно зафиксировать стандартное время в которое должны быть стендапы.

...

- Я часто забываю составить план на следующий рабочий день и утром не знаю, чем заняться.
- Я иногда засыпаю в метро по пути домой и проезжаю свою станцию.
- Я часто по вечерам сажусь играть вместо того, чтобы заняться делами по дому.
- Я часто смотрю сериалы перед сном и, как следствие, поздно засыпаю.

Тривиальные неудобства

Научитесь обращать внимание на бытовые проблемы, которые вам мешают. Примеры:

- Свет в комнате включается не с первого раза.
- Офисное кресло само по себе уезжает вниз и его приходится поднимать обратно.
- Неудобная обувь.
- В ванной полным-полно пустых флаконов из-под шампуня, постоянно приходится искать нужный.
- Вы забываете завести машину перед выходом из квартиры, каждый раз стоите на морозе 2-3 минуты.
- Забываете забрать ланч-бокс из офиса.

Такие проблемы чаще всего являются следствием наличия плохих привычек или отсутствия полезных привычек. Кроме того, люди часто вводят в свою жизнь новые привычки для обхода таких мелких бытовых проблем. Например, ходить чуть поджав мизинец, чтобы не чувствовать дискомфорт, или нажимать на выключатель 4 раза подряд, чтобы свет гарантированно включился.

Техники:

- Замедление: попробуйте выполнить привычный процесс настолько медленно и вдумчиво, насколько возможно. Обращайте внимание на нелогичные и непоследовательные операции.
- Ускорение: попробуйте выполнить привычный процесс так быстро, как только сможете. Обращайте внимание на то, что вас тормозит.

Пространство

Помогает ли ваше пространство выполнению ежедневной рутины? Помогает ли оно избавиться от ненужных привычек? Помогает ли оно получить полезные привычки?

Если вы хотите бегать по утрам, то лучше положить беговую обувь туда, где вы будете видеть её при выходе из дома.

Если вы хотите на регулярной основе заниматься гитарой, то лучше не убирать её после каждого занятия на верхнюю полку.

Если у вас плохо работает душ, то вы каждый день будете тратить ценное утреннее время на войну с ним и устранение последствий этой войны.

Legacy привычки

Есть ли у вас какие-то привычки, которые больше не приносят того результата, ради которого вы их завели? Например:

- Игры, которые больше не доставляют удовольствия
- Пробежки приносят только боль в коленях
- Лента Фейсбука больше не является источником интересной и полезной информации
- Рассылки копятся в непрочитанных

Перемотка времени

Есть ли у вас привычки, с помощью которых вы перематываете время вперёд?

Чтобы обнаружить их, отвечайте себе на вопрос, какую потребность вы удовлетворяете в ходе того или иного процесса.

Если потребность правда есть, то всё в порядке. Если потребность надумана, то и привычка её удовлетворяющая не нужна.

Если вместо этого действия вы можете совершить любое другое с равнозначным для себя исходом - значит, привычка является всего лишь способом перемотать время.

Чаще всего такими привычками являются:

- Просмотр сериалов
- Чтение низкопробной литературы
- Скроллинг соц. сетей
- Чтение неприменимой в жизни информации

Полезно заменить перемотку времени на интроспекцию: попытки понять, откуда возникло желание перемотать время вперёд, что не устраивает здесь и сейчас. Для поиска причин нежелательного поведения хорошо подходит сократовский метод, применённый к самому себе, а также когнитивная терапия.

Целеполагание

Добавьте желаемое поведение к каждому пункту, если его ещё нет. Например: "Я поздно встаю" -> "Я поздно встаю, я хочу вставать раньше".

Изменение привычек

Попробуйте применить каждый из нижеприведённых инструментов к каждому пункту в вашем списке.

План триггеров и действий

Простое десятикратное повторение одной и той же операции при срабатывании триггера. Хорошо работает с рутинными действиями, к которым у вас низкое внутреннее сопротивление. Триггеры и действия составляются по принципу "если-то". Например:

- Если я встал с кровати, то я иду в ванную.
- Если я умылся, то надеваю беговые кроссовки.
- Если я встал с рабочего кресла, то я блокирую компьютер.
- Если я залогинился в свой рабочий комп, то закрываю все неактуальные вкладки.
- Если я взял в руки телефон, то спрашиваю себя - для какой цели.

Чем меньше триггерное действие требует усилий - тем выше шанс, что триггер закрепится.

Также при планировании триггеров помните про контекст: большинству людей достаточно оказаться в ванной комнате, чтобы руки сами потянулись к раковине, а если вы уже обулись в беговые кроссовки, то вряд ли решите отменить пробежку.

Каждый триггер необходимо повторить 10 раз **подряд**. Встать с кровати и пойти в ванную 10 раз. 10 раз взять в руки телефон и задуматься о цели, с которой ты его взял и так далее.

Через несколько дней триггеры можно повторить.

Помогающий дизайн

Сделайте так, чтобы окружающее пространство направляло ваши усилия в нужную сторону. Например, если вы хотите бросить курить - откажитесь от курения дома, выкиньте пепельницу, закройте все окна, выбросьте сигареты и зажигалки. Таким образом, в следующий раз, когда вы захотите покурить, вы задумаетесь, а надо ли оно.

Если вы хотите научиться плавать - упакуйте рюкзак для бассейна и поставьте его возле входа. Каждый раз, когда вы будете через него перешагивать - вы будете вспоминать о своей цели.

Если вы хотите своевременно избавляться от пустых флаконов в ванне - выкидывайте их на пол или в раковину как только обнаружите, что они пусты. Таким образом, вы не забудете о том, что их нужно выкинуть, поскольку они будут на виду даже после того, как вы выйдете из процесса принятия душа и потеряете соответствующий контекст.

Используйте напоминалки. Допустим "медитировать 5 минут". Или "купить яйца". Или "выбросить испорченные продукты". Как только вы получили напоминалку - вашей единственной целью должно быть её выполнение.

Факторизация избеганий

Предположим, вы хотите писать в блог каждый день, но почему-то не делаете этого.

- Сформулируйте избегание.

"Я боюсь, что мои идеи не оригинальны, мой стиль письма не улучшился с пятого класса и я в ужасе от людей в Интернете".

"Я ненавижу вести блог из-за ужасной поддержки LaTeX, из-за беспокойства о проблемах с авторскими правами каждый раз, как возникает желание прикрепить картинку и ещё недавно я обнаружил, что один мой знакомый популярный блогер имеет точно такой же шаблон WordPress, но если я поменяю свой, то я проиграю, а если оставлю тот же самый, то буду чувствовать себя копией, поэтому я, пожалуй, даже думать об этом не буду".

- Решите, поддерживать ли избегание

Разберитесь, указывает ли избегание на реально существующую проблему или нет. В примере выше вполне можно поддержать избегание касательно стиля письма. Соответственно, нужно научиться писать лучше. С другой стороны, мы можем решить, что лучше делать **хоть как-то**, чем не делать никак.

- Избавьтесь от избегания или уменьшите его

Для избеганий, которые вы решили поддерживать, можно поставить новые цели. Допустим, почитать "пиши, сокращай". Для избеганий, от которых вы хотите избавиться, составьте список шагов, каждый из которых будет ощущаться безопасным и комфортным. Допустим, отправлять тексты ежедневно одному человеку кажется менее опасным, чем сразу публиковать их на всеобщее обозрение.

Вневременная теория принятия решений

Привычки формируются тогда, когда мы осознанно выбираем модель поведения в определённой ситуации и она оказывается успешной.

Поэтому полезно взвешивать каждое своё решение через призму того, что в каждой последующей концептуально похожей ситуации мы будем принимать такое же решение. Например:

Надо пойти в больницу, но я боюсь отпрашиваться с работы -> каждый раз в будущем, когда я буду себя плохо чувствовать, я буду сидеть и молчать.

Нужно принять душ, но я уже лёг. -> каждый раз, когда я забуду принять душ прежде чем лечь спать, я буду спать грязным.

Также полезно представлять влияние таких ежедневных решений на окружающий мир. Приступ аппендицита в офисе, грязное постельное бельё и гримасы отвращения от коллег в лифте.

Социальное давление

Чем больше людей знает, что вы бросаете курить, тем выше шансы бросить. Люди хотят быть последовательными в глазах других и мы можем использовать это во благо себе.

Если вы бегаете по утрам не одни, то вам сложнее убедить себя пропустить пробежку, вы ведь не хотите подводить своих товарищей.

Также вы можете отдать другу определённую сумму денег и попросить его вернуть её вам только в том случае, если вы выработаете полезную привычку или откажетесь от вредной. Например: Вась, помоги, пожалуйста. Я хочу бросить курить. Вот тебе 300 баксов. Если ты застанешь меня курящим в течение следующих трёх месяцев - отдай их на благотворительность. Если я брошу - верни их мне.

Принцип солитёра

Есть 50 человек. У каждого есть монетка.

Есть проект. Если проект будет выполнен успешно, то каждый из 50 человек получит 2 монетки, независимо от того, вкладывался он в проект или нет. Если проект будет провален, то все вложившиеся потеряют деньги.

Для того, чтобы проект успешно завершился, нужно хотя бы 20 монет.

Предположим, что все 50 человек принимают решение последовательно. То есть, сначала первый говорит, вкладывает он монетку или нет. Потом - второй. И дальше по порядку.

Какова оптимальная стратегия в такой ситуации для группы индивидов? А для команды?

Теперь другая ситуация. Есть проект. Похудеть за 50 дней. Для достижения эффекта нужно прозаниматься хотя бы 20 дней из 50.

Каждый день вы принимаете решение, вкладывать ли сегодня ресурсы в этот проект или нет.

Каждый день вы думаете, что завтрашний "я" сделает всё за вас.

Какое же поведение в такой ситуации будет оптимальным? Если вам кажется, что сегодня вы не готовы вкладываться в проект, то будет правильным сделать всё возможное, чтобы повысить шансы завтрашнего "я" внести свой вклад. Например: Мне надо в зал, но сегодня я слишком утомился. Но я могу помыть посуду, приготовить еду, упаковать рюкзак для зала на завтра и лечь пораньше, чтобы "я" завтрашний был менее утомлён.

Публичность

Попробуйте освещать свои успехи в изменении привычек в социальных сетях или на специализированных платформах. Это поможет вам не только почувствовать дополнительную ответственность и желание быть последовательным, но также, возможно, даст вам единомышленников, с которыми вы сможете обсудить интересующие вас вопросы по теме.

Пример: можно использовать стриминговые платформы чтобы стимулировать себя писать pet project.

Консультации

Социальный стимул за счёт публичности: https://smartprogress.do/

Сообщество рационалистов, регулярно обсуждающее вышеописанные техники: https://t.me/lwspbhardcore

Теория

Статьи

Раскрывают тему:

- Принцип солитёра: https://radimentary.wordpress.com/2018/01/16/the-solitaire-principle-game-theory-for-one/
- Инструментальная рациональность: https://lesswrong.ru/node/499

Умение учиться

Описание

Умение учиться – способность развивать hard и soft-скиллы для достижения лучшей производительности и результатов.

Почему ветка важна?

В процессе роста лида увеличивается область ответственности, что ставит новые проблемы и задачи. В некоторых случаях они могут быть решены ситуативно, но часто хорошим решением будет приобретение недостающих навыков.

Сложность обучения заключается в:

- Выборе приоритетных целей обучения.
- Правильном подборе способа обучения.

Что будет, если её не делать?

Отсутствие системного подхода к обучению приводит к следующим проблемам:

- Остановка в развитии
  - В определённый момент развитие команды может упереться в уровень её руководителя. Если последний не становится эффективнее, команда тоже топчется на месте.
- Развитие происходит хаотично, нет плана развития.
  - Закрываются не самые критичные области, меньше плюсов от обучения.
- Не сформулированы цели обучения.
  - Если нет целей, то бросить такую деятельность очень просто из-за того, что обучение продолжительно, а начальный запал уже прошёл. Завершение такого проекта приведёт к выброшенному времени, а в некоторых случаях к фрустрации.

На кого может быть делегирована?

Напрямую никому. С советами по обучению могут помочь коллеги, а с планом развития – непосредственный руководитель.

Способы прокачки

Практика

Можно предложить два подхода к выбору приоритетов:

1. Идти от проблем. Составить список кейсов, где можно было поступить лучше. Самостоятельно, или при помощи руководителя, обсудить какие навыки могли бы помочь решить проблему эффективнее.
2. Идти от области ответственности. Больше подходит для случая, когда явных проблем нет. В этом может помочь данный роадмап, алгоритм работы описан тут.

Консультации

- Чат TeamLead Bootcamp

Теория

Курсы

- Learning how to learn

Подкасты

- Podlodka #76 – Обучение

Рефлексия

Описание

Рефлексия – мыслительный процесс, направленный на анализ своих эмоций, мыслей, поведения. По сути является личной ретроспективой.

Почему ветка важна?

Представляет из себя разбор ситуаций, собственных действий и их последствий, мыслей и чувств. Цели такого анализа:

- Возможность продумать оптимальную модель поведения в подобных ситуациях. Сделать выводы из совершенных ошибок.
- Нахождение решения не только для текущего кейса, но и для всего класса подобных случаев.
- Появление лучших жизненных установок. Превращение в лучшую версию себя.
- Отпускание неприятных воспоминаний за счёт детального разбора ситуации.

Что будет, если её не делать?

- Многократное совершение однотипных ошибок.
- Сильное замедление личностного роста.
- Повторение неприятных ситуаций.

На кого может быть делегирована?

Не может быть делегирована.

Примеры поведения

Примеры плохого поведения

- Негативные кейсы не разбираются.
- Выводы не делаются, поведение в аналогичных ситуациях не меняется.
- Рефлексия проводится не регулярно.

Примеры хорошего поведения

- Лид выходит из негативных ситуаций с минимальными потерями.
- Ошибки совершаются, но не становятся системными.
- Разбираются не только негативные исходы, но и положительные кейсы. Понятно, что помогло получить ожидаемый результат.

Способы прокачки

Практика

Несколько советов по проведению рефлексии:

1. Определите предмет рефлексии. Это может быть некоторый промежуток времени или конкретный кейс.
2. Проводите её письменно. Это помогает сфокусироваться, выстроить причинно-следственную цепочку, через некоторое время вернуться к этим записям и пересмотреть их.
3. Выделите определённый временной слот, когда вы будете заниматься рефлексией. Это поможет сделать это действие привычкой.
4. Подготовьте список вопросов, которые помогут вашим рассуждениям. Например, такие:

- Что получилось хорошо? Что стало причиной?
- Как я мог поступить лучше? Как можно было достичь лучшего результата?
- Что я думаю про конкретные кейсы, что чувствую?

4. Периодически возвращайтесь и перечитывайте ваши размышления. Если видите изъян в логике, смело исправляйте.
5. Всегда старайтесь привести рефлексию к каким-то конкретным результатам. Если речь идёт про решение узких прикладных задач, то результатом может быть какой-то action plan. Если речь про что-то более абстрактное – то результатом может быть появление нового принципа или правила, которым вы будете руководствоваться в аналогичных ситуациях.
6. Попробуйте писать в физическом блокноте, даже если привыкли все делать в электронном виде. Сам темп написания текста от руки заставляет хорошо обдумывать написанное.

Консультации

- Чат TeamLead Bootcamp

Теория

Статьи

- How Self-Reflection Can Help Leaders Stay Motivated
- How to Regain the Lost Art of Reflection
- Why You Should Make Time for Self-Reflection (Even If You Hate Doing It)
- How Leaders Use Questions

Книги

- "Принципы", Рэй Далио

Постановка личных целей

Описание

Целеполагание в том или ином виде встречается во всех рабочих ролях. Resource Manager должен уметь ставить людям цели на развитие, Product Owner – определять цели для своего продукта, Technical Lead – цели технического развития, а Administrator – проектные. Эта ветка касается навыка личного целеполагания – того, что отличает эффективность от продуктивности.

Почему ветка важна?

Если ты не умеешь ставить правильные цели и управлять своим фокусом, то нет никакой гарантии того, что даже безукоризненная работа с остальными навыками принесёт в итоге хоть какую-то пользу. Для тимлида важно чётко представлять свои цели потому что:

- Это даёт точку опоры для принятия решений
- Позволяет планировать своё развитие и карьеру
- Помогает отделять важные задачи от неважных

Что будет, если её не делать?

- Когда-нибудь осознаешь, что потратил много времени не на то, чего на самом деле хотел
- Погрязнешь под огромным количеством задач и проектов, не понимая, какой из них тебе действительно нужен
- Не получится уделять внимание всем сторонам своей жизни

На кого может быть делегирована?

Не делегируется.

Примеры поведения

Примеры плохого поведения

- Целеполагание заменяется системой управления задачами, а продуктивность – эффективностью

- Цели ставятся бессистемно
- Определяются только краткосрочные цели, нет работы над общей картиной
- Отсутствует этап рефлексии и анализа результатов
- Целеполагание охватывает только одну область жизни, например, работу

Примеры хорошего поведения

- Есть система личного целеполагания, которая используется в ежедневной деятельности
- Используется несколько уровней постановки целей, как краткосрочные, так и долгосрочные
- Система постоянно улучшается, основываясь на результате своей работы

Способы прокачки

Практика

Система целеполагания – это очень личная и индивидуальная история. Существует много готовых моделей, с которыми можно ознакомиться в разделе "Теория". Слепо использовать не стоит ни одну из них, но в качестве стартовой точки рассмотреть вполне можно.

Консультации

- Чат TeamLead Bootcamp

Теория

Статьи

- Краткий обзор Agile Results
- Статья Стаса Цыганова про его систему личного целеполагания
- Статья Егора Толстого про его систему личного целеполагания

Книги

- Getting Results the Agile Way: A Personal Results System for Work and Life
- Космос. Agile-ежедневник
- Путь джедая
- 7 навыков высокоэффективных людей

Управление приоритетами

Описание

Управление приоритетами – это навык, который помогает тимлиду выжить в условиях полной загруженности задачами. Он помогает:

- В первую очередь делать то, что принесёт больше пользы и ценности.
- Удерживать work/life balance.
- Трезво оценивать свои личные ресурсы.

Почему ветка важна?

- Человеческий мозг очень легко путает понятия срочности и важности. Система управления приоритетами позволяет их оценивать с большей степенью объективности.
- Очень часто начинающие тимлиды сталкиваются с тем, что на них в один момент сваливается огромное количество новых незнакомых задач и проектов. Делать всё сразу – зачастую невозможно, а поступиться чем-то из списка страшно. Без системы управления приоритетами тимлид не сможет быть уверенным в том, что делает именно то, что нужно.
- Опытные тимлиды сталкиваются с другой стороной проблемы. Им периодически нужно решать, готовы ли они вписаться в какую-то новую активность. Навык управления приоритетами позволит им понять место нового проекта среди остальных и количество энергии, которое они готовы будут на него выделить, и в случае его недостатка говорить "нет".
- Для команды и руководителя наличие этого навыка у тимлида важно, так как оно делает его действия и результаты более предсказуемыми.

Что будет, если её не делать?

- Большинство энергии и времени будут уходить на задачи, кажущиеся важными в моменте, а не долгосрочные проекты, которые действительно приносят пользу. Как пример – прилетающие в течение дня письма и сообщения с просьбой быстро помочь автоматически могут вытеснить всё остальное.
- Действия тимлида будут непоследовательными для внешнего наблюдателя, а приоритеты меняться.
    - Нельзя прогнозировать результаты его работы, а, следовательно, доверять ему.

На кого может быть делегирована?

Не делегируется.

Примеры поведения

Примеры плохого поведения

- При появлении нового письма во входящих тимлид сразу бросается на него отвечать.
- Тимлид постоянно жалуется, что он перегружен, и всё вокруг горит.
- У тимлида не хватает времени на долгосрочные проекты, которые могут улучшить жизнь команды и процессы вокруг неё.
- У тимлида сломан work/life balance, и работа занимает всё доступное ему время.

Примеры хорошего поведения

- Тимлид умеет говорить "нет" новым проектам.
- Тимлид балансирует между решением задач, которые горят, и задач, которые улучшат жизнь ему и команде когда-то в будущем.
- Тимлид умеет отказываться от срочных, но не важных задач, в пользу важных.

Способы прокачки

Практика

Для начала возьмите любую возможную систему сравнения проектов и задач между собой и начните её использовать. Через какое-то время вы скорее всего столкнётесь с её ограничениями или недостатками. Тогда переходите к её постепенному изменению под вас – удаляйте, изменяйте или добавляйте новые компоненты и отслеживайте их влияние на ваши процессы и результаты.

Матрица Эйзенхауэра

Суть этого подхода в категоризации всех задач по двум переменным – срочности и важности. Срочность – это то, насколько сильно задача горит. Важность – это то, насколько задача принесёт пользы именно вашим целям. Выглядит эта матрица вот так: | Срочно | Не срочно | | Важно | Не важно | Ваша задача – раскидать по четырём квадрантам все текущие задачи и проекты таким образом, чтобы каждый из них однозначно относился к одной из областей. А после этого начинайте работать с ними по следующей логике:

- **"Важное и срочное"**. В идеале этот квадрант должен быть пустым, так как все важные задачи должны решаться ещё до того, как подошёл момент их дедлайна. В любом случае, задачи этого квадранта выполняются первыми.
- **Важное и не срочное**. Сюда требуется больше всего вашего внимания и усилий. Этот квадрант – залог продвижения по вашим целям.
- **Срочное и не важное**. Эти задачи – по большей части паразиты, съедающие ваши энергетические ресурсы. От них стоит либо отказываться, либо стараться выделять строгий лимит времени на их выполнение.
- **Не срочное и не важное**. Практически все, что сюда попадает – мусор. Смело отказывайтесь от этих проектов, они не нужны ни вам, ни кому-то ещё.

Матрицей можно пользоваться как ежедневно, при разборе своих входящих, так и еженедельно или даже ежегодно, при разборе списка дел.

Анализ Кано

Это гораздо более развесистая методология в сравнении с матрицей Эйзенхауэра. Чаще всего применяется для того, чтобы приоритизировать продуктовый бэклог, но может быть адаптирована и к приоритизации личных дел.

Ключевая суть практики в разделении всех задач на следующие категории:

- **Зелёные задачи**. Если их не сделать, в текущем моменте ничего не изменится, но если сделать – что-то станет сильно лучше. Это что-то может быть вашим приближением к долгосрочным личным целям или радостью команды.
- **Красные задачи**. Если их не сделать, то будет плохо, но если сделать – все просто воспримут это как должное.
- **Серые задачи**. Это то, что мало кого-то волнует, не зависимо от того, сделано оно или нет.
- **Чёрные задачи**. Нежелательные, их вообще делать не стоит.

После того, как вы таким образом категоризировали все свои текущие задачи, вы выстраиваете свой рабочий процесс таким образом, чтобы в нем были только красные и зелёные задачи, причём процент зелёных со временем нарастал.

Взвешенная оценка

Самый интуитивный вариант приоритизации. Вы выбираете систему критериев, по которым оцениваете все задачи, и выводите формулу, по которой сводите их в единый показатель. Например, можно воспользоваться RICE подходом:

- **Reach**. Какое количество людей или других компонентов системы затрагивает решение этой задачи.
- **Impact**. Насколько значимым будет влияние этой задачи.

- **Confidence**. Уровень вашей уверенности в утверждениях выше.
- **Effort**. Количество усилий на выполнение задачи.

Итоговый RICE score подсчитывается по формуле Reach * Impact * Confidence / Effort. В работу берутся первыми те задачи, которые оказались на верхушке получившегося бэклога.

Консультации

- Чат TeamLead Bootcamp

Теория

Статьи

- Матрица Эйзенхауэра и её применение в повседневной жизни
- Модель Кано и анализ Кано – почему это круто и как использовать
- RICE: Simple prioritization for product managers

Книги

- "Джедайские техники", Максим Дорофеев
- "Путь джедая", Максим Дорофеев

Управление временем

Описание

Умение планировать время и распределять его между задачами и видами деятельности.

Почему ветка важна?

Грамотное планирование собственного времени позволяет:

- Достигать целей в срок
- Трезво оценивать собственные силы и не перерабатывать
- Успевать больше

Что будет, если её не делать?

Неумение управлять собственным временем приводит к негативным последствиям:

- Нарушение данных обещаний или сроков
  - Приводит к потери доверия, ухудшению отношений
- Меньше достижений
- Замедление развития
- Состояние постоянной гонки, когда ничего не успеваешь
  - Приводит к негативным эмоциям, депрессии

На кого может быть делегирована?

В большинстве случаев делегирование невозможно из-за того, что для этого необходимо пошарить всю как рабочую, так и нерабочую деятельность стороннему человеку. В редких случаях делегируется ассистенту.

Примеры поведения

Примеры плохого поведения

- Новые проекты появляются быстрее, чем закрываются старые

- Бэклог растёт быстрее, чем разбирается
- Работа идёт по принципу стека, прилетающая задача становится самой важной
- Бэклог никак не защищается, берутся абсолютно все задачи
- Задачи фиксируются только в голове, договорённости забываются
- Жалобы на нехватку времени

Примеры хорошего поведения

- Цели достигаются в запланированное время
- Разбираются как запланированные, так и прилетающие задачи
- Объем сделанных задач и проектов предсказуем и прогнозируем

Способы прокачки

Практика

Существуют две наиболее популярные системы управления собственным временем GTD и Agile Results. Рекомендуем начать с первой.

Теория

Статьи

- GTD in 15 minutes – A Pragmatic Guide to Getting Things Done
- Умей говорить «нет» и умей говорить «да»
- Спонтанное планирование для тех, кто ненавидит тайм менеджмент

Книги

- Getting Things Done: The Art of Stress-Free Productivity
- Getting Results the Agile Way: A Personal Results System for Work and Life
- Джедайские техники Как воспитать свою обезьяну, опустошить инбокс и сберечь мыслетопливо
- Путь джедая. Поиск собственной методики продуктивности | Дорофеев Максим

Scrum

Описание

Фреймворк для управления продуктами. В нем явно определены роли, события и артефакты. Обычно используется в случаях:

- Необходимости часто получать обратную связь от пользователей.
- В условиях высокой неопределённости.
- Конечный результат не может быть определён.
- Когда подходит философия вложения минимальных усилий для достижения цели.

У него отмечают и слабые стороны:

- Строгость заставляет команду резко подстраиваться под фреймворк во время внедрения.
- Строгость чётко определяет роли, иногда это не даёт пространство для маневра.
- Есть мнения, что Scrum хорошо работает на продуктах, которые только запускаются. Вложение минимальных усилий для решения проблемы иногда приводит к увеличению технического долга, что нормально для запускаемого продукта, но может быть критично на дистанции.

Почему ветка важна?

Для тимлида:

- Фреймворк строго описан, в нем продуманы все основные моменты. Понятно от чего отталкиваться.
- Из-за его популярности есть коммьюнити с которым можно посоветоваться, много материалов.
- Небольшие итерации позволяют часто получать обратную связь от пользователей. Это позволяет быстро менять направление движения продукта.
- На рынке есть люди специализирующиеся на фреймворке – scrum-мастера.

Для участника команды:

- Строгость фреймворка гарантирует сохранение основных принципов.
- Основу фреймворка можно изучить самостоятельно.
- Небольшие итерации позволяют держать фокус на небольших промежутках времени.

Что будет, если её не делать?

Scrum не является «самым правильным» фреймворком, у него есть как сильные, так и слабые стороны. Авторы роадмапа считают, что важно, чтобы процесс был продуман. В остальном есть множество успешных команд, которые используют разные фреймворки, не обязательно Scrum.

На кого может быть делегирована?

Часто делегируется scrum-мастерам. У scrum-мастеров есть сертификаты.

Примеры поведения

Примеры плохого поведения

- В команде появляется карго-культ Scrum. Никто не понимает назначение ритуалов Scrum.
- Ритуалы выполняются формально, команда не получает результаты.
- Роли или ритуалы меняются. Scrum гайд говорит, что так делать нельзя.
- С фреймворком знакомы только менеджеры, участники команды – нет.

Примеры хорошего поведения

- Участники команды узнали о Scrum до внедрения.
- Внедрение фреймворк идёт при помощи kick-off.
- Внедряют Scrum люди, которые хорошо с ним знакомы.
- Команда вовлечена и чувствует пользу фреймворка.
- Участники команды получают ответы на вопросы по фреймворку.

Консультации

- Чат TeamLead Bootcamp

Теория

Статьи

Раскрывают тему:

- Библия Scrum: Scrum guide

Дополнительные материалы:

- Podlodka #46 – Scrum-мастерство

Книги

- Джефф Сазерленд «SCRUM. Революционный метод управления проектами»
- Неплохая книга с примерами: The Scrum Handbook

Получение задач

Описание

Процесс, в результате которого у команды появляются задачи, над которыми она работает.

На чем делают фокус:

- Люди, которые ставят задачи команде
- Люди, ставящие задачи конечным исполнителям
- Необходимые условия для получения задачи в команду
- Процесс внутренней подготовки задачи к реализации

Почему ветка важна?

Прозрачный для заказчиков и исполнителей процесс позволяет эффективно выполнять именно те задачи, которые решают проблему. Хорошо выстроенный процесс позволяет:

Тимлиду:

- Прогнозировать загрузку и сроки выполнения прилетающих задач
- Предоставить понятный интерфейс получения задач для заказчиков
- Уменьшить простой исполнителей

Исполнителям:

- Фокусироваться непосредственно на решении задач

Что будет, если её не делать?

Непроработанный процесс получения задач приводит к:

- Дополнительным раундам уточнения цели задачи
- Исполнители неправильно понимают проблему заказчика. Результат не решает проблему
- У исполнителей появляется больше одного постановщика задач. Необходимо приоритезировать задачи не только внутри бэклога, но и между ними
- Начало реализации откладывается, так как не выполнены все предусловия

На кого может быть делегирована?

На проектного менеджера, линейного тимлида

Примеры поведения

Примеры плохого поведения

- Конечный исполнитель переключается между несколькими бэклогами.
- Заказчик общается напрямую с разработчиками, продавливает сроки.
- Исполнители простаивают, потому что задачи в бэклоги не готовы.
- Заказчики не понимают когда будет сделана та, или иная задача.
- Появляются значительные переработки из-за многократного уточнения условий задачи.
- Излишний формализм в постановке задачи не позволяет понять цель её выполнения.

Примеры хорошего поведения

- Роли чётко определены, всем участникам понятен процесс постановки задач в команду.
- Исполнители понимают не только что надо сделать, но и зачем.
- Простои и переработки минимальны.

Способы прокачки

Практика

Следующий алгоритм поможет настроить процесс получения задач в команду

1. Необходимо понимать кто является заказчиками и стейкхолдерами задач, выполняемых командой. Необходимо прояснить ожидания от команды, а также её ответственности
2. Надо сформулировать интерфейс взаимодействия заказчика с командой. Обеим сторонам должно быть понятно в каком виде прилетают требования заказчика. В некоторых случаях это общее описание проблемы с пользовательскими кейсами, сформированные при помощи лида или системного аналитика. В других – согласованное ТЗ с макетами. Аналогично происходит со сроками: или они гибкие и оценка команды согласуется с заказчиком, или дедлайны опускаются сверху.
3. Формируется процесс изменения требований заказчиком. Возможны ли правки? Если возможны, то на каком этапе?

Эти пункты важны для выбора методологии разработки. Обычно она выбирается исходя из необходимости действовать гибко.

Проблемы, с которыми можно столкнуться, и возможные решения:

- Заказчик опускает скоуп и сроки
  - Можно пожертвовать расширяемостью решения. Но надо понимать, что это локальная оптимизация времени, которая приведёт к большим срокам на дистанции
- Во время приёмки задачи становится ясно, что решение не закрывает проблему заказчика
  - Потеря знания о проблеме происходит на одном из этапов: формулирования проблемы заказчиком или превращения требований в задачи исполнителей. Обычно ретроспективно удаётся локализовать потерю этих знаний

Консультации

- Чат TeamLead Bootcamp

PDCA

Описание

PDCA (Plan-Do-Check-Act) или его ещё называют цикл Деминга-Шухарта - это цикл организационного управления качеством и улучшения процессов. Согласно этому циклу, управление должно циклически проходить по следующим стадиям:

- P — plan — планируй
- D — do — делай
- C — check — проверяй
- A — act — действуй

На стадии планирования в цикле Деминга-Шухарта выявляются и анализируются проблемы, а также оцениваются возможности и проектируются необходимые изменения. На стадии реализации происходит воплощение запланированных изменений. На этапе контроля в PDCA оцениваются полученные результаты и строятся выводы, а на стадии корректировки принимается решение о повторении цикла с учётом внесённых поправок. Цикл Деминга очень похож на работу которая проводится над спринтом в Scrum.

Почему ветка важна?

- Помогает улучшать процессы
- Помогает проверять различные гипотезы
- Предотвращает повторение ошибок в процессе работы.

На кого может быть делегирована?

- Проект-менеджера
- Тимлида ниже уровня

Примеры поведения

Примеры плохого поведения

- Выкидывание какого-то этапа из процесса
- Одноразовое использование, PDCA - это про непрерывный процесс

- Не полный план, нет достаточного видения проблемы, которую собираются решать

Примеры хорошего поведения

- Имеется непрерывный процесс прохода по циклу
- Обсуждение с заинтересованными людьми планов
- Ретро по итогам итерации цикла

Консультации

- Чат TeamLead Bootcamp

Теория

Статьи

- Plan-Do-Check-Act (PDCA)
- How to Apply the Plan-Do-Check-Act (PDCA) Model to Improve Your Business

Книги

- Deming Cycle PDCA – Plan Do Check Act Journal in Daily Life TOYOTA way
- Optimization of Raw Material Cost Using PDCA Cycle

Стейкхолдинг

Описание

Стейкхолдер – это лицо, чьё действие, поведение или решения могут повлиять на результаты проекта. Также стейкхолдерами часто называют людей, заинтересованных в успехе проекта и ожидающих получить от него какую-то выгоду.

Работа со стейкхолдерами детально расписана в других ветках роли Administrator. В этой ветке рассматривается, как тимлиду самому выступать хорошим стейкхолдером.

Почему ветка важна?

Для тимлида:

- Если ваша компания представлена чем-то большим, чем ваша команда, то вы зависите от результатов работы других групп людей. Чтобы быть уверенным в получении от них нужного вам результата, нужно уметь выполнять роль стейкхолдера.
- Вспомните, насколько вам сложно работать с проектами, стейкхолдеры которых ведут себя пассивно. Не делайте так.

Что будет, если её не делать?

- Вы можете не повлиять на результаты проекта вовремя и не получить нужного результата
- Из-за вас команда может не достичь результатов, и вы подведёте зависящих от неё людей
- Если вы не выдвигаете ожиданий от исполнителей, либо делаете это слишком поздно, то им в какой-то момент придётся очень много переделывать

На кого может быть делегирована?

В зависимости от того, к какой группе стейкхолдеров вы относитесь, вы можете делегировать уровни своего участия в проекте любому другому члену команды.

Примеры поведения

Примеры плохого поведения

- Стейкхолдер не вовлечён в процесс работы над проектом, а появляется эпизодически в роли чайки, либо не появляется вообще
- Стейкхолдер вмешивается слишком часто и активно, в результате чего работа над проектом замедляется
- Стейкхолдер не прогнозирует, как его решения или действия влияют на все проекты, в которые он вовлечён

Примеры хорошего поведения

- Стейкхолдер понимает свою роль и действует сообразно ей – как заказчик, консультант или партнёр
- Стейкхолдер всегда представляет, чего именно он ждёт от проекта и чего с его помощью должен добиться

Способы прокачки

Практика

1. Определите свою роль как стейкхолдера в проекте:

- **Партнёр**. Уровень влияния – высокий. Уровень важности – высокий. Основной стейкхолдер, должен максимально привлекаться к принятию решений. В этой роли вы должны постоянно быть в курсе статуса проекта, выступать в активной роли, не бояться "ставить шкуру в игре". Если вы выступаете в этой роли, то от результатов проекта должно зависеть что-то, сильно вас волнующее. Будьте проактивны, не ждите, пока исполнители придут к вам сами.
- **Консультант**. Уровень влияния – высокий. Уровень важности – низкий. Второстепенный стейкхолдер. С вами согласуются ключевые решения по проекту, но вы напрямую в его итогах не заинтересованы. В этой роли можно быть более пассивным, и ждать, пока исполнители придут к вам сами.
- **Поддержка**. Уровень влияния – низкий. Уровень важности – высокий. Второстепенный стейкхолдер. Вы не принимаете непосредственного участия в принятии решений по проекту, но должны быть с ними ознакомлены. В результатах проекта вы заинтересованы, но не настолько сильно, чтобы вовлекаться в него активно. В этой роли вы выступаете помощником исполнителя проекта, и он вас задействует, когда сам сочтёт нужным.
- **Временный работник**. Уровень влияния – низкий. Уровень важности – низкий. Второстепенный стейкхолдер. Скорее всего, вы просто эпизодически выполняли какую-то роль в процессе работы над проектом, но в целом в нем ни заинтересованы, ни помочь чем-то не можете. В этом случае не нужно вообще в него вовлекаться, просто оставьте исполнителей в покое. Вы там не нужны.

2. Чётко проговорите свою роль и ожидания от неё с исполнителями проекта. Договоритесь с ними о процессе вашего вовлечения, конкретных датах и критериях успешности.

3. Запрашивайте обратную связь от исполнителей проекта по тому, как вы выполняете свою роль.
4. Периодически переоценивайте свою роль в проекте. Вполне возможно, что со временем из роли партнёра вы можете переместиться в роль поддержки или временного работника. Если такое произошло – проговорите с исполнителями ожидания ещё раз.

Консультации

- Чат TeamLead Bootcamp

Теория

Статьи

- Стейкхолдер в Systems Engineering Thinking Wiki
- Project Stakeholder Management According to the PMBOK
- Stakeholders in Software Architecture

Знание бизнеса

Описание

Любая команда разработки работает не в вакууме. Итоговая цель – не выпустить больше задач, не сделать самый качественный продукт, а принести пользу бизнесу. Чтобы понять, откуда берётся и в чем выражается эта польза, тимлиду важно ориентироваться в бизнесе компании. Минимальный достаточный набор знаний:

- Бизнес-модель компании Как устроен бизнес, за счёт чего функционирует и зарабатывает деньги.
- Структура доходов Как распределяется прибыль компании между разными каналами доходов. Что приносит деньги, а что является экспериментом.
- Стратегия компании Долгосрочные цели компании и планы по их достижению.
- Бенефициары Кто в итоге получает прибыль компании – один человек, несколько, или большая группа акционеров.

Почему ветка важна?

Для менеджера:

- Можешь донести до команды ценность результатов её работы.
- Есть компас, по которому можно ориентироваться, чтобы понимать, какие проекты делать, а какие – нет.
- Уменьшается степень неопределённости будущего, шаги развития компании становятся более предсказуемыми.

Для команды:

- Проще генерировать новые идеи, опираясь на понимание нужд бизнеса.
- Внешние слухи и новости о состоянии компании проходят через фильтр критической оценки сотрудниками.
- Видя прозрачность со стороны компании, понимая больше цели и планы, команда чувствует себя частью чего-то большего, а не отдельной независимой сущностью – повышается мотивация и вовлеченность.

Что будет, если её не делать?

- В один прекрасный день можно узнать, что твой проект закрывают, как не прибыльный или не профильный.
  - Команда будет не подготовлена к такому сценарию развития событий, повышается риск демотивации и ухода сотрудников.
  - Впустую потратятся усилия по развитию проекта, хотя ты мог бы предсказать такой сценарий и раньше.
- Не увидишь новых возможностей для развития компании.
  - Потеряешь возможность сделать что-то, что поможет твоей карьере.
- Команда будет узнавать о состоянии компании от третьих сторон и СМИ, не различая факты и домыслы.

На кого может быть делегирована?

Эта задача не делегируется. Тимлид обладает более широким контекстом, чем любой отдельно взятый член его команды.

Примеры поведения

Примеры плохого поведения

- Менеджер не может ответить на вопрос как компания зарабатывает деньги.
- Команда не знает, как результаты её работы влияют на стратегию компании или её прибыль.
- Для менеджера и команды все новости от руководства становятся неожиданными.
- Новости о состоянии дел в компании команда черпает из новостей.

Примеры хорошего поведения

- Менеджер и команда разбираются в базовых экономических показателях своего продукта.
- Менеджер может рассказать о структуре доходов компании и её бизнес-модели.
- Команда знакома со стратегией развития компании и понимает причины её принятия.

Способы прокачки

Практика

Вся задача разбивается на две части – получить первичные знания и наладить канал получения обновлений и новостей.

Получение первичных знаний

1. Найди в структуре компании человека, который обладает нужными знаниями: может рассказать про бизнес-модель, структуру доходов, стратегию. Начать можно с непосредственного руководителя. Он или ответит на часть вопросов сам, или подскажет, к кому ещё можно подойти. Другие потенциальные кандидаты – все люди с приставкой Chief (CEO, CTO, CFO), продакт-менеджеры и их руководители, финансисты.
2. Получи ответы на следующие вопросы:

- Как устроена бизнес-модель компании?
- Какие основные источники доходов и какой процент от общей прибыли они составляют?
- Как выглядят базовые финансовые показатели компании?
- В чем заключается стратегия компании и где с ней можно ознакомиться?
- Почему стратегия компании выглядит именно так? Какие альтернативы рассматривались?
- Кто владеет компанией, получает прибыль и принимает решения по её судьбе?

Обновление знаний

1. Договорись с человеком из предыдущего раздела о том, где и как узнавать про обновления в финансовых показателях и стратегии. Это может быть какая-то e-mail-рассылка, аналитика, документы или регулярные встречи.
2. Регулярно обновляй свои данные и рассказывай команде об этих изменениях. Не забывай объяснять, как эти изменения могут повлиять на них и что они на самом деле значат.

Консультации

- Чат TeamLead Bootcamp

Теория

Статьи

- Что такое бизнес-модель
- Виды бизнес-моделей на примере разных компаний
- 10 популярных бизнес-моделей с их плюсами и минусами
- Анализ стратегии 1С-Битрикс
- Ключевые метрики бизнеса

Книги

- Business Model Generation – Alexander Osterwalder, Yves Pigneur

Видео

- Краткое и простое введение в теорию бизнес-моделей и инструменты для выявления потребностей пользователя (модули 2,3)
- От хорошего к великому. Почему одни компании совершают прорыв, а другие нет..., Джим Коллинз
- Созидательное разрушение. Почему компании, "построенные навечно", показывают не лучшие результаты и что надо сделать, чтобы поднять их эффективность, Ричард Фостер, Сара Каплан
- И ботаники делают бизнес, Максим Котин

Структура компании

Описание

Роль тимлида в рамках этой ветки состоит в следующем:

- Разбираться в разных типах оргструктур.
- Определять место своей команды внутри организации, связывать их цели.
- Оптимизировать процессы таким образом, чтобы они не входили в конфликт с организационной структурой.

Типологий организационных структур очень много, причём большинство из них довольно развесистые. Чтобы не усложнять ветку, приведём основные варианты, встречающиеся в IT:

- **Единая команда** Все IT-специалисты, вне зависимости от функции, находятся в одной команде, подчиняющейся одному человеку. Чаще всего актуально для небольших стартапов.
- **Функциональная структура** Разделение на команды в зависимости от функции человека: mobile, backend, design, qa. Руководитель обычно принадлежит той же функции.
- **Матричная структура** Сотрудники определяются своим положением в матрице, на одной оси которой – функциональная принадлежность, на другой – проектная. У каждого два руководителя – функциональный и проектный.
- **Кроссфункциональная структура** Разделение на автономные команды, каждая из которых может работать независимо от других. В таких командах смешиваются специалисты разных функций, часто под одним руководителем.

Почему ветка важна?

Для менеджера:

- Исходя из закона Конвея, структура коммуникаций в компании определяет архитектуру разрабатываемых систем. Это значит, что при их проектировании нужно эту структуру ясно представлять и учитывать.
- У всех моделей структуры есть как плюсы, так и минусы. Менеджер должен в них ориентироваться, чтобы понимать, что является фундаментальным недостатком, а что можно исправить.
- Иногда у тимлида есть возможность либо повлиять на организационную трансформацию, либо самостоятельно в рамках своего отдела экспериментировать со структурой.

- На вопросы сотрудников "А почему у меня два руководителя?" получится отвечать не только, что так исторически сложилось, а детально раскрывая предпосылки такого решения и его последствия.

Что будет, если её не делать?

- Можно вслепую пытаться интегрировать процессы и практики, отлично работающие в какой-то определённой оргструктуре, но никак не подходящие под твой сценарий.
- Из-за архитектуры, не соответствующей модели компании, разработка будет замедляться, а технический долг накапливаться.

На кого может быть делегирована?

Часть, касающаяся знания подходов к организационной структуре, не делегируется. Этими знаниями должен владеть сам менеджер. Интеграция команды в организацию и оптимизация процессов может быть делегирована проектному менеджеру, продуктовому менеджеру, руководителю уровнем выше.

Примеры поведения

Примеры плохого поведения

- В текущей организационной структуре тимлид видит только минусы или только плюсы.
- При проектировании архитектуры не учитывается структура компании или её доменная модель.
- Сотрудники не понимают, чем занимаются или за что отвечают другие команды.
- Цели команды никак не связаны с целями остальной части компании.
- Есть части системы, за которые никто в компании не отвечает.
- У тимлида не получается выстраивать процессы, требующие взаимодействия между разными командами. Каждый раз проблемы решаются ситуативно.

Примеры хорошего поведения

- Каждый сотрудник понимает, какое место команда занимает во всей организации, как их цели связаны друг с другом, чем занимаются его коллеги.
- Тимлид понимает, когда текущая модель организационной структуры перестаёт решать поставленные перед ней задачи, и указывает на это руководству.
- Тимлид принимает активное участие в определении организационной структуры компании.

Способы прокачки

Практика

1. Нарисуй простую схему своей организации. Отрази в ней существующие отделы, их связи друг с другом и структуру подчинения людей. Фокусироваться имеет смысл именно на техническом департаменте.
2. Смотря на схему, определи, к какому типу организационной структуры относится твоя компания. Обрати внимание на следующие признаки:

- На какое количество команд поделена вся разработка?
- Состоят ли эти команды из специалистов одной функции или разных?
- По какому принципу выделяется команда?

3. Почитай статьи из раздела "Теория" про организационную структуру твоей компании. Выдели плюсы и минусы. Посмотри, как они проявляются конкретно в вашем случае.
4. Проанализируй позицию своей команды в рамках компании. Выделена ли она по тем же принципам, что и остальные? Как её цели соотносятся с целями всей организации и других команд?
5. Обсуди результаты своего исследования с руководителем. Правильно ли составлена схема, верно ли ты выделил её составляющие части? Все ли плюсы и минусы нашёл? Как быть, если твоя команда в текущем виде не ложится в остальную компанию?
6. Расскажи своей команде о результатах исследования. Покажи всем схему компании, расскажи, кто и за что отвечает. Можно это сделать на общей командной встрече, можно на серии one-to-one.

Консультации

- Чат TeamLead Bootcamp

Теория

Статьи

- Введение в холакратию
- Эволюция организационной структуры Badoo
- Ещё один подход к типологии организационных структур
- Недостатки матричной структуры
- Как создавать кроссфункциональные команды
- 4 Tips to Help You Lead Matrix Teams Successfully

Видео

- Spotify Engineering Culture

Корпоративная культура

Описание

Роль тимлида в вопросах корпоративной культуры и ценностей достаточно простая – он должен уметь их идентифицировать, использовать в действиях и принятии решений и передавать своей команде. Помимо этого, также как и любой другой человек в компании, менеджер неявно влияет на формирование культуры через свои осознанные и неосознанные действия.

Корпоративная культура состоит из следующих компонентов:

- **Видение.** Смысл существования компании, её большая цель и миссия. Именно для его реализации компания была когда-то создана.
- **Ценности.** Набор принципов, определяющих образ мышления и поведение сотрудников компании.
- **Практики.** Реализация ценностей в конкретных практиках: развитии, оценке, принятии решений.
- **Люди.** Насколько сотрудники компании разделяют её видение и ценности, как производится найм новых.
- **Нарратив.** История компании, мифы, ключевые решения.
- **Место.** Страна, город, конкретная локация. Внутренний дизайн офиса и организация рабочего пространства.

Почему ветка важна?

Для компании:

- Культура – основа для принятия решений и действий в условиях неопределённости. Даже в случае различающихся целей люд в компании могут договориться, если разделяют единые ценности и сходны по культуре.
- Корпоративная культура, подкрепляющая стратегию компании, позволяет добиваться более высоких результатов.

Для менеджера:

- Соответствие корпоративной культуре и ценностям – существенный маркер для приёма на работу.
- Построение любых процессов должно отталкиваться от корпоративной культуры.

Для сотрудника:

- Работать рядом с людьми, разделяющих те же ценности, что и ты – важный мотивационный фактор.

Что будет, если её не делать?

- Ценности, существующие только на бумаге, и не соответствующие реальности, не будут приняты командами и будут восприниматься как корпоративный буллшит.
  - Создаётся иллюзия контроля и понимания корпоративной культуры, что может привести к неожиданному поведению и результатам.
  - Даже при последующих более удачных попытках сам факт использования ценностей будет уже дискредитирован.
- Команда, не разделяющая корпоративную культуру, будет оторвана от остальной компании и расходиться с ней даже в базовых вопросах.
  - Команде будет трудно договариваться.
  - Результаты, которыми команда будет гордиться, с точки зрения культуры или ценностей могут быть некорректными.
- Если не ориентироваться на корпоративную культуру в работе с людьми, со временем она распадётся.
  - Будут наниматься не соответствующие остальной команде люди, которым самим будет тяжело интегрироваться в коллектив.

На кого может быть делегирована?

Миссия и видение обычно спускаются на команду сверху. В распространении ценностей и встраивание их в общекорпоративные процессы часто принимают участие HR. Корпоративная культура и ценности должны одинаково разделяться не только менеджером, но и его командой. Задача менеджера быть проводником этой культуры, и своим примером показывать, как она применяется на практике.

Примеры поведения

Примеры плохого поведения

- Соответствие культуре и принятие ценностей не проверяются на входных интервью.
- Ценности написаны на стенах офиса, но не применяются на практике.
- Упоминание ценностей вызывает смех и шутки у команды, они стали источником мемов.
- Команда не понимает миссии компании, либо считает, что она заключается только в зарабатывании денег.

- В команде постоянно возникают споры насчёт базовых вопросов. Например, что важнее – скорость или качество.
- Менеджер не поддерживает ценности и культуру своим примером.

Примеры хорошего поведения

- В процессе интервью проверяется соответствие кандидата культуре компании с использованием кейсов и примеров из его жизни.
- При даче обратной связи менеджер ориентируется на ценности компании и принятые в ней модели поведения.
- При отсутствии в компании внятной культуры менеджер создаёт "пузырь" в своей команде, культивируя те качества, которые помогают добиваться лучшего результата для бизнеса.

Способы прокачки

Практика

Определение культуры своей компании

Есть много различных подходов к типологии корпоративных культур. Один из сильных вариантов описан в статье "The Leader's Guide to Corporate Culture" от Harvard Business Review. Для определения типа культуры используются две оси:

- Взаимодействие между людьми (от независимой к взаимозависимой работе).
- Реакция на изменения (от стабильности к гибкости).

В зависимости от позиции компании по каждой из этих шкал выделяется восемь основных типов корпоративных культур:

- **Забота.** Фокус на взаимоотношениях между людьми и взаимном доверии. Люди помогают и поддерживают друг друга, лидеры пропагандируют командную работу, честность и открытость.
- **Смысл.** Фокус на идеализме и альтруизме. Все сотрудники компании думают о большой и значимой цели, идеалах, создании добра.
- **Обучение.** Фокус на креативности и исследовании нового. В работе приветствуется открытость, генерация новых идей и обмен ими. Людей объединяет любопытство, инновации, приключения.
- **Удовольствие.** Фокус на веселье и кайфе. Люди занимаются в первую очередь тем, что делает счастливыми именно их.
- **Результат.** Фокус на достижениях и победе. Большой упор делается на конкретные результаты и заслуги, высокий перфоманс. Люди объединяются вокруг успеха и высокого потенциала.
- **Власть.** Фокус на силу и решимость. В компании культивируются индивидуальный перфоманс и достижения. Людей объединяет строгий контроль.

- **Безопасность.** Фокус на планировании, готовности к изменениям, осторожности. Люди думают о рисках, продумывают все свои действия наперёд. Для сотрудников компании важно чувствовать уверенность в завтрашнем дне, изменения принимаются неохотно.
- **Порядок.** Фокус на уважении, структурности, правилах. Люди привыкли играть по правилам, и им это нравится.

Чтобы определить, к какому из этих типов культур относится именно ваша компания, можно пройти онлайн-тест.

При определении вашей культуры и ценностей важно смотреть не на то, что заявляется CEO, пишется на стенах или бесплатных футболках, а на то, как эти принципы применяются людьми в реальной жизни. Например, если одной из ценностей является доверие и честность, но люди все равно боятся давать друг другу открытую обратную связь, это значит, что желаемое расходится с действительностью.

Использование культуры

1. Если выделить единые ценности компании не получается, то есть смысл определить список ценностей на уровне своей команды. Они должны быть такими, чтобы:

- Отвечать культуре (она все-таки должна выявляться).
- Помогать команде достигать результатов, которые ожидает бизнес.
- Разделяться членами команды.
- Иметь возможность быть реализованными через процессы и практики команды. Пример таких командных ценностей для команды iOS разработки Rambler&Co.

2. Пересмотрите все существующие процессы. Учитываются ли ваши ценности в них? Стоит отдельно проработать ветки:

- Найм. Проверяем ли новых людей на culture fit.
- Мотивация. Соотносятся ли наши инструменты влияния на мотивацию с ценностями.
- Развитие. Каким образом развиваем людей, учитываем ли по большей части их желания или нужды компании.
- Промо. За что повышаем людей.
- Обратная связь. Даётся ли, как часто, насколько она полная и честная.
- Климат в команде. Сходится ли то, что заявляется в ценностях, и реальный климат внутри коллектива.
- Управление знаниями. Делимся ли мы знаниями или предпочитаем держать их внутри.

3. На one-to-one при даче обратной связи своим подчинённым используй соответствие ценностям и культуре как один из сигналов. Рассказывай, почему это важно, но старайся не уходить в буллшит и корпоративщину – подтверждай все простыми примерами.

Консультации

- Чат TeamLead Bootcamp

Теория

Статьи

- Лонгрид про составляющие корпоративной культуры и хороший фреймворк по её определению
- Виды ценностей и стандартные ошибки работы с ними
- Ценности GitLab
- Культура и ценности Miro
- Из чего состоит корпоративная культура

Книги

- Powerful: Building a Culture of Freedom and Responsibility

Административная работа

Описание

Административная работа тимлида – это набор действий и знаний, которые требуются ему для соблюдения текущего трудового законодательства и правил компании при управлении командой. Сюда включаются:

- Финансовые отношения с сотрудником: правила формирования заработной платы, проведения индексаций.
- Отпуска, отгулы, больничные.
- Трудовой договор, его заключение и прекращение.
- Рабочий распорядок. Чаще всего эта ветка вырождается в подписывание разных бумаг и периодические консультации с юристами, финансистами и HR.

Почему ветка важна?

- Выполнение таких вещей, как подписание заявок на отпуск, согласование отгулов и больничных, увольнения, входят в ожидания компании от тимлида.
- Для комфорта и безопасности сотрудников важно, чтобы их права соблюдались.
- Соблюдение прав сотрудников важно для того, чтобы в конфликтных ситуациях не появилось проблем с законом.

Что будет, если её не делать?

Если говорить про знание ТК и текущих условий компании на уровне большем, чем слепое подписывание бумаг, то в большинстве случаев не будет ничего критичного. Примеры некоторых нежелательных последствий:

- Сотрудник взял недельный отпуск в январе. В следующей зарплате он получает значительно меньше денег из-за правил расчёта стоимости отпускных дней, о котором он не знал.
- Сотрудник отказывается уходить из компании по соглашению сторон, но из-за некорректно составленного трудового договора вы не можете его уволить из-за невыполнения должностных обязанностей.

На кого может быть делегирована?

- На корпоративных юристов, финансистов, HR.

Примеры поведения

Примеры плохого поведения

- Тимлид не знает, как у его сотрудников формируется заработная плата.
- Тимлид не разбирается в отличиях отгулов, оплачиваемых и не оплачиваемых отпусков.

Примеры хорошего поведения

- Тимлид способен ответить на базовые вопросы сотрудника, касающиеся правил компенсации, трудового распорядка и отпуска.
- Тимлид знает, к кому в компании с какими вопросами по соблюдению прав сотрудника можно подойти.

Способы прокачки

Консультации

- Чат TeamLead Bootcamp

Теория

Статьи

Раскрывают тему:

- Что нужно знать о трудовых правах каждому работнику
- Что нужно знать о заработной плате каждому сотруднику

Дополнительные материалы:

- Трудовой Кодекс РФ

Делегирование

Описание

Процесс передачи полномочий или части полномочий сотрудникам.

Почему ветка важна?

Собственное развитие лида зависит от развития людей, с которыми он работает. Лид не сможет выполнять все более сложные задачи, если он не будет обучать людей делать то, что уже сам умеет, а в последствии полностью делегировать.

Почему необходимо развивать навыки делегирования? Для лида:

- Освобождение своего времени.
- Развитие сотрудников.
- Подготовка преемника.
- Снижение bus factor в команде.

Для сотрудника:

- Возможность развития.
- Расширение полномочий.
- Карьерные перспективы.

Для команды:

- Повышение кросс-функциональности.
- В некоторых случаях получение большей самостоятельности.

Что будет, если её не делать?

Причины, по которым лиды не делегируют собственные задачи:

- Опасения за качество выполненной работы.
- Боязнь, что его «подсидят».
- Как это не забавно, «нет времени».

Это приводит к:

- Команда не делает всё более сложные задачи
- Развитие руководителя замедляется/останавливается
- У людей в команде не происходит расширения полномочий
  - Влияет на удержание людей

На кого может быть делегирована?

Примеры поведения

Примеры плохого поведения

- Написание кода на «критическом» пути не делегируется.
- Задачи делегируются, но нет контроля исполнения.
- Лид ошибается в выборе уровня делегирования для конкретного сотрудника.
- Делегирование происходит сразу всей команде в надежде, что кто-то да сделает.
- Не обозначена периодичность отчётности по задаче.
- На человека делегируется больше задач, чем он способен выполнить.

Примеры хорошего поведения

- Часть полномочий и задач регулярно передаётся участникам команд.
- Лид вкладывается в людей так, что при делегировании требуется всё меньше участия.
- Обозначены сроки, критерии оценки и критичные условия при постановке задачи.
- В случае, если исполнитель не способен решить задачу целиком, формулируется и отдаётся только та часть, которая может быть выполнена.

Способы прокачки

Практика

1. В первую очередь необходимо оценить, что из собственной работы, и кому, можно делегировать прям завтра. Такие области легко найти задав себе вопрос: что произойдёт, если отдать эту задачу конкретному человеку в команде?
2. Надо научиться разным стилям делегирования и подбирать их под конкретных людей. Со стилями можно ознакомиться например в этой статье.
3. Необходимо встретиться с исполнителем, делегировать ему задачу и проконтролировать результат. После нескольких повторяющихся кейсов можно полностью передать данную ответственность этому исполнителю.

4. Искать новые области для себя, делегировать собственные ответственности участникам команды.

Консультации

- Чат TeamLead Bootcamp

Теория

Статьи

Раскрывают тему:

- The 7 Levels of Delegation

Развитие

Описание

Развитие навыков сотрудников – основной способ увеличения производительности и качества работы команды. Основной фокус в этом направлении делается на технических и софт скиллах конкретного сотрудника.

Увеличение перфоманса команды – одна из ключевых задач руководителя. Эту задачу можно решать при помощи найма, но количество мест в офисе ограничено, также при росте команды добавляются стоимость на управление и усложняются коммуникации.

Почему ветка важна?

Для менеджера:

- Один из хороших способов выстраивания отношений руководитель <-> подчинённый.
- Повышение перфоманса конкретного человека.

Для сотрудника:

- Возможность решать более сложные и интересные задачи.
- Рост зарплаты и стоимости на рынке.

Что будет, если её не делать?

Без целенаправленного развития во многих случаях люди будут развиваться хаотично.

- Сотрудник просто стоит на месте в развитии. Это плохо, потому что на рынке есть намного более перспективные люди, в том числе и у конкурентов.
- Человек развивается не в том направлении. В конечном счёте аналогично предыдущему пункту.
- Сотрудник не чувствует, что ему уделяют внимание, складывается ощущение, что «работает и работает».

На кого может быть делегирована?

- Тимлиды ниже уровнем
- Scrum Master

Примеры поведения

Примеры плохого поведения

- Не ставить индивидуальные цели по развитию для сотрудника, или ставить их не в соответствии со SMART
- Ставить цели по развитию сотрудника, не учитывая его мнение и желание - компания и сотрудник успешны, когда их цели пересекаются
- Не фиксировать достижение промежуточных целей - сотрудник может потерять интерес к выполнению целей без позитивного подкрепления и конструктивной обратной связи
- Не пересматривать цели по развитию - цели могут устареть и потерять актуальность
- Не создавать общую библиотеку для сотрудников - многие не готовы покупать дорогие книги, особенно сразу после их выхода
- Не иметь бюджет на обучение - далеко не каждый сотрудник может и умеет учиться самостоятельно, используя документацию или книги

Примеры хорошего поведения

- Ведение "звёздной карты" команды для того, чтобы сотрудники могли как усиливать свою основную компетенцию, так и изучать смежные профессии
- Составление плана развития сотрудника, пересмотр данного плана на периодической основе
- Поощрение самостоятельного обучения на рабочем месте
- Book sharing - общая библиотека для обмена бумажными книгами
- Подписка для сотрудников на онлайн-библиотеки или обучающие порталы
- Оплата внешних курсов/конференций/материалов для поддержания плана развития сотрудника

Способы прокачки

Практика

Можно начать с постановки целей на долгосрочный промежуток времени: квартал, полугодие. Идеи для целей можно генерировать при помощи коллег, тимлида ниже уровнем, по результатам текущей работы, а также из хотелок самого человека. Эти идеи нужно обсудить на личной встрече, совместно отбросить бесполезные, выбрать на чем сфокусироваться.

Формирование конечных целей, а также их достижение лучше оставить на сотруднике, потому что цели должны быть в первую очередь его. В том, что является DoD цели можно помочь, лучше использовать, например, SMART. Надо дать возможность подумать над ними и через некоторое время их зафиксировать.

Если говорить про непосредственно достижение целей, то хорошей практикой будет обучение 70/30, где 30% – это теоретическое обучение: курсы, книги, тренинги, а 70% непосредственно задачи на работе. Также цели лучше ставить исходя из повседневной деятельности.

Необходимо мониторить и обсуждать продвижение по этим целям, иначе к концу периода постановки целей может оказаться, что о них просто забыли. Например, их можно обсуждать на встречах one-to-one.

Консультации

- Чат TeamLead Bootcamp

Теория

Книги

- Семь навыков высокоэффективных людей. Мощные инструменты развития личности, Стивен Кови
- Драйв. Что на самом деле нас мотивирует, Дэниел Пинк

Обратная связь

Описание

Обратная связь – это механизм корректировки поведения и результатов работы сотрудника. Всю работу с обратной связью можно разделить на два процесса: сбор и дача сотруднику. Обратная связь характеризуется двумя главными свойствами – своевременностью и полнотой. Нахождение правильного баланса между ними в каждой конкретной ситуации и является задачей тимлида.

Тимлид должен обеспечить поступление обратной связи к сотруднику. В самом базовом варианте всю обратную связь он пропускает через себя, в более продвинутом – обучает команду ценности обратной связи, способам её дачи и способствует развитию соответствующей культуры.

Можно выделить три типичных подхода к обратной связи:

- **Периодическая.** Привязана к какому-то циклу – регулярных встреч, целеполагания, календарному. С увеличением длины цикла своевременность снижается, но повышается полнота благодаря возможности привлекать к сбору информации большее количество людей.Даже если собирается письменно, должна сопровождаться устным обсуждением на one-to-one.
- **Постоянная.** Даётся сотруднику по ходу работы, без проведения дополнительных ритуалов по её сбору и обработке. Характеризуется повышенной своевременностью и пониженной полнотой – на постоянной основе собирать обратную связь со всех людей, с которыми сотрудник взаимодействовал невозможно. Может быть не привязана к каким-то регулярным встречам, а даваться прямо на месте без отрыва от рабочего процесса.
- **Ситуативная.** Даётся в какой-то определённый момент времени, чаще с привязкой к какому-то событию. Например, после какого-то конфликта в команде. Здесь управление полнотой находится полностью в руках тимлида.

Почему ветка важна?

Для сотрудника:

- Калибровка собственного восприятия реальности.
- Определение направлений для развития.

Для коллег:

- Возможность повлиять на поведение или результаты другого человека.

Для менеджера:

- Корректировка методов работы сотрудника.

Что будет, если её не делать?

- Развитие сотрудника будет происходить хаотично, как следствие медленно. Задача руководителя в том, чтобы его направлять.
- Теряется канал связи между руководителем и сотрудником.

На кого может быть делегирована?

- Дача постоянной обратной связи может быть делегирована на всю команду при развитии соответствующей культуры.
- Сбор регулярной обратной связи может быть делегирован на HR, которые для этого используют инструменты вроде 360 review.

Примеры поведения

Примеры плохого поведения

- Дача обратной связи без запроса
- Даётся только корректирующая обратная связь
- Даётся только позитивная обратная связь
- Отсылается сотруднику в виде отчёта письмом через несколько недель

Примеры хорошего поведения

- Обратная связь даётся только после разрешения сотрудника, которому она предназначена
- Обратная связь сбалансирована, т.е. содержит и зоны роста и позитивные моменты
- Обратная связь адресуется сотруднику в личной беседе
- Обратная связь своевременна, т.е. даётся сразу после наблюдаемого поведения
- Обратная связь конструктивна, даны объяснения, какое поведение правильное/не правильное
- Запрашивать обратную связь у окружающих по своему поведению

Способы прокачки

Практика

- В крупных компаниях часто бывают порталы или сервисы HR-служб для получения/дачи обратной связи
- Ролевые игры, например при отработке навыков ведения интервью. Т.е. каждый участник сможет себя попробовать в роли интервьюера, интервьюируемого и наблюдателя
- Специализированные курсы, например Как правильно давать обратную связь
- Обучение профессиональному коучингу

Консультации

- Чат TeamLead Bootcamp

Теория

Статьи

- Правила дачи обратной связи
- Ещё несколько правил дачи и приёма

Книги

- Пять пороков команды, Патрик Ленсиони
- Фидбек. Получите обратную связь!, Елена Золина и Игорь Манн

Видео

- Инструменты регулярного сбора обратной связи в маленькой команде

Увольнение

Описание

Увольнение – это прекращение трудовых отношений между сотрудником и компанией. Сотрудник может уйти по собственному желанию, по соглашению сторон или в результате расторжения компанией трудового договора. Если сотрудник уходит сам, то роль тимлида состоит в том, чтобы сделать процесс его ухода безболезненным для команды, уменьшить связанные с ним риски и найти замену. Если сотрудника нужно уволить, то роль тимлида расширяется – он тот, кто решение инициирует и исполняет.

Почему ветка важна?

Для подчинённого:

- Увольнение – тяжёлая личная история. Сотруднику важна поддержка руководителя и чёткое объяснение причин решения.

Для команды:

- В каждой команде есть свой микроклимат. Уход одного человека может оказать сильное влияние на всю команду.
- Из-за уменьшения количества рабочих рук или определённых компетенций команда может замедлиться в решении рабочих задач.

Для руководителя:

- Это финальный инструмент для управления производительностью сотрудника.
- Руководитель несёт полную ответственность за нанятых им людей.

Что будет, если её не делать?

- Слишком раннее увольнение может быть вредно для компании, так как найм нового специалиста может стоить очень дорого.
- Слишком позднее увольнение тоже может быть вредно, так как неудовлетворительный перфоманс сотрудника влияет не только на него, но и на всю команду и её результаты.
- Рынок IT довольно тесный. При неправильно проведённом увольнении может пострадать репутация и технический бренд компании.

- Сотрудник, с увольнением которого ты затягиваешь, теряет своё время и энергию впустую, хотя в другой компании или на другой роли мог бы оказаться более полезен.

На кого может быть делегирована?

- На тимлида ниже уровнем.
- В качестве поддерживающей функции – на HR.

Примеры поведения

Примеры плохого поведения

- Сотруднику не давалась негативная обратная связь, он удивлён, узнав об увольнении.
- Не было попыток исправить недостатки сотрудника.
- Не рассматриваются варианты перевода сотрудника в другую команду или роль.
- Руководитель не увольняет сотрудника сам, а полностью перекладывает это на плечи HR.
- Команде не сообщаются причины увольнения.

Примеры хорошего поведения

- Сотруднику давалась обратная связь по недостаткам в его работе.
- Были подготовлены цели и план исправления недостатков.

Способы прокачки

Практика

Увольнения чаще всего единичная история, поэтому о системной прокачке речи не идёт. В этом разделе представлен алгоритм, которым можно пользоваться, если появилась необходимость кого-то уволить.

Когда появилась необходимость уволить сотрудника

- Опиши причины, из-за которых ты планируешь уволить подчинённого. Стоит посмотреть на:
  - Перфоманс сотрудника (успешность выполнения целей, сложность выполняемых задач).
  - Ценность сотрудника (наличие критичных для бизнеса навыков или знаний).
  - Положительный вклад в работу команды (вклад в общий результат, влияние на внутренний климат, других коллег).
  - Вероятность того, что перфоманс сотрудника может подняться.
  - Стоимость поиска замены после увольнения (насколько дорого стоит найм).
  - Стоимость отсутствия человека на проекте между его увольнением и наймом замены.

- Подтверди свою оценку, собрав обратную связь с коллег или проведя анализ перфоманса сотрудника за последнее время.
- Проверь, какие у этого человека мотивационные факторы, и удовлетворяются ли его потребности.
- Подумай, какая работа ему интересна, и может ли компания предложить ему перейти в другую команду или на другую роль.
- Определи, влияет ли перфоманс или поведение сотрудника только на успешность исполнения его роли, либо на всю команду. Ответ на этот вопрос даст понять, нужно ли пытаться исправить ситуацию и разработать с сотрудником план действий, или сразу перейти к увольнению.

Если есть возможность коррекции перфоманса или поведения, то:

- Дай сотруднику чёткую обратную связь, используя подготовленную аргументацию и факты. Удостоверься, что он её услышал и принял.
- Составьте вместе план исправления проблем.
  - Он должен соотноситься с причинами, из-за которых зашла речь про увольнение.
  - Он должен чётко описывать требования к сотруднику.
  - Он должен содержать понятные чекпойнты и критерии приёмки.
  - Он должен быть реалистичным для выполнения в обозначенные сроки.
- Чётко обозначь подчинённому, что если план не будет выполнен, то он будет уволен. Удостоверься, что он это услышал.

Если дело дошло до увольнения:

- Заранее продумай план, как ты выстроишь работу команды без этого человека. Прими решение, как распределится его нагрузка, оцени время на поиск замены, прикинь, какие дела нужно передать.
- Обсуди с HR все финансовые и юридические вопросы. Узнай, сколько денег компания должна сотруднику за отработанные рабочие дни и оставшийся отпуск. Подготовься ответить на любые потенциальные вопросы.
- Определись с HR, готовы ли вы выплатить сотруднику дополнительную компенсацию. Обычно это 1-3 месячных оклада. Эта сумма нужна, чтобы сотрудник мог комфортно найти себе новое место работы, позволяет сделать процесс увольнения более спокойным и сгладить негативное отношение сотрудника к уволившей его компании.
- Реши, когда у сотрудника будет последний рабочий день. Обычно выбирают между увольнением одним днём и сроком на передачу дел от пары дней до двух недель. При решении учитывай следующие факторы:
  - Нужно ли сотруднику передавать кому-то дела.
  - Есть ли риск того, что его присутствие на работе негативно повлияет на команду.
  - Есть ли риск того, что команда негативно воспримет увольнение одним днём.
- Подготовь план разговора. Проиграй возможные варианты развития событий, подготовься к ним.
- Назначь и проведи встречу с сотрудником.

    - Начни разговор сразу с того, что ты принял решение уволить сотрудника, принятое тобой решение финальное, и это не очередная попытка исправить ситуацию. Сделай отсылки к вашему прошлому общению, плану исправления проблем, если он был, обратной связи от тебя и коллег. Отдельно проговори заслуги сотрудника перед компанией и его сильные стороны, объясни, почему они не перевешивают недостатки.
    - Дай время обдумать сказанное. Спроси, есть ли вопросы. Если вопросов нет, перейди к вопросам компенсации.
    - Если на какие-то вопросы ответить не можешь, попроси дать тебе время разобраться.
    - Старайся расставаться мягко, без ссор и скандалов. Мир айти тесен – и если есть возможность расстаться позитивно, это нужно сделать.
    - Предложи себя как контакт для рекомендаций при собеседованиях в другие компании. Уточни, что будешь честным – расскажешь как о сильных сторонах, так и о недостатках.
    - Договоритесь о дальнейших шагах.
- Поговори с командой. В зависимости от ситуации, этот разговор может состояться как до ухода сотрудника, так и после.
    - Объясни причины произошедшего. Подчеркни, что решение не внезапное. Расскажи о том, что произойдёт с его задачами и ответственностью.
    - При необходимости проведи личные встречи с остальными членами команды и дай им больше деталей.

Консультации

- Чат TeamLead Bootcamp

Теория

Статьи

Раскрывают тему:

- Про увольнение глазами сотрудника. Интереснее всего - 600 комментариев с кучей историй
- Алгоритм увольнения и основные принципы
- Куча советов от разных людей по тому, как правильно уволить
- Подробный гайд построения разговора при увольнении

Дополнительные материалы:

- Ещё одна заметка про то, как увольнять
- Нужно ли увольнять good enough кандидатов
- Как составить и использовать performance improvement plan

Собеседования

Описание

Собеседование – процесс, результат которого в первую очередь – принятие решения о найме или отказе соискателю. Оно необходимо для того, чтобы:

- Увеличивать численность команды.
- Подбирать правильных людей, которые соответствуют не только технических требованиям, но и культуре компании.
- Первая точка выстраивания отношений с сотрудником, фиксация ожиданий.

Почему ветка важна?

Грамотное выстраивание процесса собеседований позволяет:

- Уменьшить срок закрытия вакансий.
- Оптимизировать время команд и руководителей.
- Подбирать сильных сотрудников.
- Работать на технический бренд.

Что будет, если этого не делать?

- В команду будут попадать случайные по профессиональному уровню и культуре люди.
- Воронка прохождения собеседования кандидатом будет построена таким образом, что нагрузка будет распределена неоптимально.
- Собеседования будут проходить лучше те, кто хорош в прохождении собеседований, а не непосредственно в работе.

На кого может быть делегирована?

- В зависимости от этапа можно делегировать техническое собеседование техническими специалистам.
- Тимлидам ниже уровнем.
- Определённые этапы могут быть делегированы специалисту из HR.

Примеры поведения

Примеры плохого поведения

- Вопросы на собеседовании задаются несистемно. Невозможно сравнивать кандидатов между собой, фактор случайности увеличивается.
- Соискатель впервые видит членов команды в первый рабочий день. Есть риски, что команда его не примет.
- Соискатель ставится в неравное положение с собеседующими. Коммуникация идёт односторонняя. Из-за этого соискатель сильнее стрессует, что вносит дополнительный случайный фактор в результаты.
- Техническое собеседование идёт неоправданно долго.
- На последних этапах срезается большой процент кандидатов. Процесс не оптимизирован, отказы на последнем этапе слишком дорогие.
- Данные по общению с кандидатом теряются, повторное общение начинается с чистого листа.

Примеры хорошего поведения

- Составлен профиль кандидата. HR понимает, кого набираем не только по техническим скиллам, но и по характеру.
- Есть стандартный список вопросов, собеседующие по ходу встречи оставляют заметки, после неё могут защитить свою позицию.
- Соискателю рассказывают о структуре компании, технологическом стеке, ожиданиях руководителя, особенностях компании.
- Есть анализ воронки.
- Даётся развёрнутый фидбек в случае отказа.
- История общения с кандидатом сохраняется. Её наличие здорово помогает при повторном общении. Например если были просадки по хард-скиллам, то можно поднять и оценить даже конкретные области. Но если человек совсем не подходит по культуре, вряд ли это изменится за полгода.

Способы прокачки

Консультации

- Чат TeamLead Bootcamp

Практика

Как внедрить процесс. Порядок этапов не является каноническим, впрочем как и не обязательно их наличие.

Предусловия

- Пойми бизнес-задачу. Можно ли её закрыть существующими ресурсами?

Профиль кандидата

- Какой у кандидата должен быть уровень? Подойдёт ли вчерашний студент? Есть ли достаточно интересные задачи для синьора с 10 годами опыта?
- Составь вилки по уровням соискателей, если есть возможность, сделай их публичными.
- Составь список мотивационных факторов, которые должны быть у человека. Например он должен любить прекрасные интерфейсы, или у него должна быть тяга к тому, чтобы закапываться в «кишки».
- Скоммуницируй ожидания в явном виде HR'у.

Скрининг HR

Этот этап призван составить первое впечатление кандидата о компании. Он позволяет оценить соответствие кандидата профилю, а интересам кандидата предложение компании. Может проходить по телефону, почте или лично.

Техническое собеседование

Список для проверки

- Составь собственный список вопросов для технического собеседования, привлеки экспертов.
- Необходимо в первую очередь закрыть области, которые действительно требуются на работе.
- Обозначь ожидания из профиля кандидата, релевантные для технической части.
- Для проверки технических скиллов используются как технические решения, так и white board.

Участники

- Собеседование лучше проводить вдвоём, это позволяет получить разные мнения и снизить фактор субъективности.
- Непосредственное участие тимлида необходимо только для первых собеседований, после обучения процесс можно делегировать экспертам.

Расскажи о компании

- Продумай, что ценят соискатели, обозначь ваши преимущества перед другими компаниями.
- Удели достаточно времени на собеседовании ответам на вопросы соискателя.

- Познакомь человека с командой. Можно провести небольшую встречу с непосредственно участниками команды, где можно поговорить про текущую работу, про взаимодействие с разными ролями. Можно провести тестовый день, в который соискатель садится с кем-то из команды, они вместе смотрят проект, обсуждают идеи.

Софт-скиллы

- Ожидания от сотрудника должны совпадать с культурой компании.
- Необходимо заранее подготовить список вопросов.
- На собеседовании необходимо записывать ответы соискателя. В случае успешного прохождения оно может быть использовано как первоначальный материал для личной карточки сотрудника.
- В профиле должен быть список качеств, на которые ты хочешь обратить внимание. Например «умение достигать целей».
- Вопрос или несколько вопросов должен подтверждать или опровергать личное качество. Для «умения достигать цели» это может быть «Расскажи про рабочий или домашний проект, результатом которого гордишься»
- Информацию можно получать при помощи кейсов – гипотетических ситуаций, в которые помещается кандидат.
- Собери рекомендации, если возможно. Контакты для рекомендаций можно попросить у соискателя.
- Храни информацию, используя любую подходящую систему, чтобы вести базу знаний по людям, с которыми ты общался. После каждого этапа обновляй данные.

Обратная связь

- После проведения отбора сообщите кандидату о результатах. Игнорирование соискателя - некорректная практика, которая может повлечь за собой потери репутации.
- Отказ кандидату может негативно сказываться на репутации компании, поэтому необходимо максимально корректно сформулировать реальные причины.
- По возможности посоветуй соискателю материалы, при помощи которых он мог бы закрыть свои пробелы.
- В случае отказа по личным качествам хорошей тактикой является «дело не в тебе, а в нас». Тем более, что отказав кандидату, мы должны верить, что команде по какой-либо причине будет некомфортно работать с ним.
- Полезно получить обратную связь от кандидата, которую можно использовать для улучшения процесса. Собирать её надо уже после того, как сделан оффер или отказ, так как на этом этапе она будет максимально непредвзятой.

Ретро

- Проанализируй конверсию прохождения этапов. Этапы с самой низкой конверсией надо отправить в начало воронки. Провалидируй необходимость этапов со слишком большой конверсией.
- Проведи разбор обратной связи, собранной от кандидатов.
- Разбери инциденты: выходили из обычных таймингов, кандидат давал негативную обратную связь.

Способы прокачки

Консультации

- Чат TeamLead Bootcamp

Теория

Книги

- Брать или не брать? или Как собеседовать разработчика, Константин Борисов
- Кто. Решите вашу проблему номер один, Джефф Смарт и Рэнди Стрит

Статьи

- Опыт проведения собеседований разных компаний
- Про процесс подбора в целом
- An inside look at Google's hiring and onboarding processes
- Как собеседует Google: чему быть, чего не миновать
- Как пройти собеседование в Google: советы по подготовке
- Собеседование на junior позицию. Антипаттерны собеседующих
- Пять способов отпугнуть соискателя
- Я провёл сто собеседований, отказал сотне людей — и только потом научился собеседовать
- Краткий гайд о том, как нанимать нормальных людей

Onboarding

Описание

Процесс адаптации новичка в компании. Он может идти параллельно работе или быть выделенным этапом в жизни сотрудника. Основные цели онбординга:

- Вывести человека на уровень перфоманса сравнимый с командой.
- Нивелировать разницу в скиллах между новичком и командой.
- Познакомить сотрудника с процессами, подходами, отделами и людьми в компании. Дать представление о внутренней кухне.

Почему ветка важна?

Этап адаптации – обычно первый этап в жизни человека в компании, первые впечатления, которые во многом определят дальнейший путь в организации. Грамотный онбординг может дать:

- Новичок начнёт приносить пользу как можно раньше.
- Будет получен хороший толчок для дальнейшего развития.
- Больше контекста – больше самостоятельности в дальнейшем.

Что будет, если её не делать?

Новый коллектив – это стресс. Если отнестись к новичку без достаточного внимания, он будет дольше вкатываться в команду, а как следствие позже начнёт приносить реальную пользу. В некоторых случаях это может вызвать негатив. Отсутствие или непродуманный онбординг может привести к тому, что:

- Новичок получит впечатление об организации как о бюрократической в случае, если онбординг будет работать плохо или приносить мало пользы.
- В случае, если онбординг совсем не проводится, это может сложить впечатление об организации как о хаотично управляемой.
- По его окончании останется разница в контексте или скиллах между сотрудником и другими членами команды.
- Не будет раскрыт потенциал новичка, по уровню пользы он будет ниже «старичков» с аналогичным уровнем скиллов.

На кого может быть делегирована?

- Персональный наставник
- Специалист из HR
- Тимлиды ниже уровнем
- Отдельная команда

Примеры поведения

Примеры плохого поведения

- Нет наставника или "Buddy"
- Нет вводных курсов
- Нет чек-листов "Доступы новому сотруднику" , "Список ПО для нового сотрудника"
- Не накапливать знания в FAQ

Примеры хорошего поведения

- Наставник знакомит с техническим стеком команды, помогает разобраться в текущем проекте, даёт советы по настройке оборудования
- В крупных компаниях, как правило, у нового сотрудника помимо наставника есть "Buddy" (англ. приятель, дружище) - сотрудник компании, хорошо знакомый с её ключевыми сотрудниками, а также порядками и процессами
- Встреча руководителя компании с новыми сотрудниками для их знакомства с ценностями и миссией компании
- Обновление FAQ через обучение: сталкиваясь с проблемами сотрудник учится их решению и фиксирует их
- Адаптационный курс / FAQ по сервисам компании в очной или дистанционной форме - "куда нести больничный?", "как заказать справку?", "где получить пропуск?"
- Сувениры с символикой компании позволяют новому сотруднику быстрее почувствовать себя частью команды

Способы прокачки

Практика

- Выращивайте наставников, поощряйте волонтерство

- Развивайте сообщество "Buddy", чтобы сотрудники разных команд или отделов могли вместе помогать новым сотрудникам
- Опросите текущих сотрудников, с какими трудностями они столкнулись после выхода на работу и как их решили. Составьте FAQ на основе этой информации для нового сотрудника
- Создавайте чек-листы, презентации, видео или любую другой контент, который поможет в освоении информации

Примеры чек-листов для новых сотрудников

- Onboarding checklist example

Консультации

- Чат TeamLead Bootcamp

Теория

Статьи

- Гайд с вопросами для создания онбординга
- Подробный гайд по онбордингу
- Buddy program: 5 компонентов наставничества

Профиль кандидата

Описание

Профиль кандидата – набор требований к профессиональным навыкам и личным качествам, необходимый для открытой позиции. Оптимально, если дополняется списком задач на ближайшее время, а также техническим стеком, с которым предстоит работать. Также необходимо понимать зарплатную вилку, которую может предложить компания.

Почему ветка важна?

- Участники процесса подбора понимают целевой образ соискателя.
- Полезна для «чистоты эксперимента». Мы заранее коммитимся на некоторые требования к кандидату, после чего сравниваем реальных соискателей с ним, а не с собственными субъективными ощущениями.
- Грамотно составленный профиль улучшает процесс подбора кандидатов.

Что будет, если этого не делать?

- Возможно принятие спонтанных эмоциональных решений.
- HR не сможет объяснить соискателю в чем будет заключаться работа.
- Сложно составить обратную связь для отказа кандидату, ведь неясно, чем он не подошёл.

На кого может быть делегирована?

- Тимлид ниже уровнем
- Специалист из HR

Примеры поведения

Примеры плохого поведения

- Профиля нет.
- В профиль входят исключительно технические скиллы.
- Профиль не помогает отвечать на вопросы соискателей по предложению.

- Профиль не отражает никаких особенностей, «за все хорошее, против всего плохого».

Примеры хорошего поведения

- Прочитав профиль, любой участник процесса собеседований понимает, кого нужно нанять не только в плане технических скиллов, но и по характеру.
- К профилю кандидата обращаются для определения проходит ли человек тот или иной этап собеседования.

Способы прокачки

Консультации

- Чат TeamLead Bootcamp

Практика

В первую очередь его необходимо составить. Что туда можно и нужно включить:

- Технические скиллы. Не все подряд, а именно те, что необходимы на конкретной работе. Надо расписать подробнее, чем «мидл+», потому что у всех, даже внутри компании, не может быть разным. Их можно определить совместно с техническими экспертами в этой области.
- Софт-скиллы. Насколько часто участникам вашей команды необходимо коммуницировать с внешними отделами? Какие личные качества сотрудников цените лично вы? Какая культура в вашей компании, какую вы выращиваете в команде?
- Зарплатная вилка. Это может быть ключевым критерием, по которому будет идти подбор. Её необходимо согласовать с человеком, принимающим такие решения.
- Зона ответственности и технологический стек. Предстоит работать в монолите, или команда пишет новый код на микросервисах? Везде используется Rx, или переходим на корутины? Важно явно скоммуницировать, над чем предстоит работать, и при помощи каких инструментов необходимо решать задачи. Это важно в явном виде сформулировать в случае, если вопрос об этом зашёл на HR-собеседовании, или на техническом с представителями других команд.
- Мотивация. Если вы разрабатываете платформенную часть, то человек с желанием делать продукт для миллионов людей будет у вас страдать.

Улучшать профиль можно при помощи проведения ретроспектив. Возможные фокусирующие вопросы:

- Насколько кандидат понял зону ответственности и стек при первом контакте?
- Насколько уверенно мы принимали решения об отказе кандидату?

Теория

Статьи

- 7 Steps To Develop An Ideal Candidate Profile
- 4 Steps to Creating a Candidate Profile
- Составление профиля должности специалиста

Тестовый период

Описание

Тестовый период необходим для того, чтобы окончательно понять, насколько сотрудник подходит компании и наоборот. Сотруднику ставят цели, а по окончанию испытательного проводят встречу с итогами. Также обычно в него входит онбординг.

Почему ветка важна?

Тестовый период – последний рубеж отсеивания неподходящих кандидатов. По ТК РФ он составляет не более 3 месяцев (не более 6 месяцев для руководителей), за которые новичок должен освоиться, притереться к команде, получить контекст и выйти на свою крейсерскую скорость.

Что будет, если её не делать?

Тестовый период – время, когда с неподходящим сотрудником можно расстаться относительно безболезненно. Что может пойти не так:

- Новичок делал «игрушечные» или нестандартные задачи, оказалось, что стандартные он делал плохо. В реальных боевых задачах после испытательного он показывает себя очень плохо.
- По ходу испытательного срока, новичок держится в стороне от команды. После испытательного оказывается, что команда его не принимает.
- На испытательном сроке человек «ходит по струнке», а по окончанию показывает своё истинное лицо.

На кого может быть делегирована?

- Специалист из HR
- Тимлиды ниже уровнем

Примеры поведения

Примеры плохого поведения

- Не обговаривать критерии успешного завершения испытательного срока
- Не проводить промежуточные встречи для инспекции прогресса по удовлетворению критериев испытательного срока
- Повышать оклад или выплачивать бонусы после прохождения испытательного срока - может заставить человека притворяться ради вознаграждения
- Поручать только "тестовые задачи" в песочнице - отношение к ним и их решение не будет показательным

Примеры хорошего поведения

- Чётко определить критерии успешного прохождения испытательного срока и фиксировать их на каком-либо носители
- Проводить промежуточные 1-1 встречи для инспекции прогресса по удовлетворению критериев испытательного срока и выявлению возможных проблем
- Максимально погружать сотрудника в контекст "боевых" задач

Способы прокачки

Практика

Первое что необходимо сделать – это составить цели на тестовый период. Их можно сформировать на основе собеседований. Из результатов технической секции берём «белые пятна», а из собеседований по софт-скиллам риски, связанные с личными качествами. С целями тоже не стоит торопиться, потому что их возможно понадобится резко менять. Их можно составить через несколько недель после прихода сотрудника, когда он уже успел немного пожить в команде и показал себя в задачах. Напарник или наставник могут помочь с составлением такого списка.

Второй по важности пункт – прозрачность и коммуникация. Необходимо собирать обратную связь от коллег, мониторить работу новичка, а также постоянно давать обратную связь на регулярных встречах. В случае, если что-то идёт не так, и это в конечном счёте может привести к увольнению, необходимо об этом говорить, подчёркивая серьёзность проблемы. Плохая история, когда человек не понимает, к чему могут привести незакрытые ожидания.

Испытательный срок может заканчиваться раньше как с положительным, так и с отрицательным результатом. Окончанием испытательного срока бывает встреча, на которой подводятся итоги и даётся резюмирующая обратная связь от менеджера и коллег.

Консультации

- Чат TeamLead Bootcamp

Теория

- Wikipedia: Испытательный срок
- ТК РФ Статья 70. Испытание при приёме на работу

Мотивация

Описание

Мотивация – это эмоциональное и психологическое состояние сотрудника, влияющее на достижение им результатов работы. Если мотивация работать низкая – результатов нет. Если высокая – результаты соответствующие.

Есть много разных теорий, которые декомпозируют работу с мотивацией: двухфакторная теория Герцберга, теория потребностей Маслоу, теория приобретённых потребностей МакКлелланда и много других. Каждая имеет право на жизнь и подкреплена существенными психологическими исследованиями. Тимлиду важно понимать принципы, на которых построены эти теории, и уметь определять основные мотивационные факторы своих сотрудников: деньги, признание, власть, технический вызов.

Почему ветка важна?

Мотивация – это главный инструмент, который позволяет тимлиду выполнять свою работу. Если команда не мотивирована, результата не будет.

Во многих случаях вся работа с мотивацией может заканчиваться закрытием гигиенических факторов вроде комфортного рабочего пространства, достаточно производительного железа и адекватной команды, и использованием финансовых инструментов вроде опционов или зарплаты выше рыночной вилки. Тем не менее, для повышения индивидуальных результатов каждого сотрудника и его удержания в компании важно понимать его личные мотивационные факторы и работать непосредственно с ними. Кому-то важнее постоянный технический вызов, а кому-то – регулярное признание заслуг от команды или пользователей.

Что будет, если её не делать?

- Снижаются результаты личные и командные
- Повышается шанс потери сотрудника, что является дорогой для компании проблемой. Стоимость замены складывается из нескольких факторов:
  - Работа специалиста по рекрутменту
  - Потери времени на проведение собеседований
  - Потеря производительности команды на время поиска замены
  - Слабая производительность нового сотрудника на период онбординга

    - Пониженная производительность сотрудника, выступающего ментором

На кого может быть делегирована?

Мотивация не может быть делегирована и является непосредственной обязанностью тимлида. Разработка программ по удержанию может быть передана в различные функции в HR:

- T&D: Training and Development
- C&B: Compensations and Bonuses

Примеры хорошего поведения

- Повышение уровня компенсации до "среднего уровня по рынку" (а лучше ещё +10-15%), как мера улучшения гигиенических факторов: человек перестаёт думать о деньгах и фокусируется на поставленных задачах
- Пересмотр ("индексация") уровня компенсации каждые 12-18 месяцев на основе оценки других членов команды, клиентов, заказчиков, партнёров
- Спонтанная выдача бонусов/нематериальное поощрение в виде грамот от руководства, конференций, курсов, книг является подбадривающими факторами
- Публичное или в режиме 1-1 словесное признание достижений
- Повышение уровня сложности задач и уровня автономности в их решении позволяют оставаться сотруднику в "потоке"

Примеры плохого поведения

- Повышение уровня компенсации, как мера увеличения эффективности: такая схема работает только для простого механического труда
- Условия "сделай задачу A - получишь ценность B" скорее фокусирует на вознаграждении, а не на качественном решении задачи
- Контроль времени нахождения на рабочем месте: ограничивает возможность креативного подхода к решению задач

Способы прокачки

Практика

- Способы определения мотивационных факторов:
    - Тест 10K - более конкретный и детализированный, но фокусирующий на ограниченное количество вариантов ответов

- Мотивационные карты - более общий, даёт возможность на свою интерпретацию каждого мотиватора

Консультации

- Чат TeamLead Bootcamp

Теория

Виды мотивации

Мотивация	Внутренняя	Внешняя
Позитивная	🥕 *внутри*: интерес, любопытство, значимая цель, ...	🥕 *впереди*: одобрение окружающих, бонусы, соперничество, ...
Негативная	*гнилая* 🥕 *внутри*: вина, месть, жажда власти, ...	🥕 *сзади*: страх, желание избежать наказания или лишения, ...

Лучшей мотивацией считается позитивная внутренняя, она является независимой от внешних факторов и не нуждается в искусственной "стимуляции" извне.

Инструменты мотивации делятся на два вида:

- Материальные: деньги, подарки, награды.
- Нематериальные: программы обучения, интересные задачи, общественное признание.

Пирамида потребностей А. Маслоу

Широкую известность получила теория иерархии потребностей А. Маслоу, современного американского экономиста. Согласно этой теории потребности человека развиваются от низших к высшим, и индивидуум должен сперва удовлетворить потребности низшего порядка для того, чтобы возникли потребности высшего уровня.

Иерархия потребностей по А. Маслоу включает следующие компоненты:

- Физиологические (голод, жажда, секс ...);
- Потребность в безопасности (защита от боли, гнева, страха...);
- Потребность социальная (любовь, семья, друзья, общение);
- Потребность в самоутверждении (самоуважение, престиж, карьера, успех);
- Потребность в самоактуализации (реализации способностей, понимании, осмыслении и т.д.).

Потребности образуют пять уровней, каждый из которых может служить в качестве мотивации лишь после удовлетворения потребности, находящейся на более низкой ступени. То есть в первую очередь человек стремится удовлетворить наиболее важную потребность. Только после

удовлетворения первой потребности человек начинает думать о другой. Таким образом голодный человек не будет думать о безопасности или уважении или признании в обществе пока он не удовлетворит свою потребность в пище.

А. Маслоу описал данные уровни в своей книги "Мотивация и личность" и значительную её часть посвятил описанию случаев, когда поведение людей не подчиняется данной парадигме.

Теория МакКлелланда

Теория делает основной упор на потребности высших уровней: стремление к власти, успеху и причастности (признанию и принадлежности). Но эти потребности не расположены иерархически и не исключают друг друга. В иерархии Маслоу стремление к власти и успеху находятся где-то между потребностями в уважении и в самовыражении. Однако обе этих потребности могут стать очень сильными побуждающими факторами. Стремление к власти заставляет не только не двигаться вверх по карьерной лестнице, но и при правильном понимании социальной роли лидера понимать и пропагандировать цели организации, брать инициативу на себя, искать способы и средства достижения цели. Стремление к успеху тоже заставляет человека брать ответственность на себя, искать способы достижения цели. Мотивация на основе потребности в причастности означает, что сотрудники с потребностью в причастности будут очень хорошо себя чувствовать на работе с большим количеством контактов.

Теория "X" и "Y" Д. МакГрегора

Дугласом МакГрегором были предложены теория "X" и теория "Y", рассматривающие мотивацию человека с двух противоположных сторон. Теория "X" допускает, что большинство людей не заинтересованы в ответственности и что люди работают либо только из-за денег либо из страха перед некими угрозами. Однако, создав теорию "X", МакГрегор пришёл к выводу, что такое понимание человеческой природы не соответствует действительности, а менеджмент, построенных на этом подходе не отвечает современным потребностям. Таким образом, была создана теория "Y", основным постулатом которой является то, что люди не ленивы и не безответственны. Эта теория доказывает, что люди могут быть самоуправляемыми и творческими в работе при правильной мотивации. Следует особо подчеркнуть, что теория "X" и теория "Y" не являются взаимоисключающими противоположностями. Наоборот, их автор считает; что большинство людей имеет потенциал для того, чтобы быть зрелым и сознательным, таким образом, существует разница между позициями и поведением. Теории "X" и "Y" описывают позиции и склонности людей. Руководителю следует придерживаться теории "Y", но также следует помнить и о теории "X", так как с некоторым людьми необходимо некоторое время обращаться согласно теории "X", чтобы помочь им самореализоваться и перейти в категорию "Y".

	Индивид "X"	**Индивид "Y"**
Отношение к работе	Избегает её любыми способами	Считает её естественной, как отдых или приём пищи

	Индивид "X"	Индивид "Y"
Ответственность	Не готов брать на себя	Готов брать на себя
Контроль	Нуждается в контроле менеджмента для выполнения работы	Наличие контроля скорее влияет негативно на желание работать, может сам решать поставленные задачи

Двухфакторная модель Ф. Герцберга

В основе двухфакторной теории Ф. Герцберга лежат две большие категории потребностей: гигиенические факторы и мотивирующие факторы. Гигиенические факторы связаны с окружающей средой, в которой осуществляется работа, а мотивирующие — с характером работы.

Герцберг назвал первую категорию потребностей гигиеническими, употребив медицинское значение слова "гигиена" (предупреждение), так как, по его мнению, эти факторы описывают окружение сотрудника и обслуживают первичные функции, предупреждая неудовлетворённость работой. Вторую категорию факторов Герцберг назвал мотивирующими или способствующими, так как они побуждают сотрудников к лучшему исполнению.

Гигиенические факторы	Мотивирующие факторы
Политика организации и руководства	Успех
Условия работы	Продвижение по службе
Заработная плата, социальный статус	Признание и одобрение результатов работы
Межличностные отношения с начальником, коллегами и подчинёнными	Высокая степень ответственности
Степень непосредственного контроля за работой	Возможность творческого и профессионального роста

Следует обратить внимание на то, что Герцберг сделал парадоксальный вывод о том, что **заработная плата не является мотивирующим фактором**. Действительно, в таблице заработная плата находится в категории факторов, приводящих к удовлетворённости или неудовлетворённости работой.

Поток М. Чиксентмихайи

М. Чиксентмихайи утверждает, что "состояние потока" или "условия переживания радости" относятся к внутренней позитивной мотивации и состоят из 8 компонентов:

> В результате наших исследований было выделено восемь основных компонентов переживания радости. Когда люди размышляют о своих чувствах в особенно позитивные моменты, они, как правило, упоминают не менее одного из них, а чаще — все восемь. Во-первых, задача, которую

> ставит себе человек, `должна быть для него посильной`. Во-вторых, он должен иметь `возможность сосредоточиться`.

> В-третьих и в-четвёртых, концентрация, как правило, становится возможной потому, что задача позволяет `чётко сформулировать цели` и `немедленно получить обратную связь`. В-пятых, в процессе деятельности `увлечённость субъекта` настолько высока, что он забывает о повседневных тревогах и проблемах.

> В-шестых, занятия, приносящие радость, позволяют человеку `ощущать контроль над своими действиями`. Седьмая особенность этого состояния заключается в том, что `осознание своего Я` в момент совершения действия как будто исчезает, зато после окончания потокового эпизода оно становится сильнее, чем раньше. Наконец, `изменяется восприятие течения времени`: часы превращаются в минуты, а минуты могут растягиваться в часы. Сочетание всех этих составляющих порождает чувство настолько глубокой радости, что люди не жалеют сил, чтобы снова и снова испытывать её.

> *Чиксентмихайи М. Поток. Психология оптимального переживания*.

Драйв Д. Пинка

Подход Д. Пинка построен на основе более ранней теории М. Чиксентмихайи, упрощает её для понимания и сводит к 3 компонентам:

- **Автономность**. Человек или команда должны иметь возможность сами принимать решения. Нужно, чтобы каждый понимал свою ответственность за то, что он делает. В свою очередь тимлид должен разрешить человеку ошибаться в допустимых пределах, если что-то пойдёт не так.
- **Мотивирующая цель**. Каждый из членов команды должен понимать общую цель и осознавать, какой вклад несёт его часть работы. Если этого нет, то люди, как правило, приходят на работу за зарплатой. Они знают, что сделают свой кусочек, перекинут "через забор" на следующую стадию и пойдут домой.
- **Мастерство**. То, что позволяет людям проявить свои лучшие качества, узнать что-то новое, в чём-то быть лучшим.

Как раз все эти факторы мотивируют авторов наполнять информацией данный ресурс.

Книги

- Маслоу А. Мотивация и личность
- Чиксентмихайи М. Поток. Психология оптимального переживания
- Пинк Д. Драйв. Что на самом деле нас мотивирует
- Иванова С. Мотивация на 100%

- Бланшар К. Киты. Выше и лучше, или уроки мотивации, вдохновения и определения целей
- Том ДеМарко, Тимоти Листер. Человеческий фактор. Успешные проекты и команды
- Сьюзен Фаулер. Почему они не работают? Новый взгляд на мотивацию сотрудников

Видео

- Драйв! Что на самом деле нас мотивирует
- Обзор и объяснение терминов: стимул, мотив, потребность. Обзор теорий мотивации Неделя 1, 2
- Анна Обухова. Хочу/буду и деньги. Мотивация Agile команды

One-to-one

Описание

One-to-one – это регулярные встречи менеджера и его подчинённого, на которых обсуждаются те вопросы, которые нельзя обсудить в группе большего размера. Будучи правильно построенными, эти встречи должны решать задачи как менеджера, так и подчинённого. Они служат для:

- Построения доверия и тесной связи между менеджером и подчинённым.
- Обсуждения вопросов, влияющих на перфоманс сотрудника.
- Постоянного прояснения и выравнивания целей сотрудника, команды и компании.

Почему ветка важна?

Для подчинённого:

- Знает, что у него гарантированно есть время и место для обсуждения сложных вопросов.

Для менеджера:

- Глубже понимает мотивацию своего сотрудника.
- Получает уверенность, что никто в команде не остаётся обделённым вниманием.
- Быстрее выявляет возможности и проблемы и может на них среагировать.
- Корректирует цели и ожидания от сотрудника, помогает в их достижении.
- Показывает сотрудникам, что они важны для него.

Что будет, если её не делать?

- Уменьшится доверие и связь между менеджером и подчинённым.
  - Люди, которым сложно обсуждать вопросы денег, развития и мотивации, будут замалчивать свои проблемы до того момента, когда исправить что-то будет уже невозможно.
  - Менеджер потеряет ценный источник информации о состоянии проекта, команды, других сотрудниках.
  - Перфомансом и развитием сотрудника станет сложнее управлять.
  - Невовлеченный руководитель демотивирует многих.
- Общение менеджера и сотрудника сведётся к обсуждению операционных рабочих вопросов.
  - Уменьшится количество оперативной обратной связи в обе стороны.
  - Развитие и карьера сотрудника могут развиваться неоптимально без видения общей картины.

- Менеджер может непроизвольно уделять больше времени и внимания отдельным участникам команды.
  - Демотивационный фактор появления любимчиков.

На кого может быть делегирована?

- На тимлидов ниже уровнем.

Примеры поведения

Примеры плохого поведения

- Встречи one-to-one не проводятся вообще или проводятся реже раза в месяц.
- Встречи проводятся нерегулярно и отсутствует отслеживание даты последней из встреч для каждого сотрудника.
- One-to-one состоит из пустых разговоров на отвлечённые темы или обсуждения не приватных рабочих вопросов.
- Менеджер говорит больше, чем сотрудник.
- Сотрудник даёт односложные ответы и не вовлечён в дискуссию.
- Происходит постоянное хождение по кругу – часто обсуждается одна и та же тема без видимого прогресса по ней.
- По обсуждаемым вопросам не фиксируются конкретные action points.
- Встречи заканчиваются сильно раньше запланированного времени.
- Встречи постоянно отменяются или переносятся.
- Сотрудник ищет повода избегать встреч.
- Отсутствует историчность, каждая новая встреча проводится с чистого листа.

Примеры хорошего поведения

- Встречи проводятся чаще раза в месяц и они регулярны.
- Менеджер знает, что для его сотрудников важно, что их мотивирует и демотивирует.
- На встречах обсуждаются вопросы, которые важны подчинённым, а не руководителю.
- По каждой встрече менеджер ведёт заметки того, что обсуждалось, фиксирует все договорённости.
- Оба участника задают неловкие вопросы, вскрывающие сложные и важные темы.
- Менеджер на встрече больше слушает, чем говорит.
- Открыто обсуждаются вопросы карьерного роста и перспектив.
- Оба участника заранее готовятся ко встрече.

Способы прокачки

Консультации

- Чат TeamLead Bootcamp

Практика

Как внедрить процесс

- Подготовь личную карточку для каждого сотрудника, в которой будут храниться:
  - Мотивационные факторы
  - История договорённостей
  - Важные факты
  - Перечень достижений и провалов
  - История встреч
- С каждым сотрудником проведи первую встречу, на которой обозначь цели встреч и договорись о правилах игры:
  - Как часто будут проходить встречи и какой они будут продолжительности. Хотя в определении присутствует указание на “регулярность” – это не значит, что one-to-one нельзя проводить вне графика. Если возникает какой-то срочный вопрос или появляется возможность дать обратную связь раньше – это нужно делать.
  - Какую пользу ожидаем от встреч, какие конкретные вещи и действия ожидаем на выходе. Важно, чтобы встречи не превращались в площадку для обсуждения последних новостей и весёлого трёпа.
  - Где и в каком формате проводятся встречи. Это может быть ограниченная календарём встреча в переговорке, совместный поход за кофе, обед в столовой.
- Запланируй регулярные встречи с каждым сотрудником, назначь их в календарь. Перед каждым one-to-one выдели себе отдельное время на подготовку.
- Подготовь список общих открытых вопросов, которые помогут начать дискуссию, раскачать сотрудника, перейти к обсуждению личных вопросов.
- Начни проводить регулярные встречи с каждым сотрудником. На этих встречах проходи по списку вопросов, составленному заранее, которые помогают вам лучше узнать друг друга, выровняться по пониманию целей и ожиданиям.
  - Во время проведения встречи придерживайся правила того, что менеджер 10% времени говорит, а 90% слушает.
  - После встречи пришли сотруднику follow-up с кратким содержанием встречи и вашим договорённостям.
  - Обнови карточку сотрудника с учётом новой информации.
  - Запроси обратную связь по тому, как прошла встреча.

Как улучшать процесс

- Перед каждой встречей проси сотрудника сформулировать повестку и прислать её. Со своей стороны сделай то же самое – и на самой встрече придерживайтесь получившейся агенды.
- Логируй основные моменты обсуждений в блокноте или ноутбуке. Отдельно отмечай все договорённости и сроки по ним.
- Выстрой удобную систему хранения всех логов встреч. Потом она будет использоваться для анализа ваших договорённостей, статуса мотивации сотрудника, вещей, которые его волнуют.
- В конце встречи подбивай краткий итог того, что вы обсудили и о чем договорились.
- Раз в несколько месяцев перетряхивай свою систему:
  - Собирай обратную связь по своим one-to-one. На её основе пересматривай свой подход.
  - Проходи по всем карточкам и проверяй актуальность информации в них.

Теория

Статьи

Раскрывают тему:

- 100 вопросов, которые можно использовать на one-to-one
- Про то, почему на каждом one-to-one важно задавать неловкие вопросы
- Про подготовку и проведение one-to-one с несколькими кейсами
- Про конкретные направления разговора, которые могут зайти в one-to-one, их смысл и вопросы по теме
- Признаки плохих one-to-one и способы их исправить

Дополнительные материалы:

- Организация системы логирования one-to-one с помощью блокнотов
- Исследование вопросов для one-to-one, которые вскрывают больше всего неожиданных фактов
- Как задавать конкретные открытые вопросы
- Как сделать one-to-one полезным со стороны сотрудника
- Мысли о том, почему one-to-one важен, и какие вопросы на нем можно задавать
- Про три типа one-to-one и модели поведения и использования информации каждого из них
- Про важность прозрачность и агенды на one-to-one
- Как тимлид в Spotify трансформировал one-to-one из бездушной встречи в полезный инструмент
- Ещё один чек-лист для проведения one-to-one
- Список вопросов для one-to-one

Книги

- Behind Closed Doors: Secrets of Great Management, Johanna Rothman, Esther Derby, 2005

Ассессмент

Описание

Ассессмент – процесс оценки навыков сотрудника, часто проходит с некоторой периодичностью.

Тесты и экзамены – необходимые элементы обучения и развития, они обычно дают ответ, насколько хорошо усвоен материал. В работе в явном виде экзаменов обычно не проводят, но проверка прогресса по индивидуальному плану развития может осуществляться при помощи целей со строго сформулированными критериями их закрытия.

Почему ветка важна?

Для руководителя:

- Провести срез и проверить соответствие занимаемой/желаемой должности
- Оценить прогресс сотрудника
- Дать оценку как хард, так и софт скиллам, а также выработать план на следующий период

Для сотрудника:

- Получить обратную связь от коллег за некоторый период
- Возможность расти в карьерном и материальном плане

Что будет, если её не делать?

Если нет среза и оценки, непонятно насколько человек продвинулся. В этом случае оценка руководителем и сотрудником могут отличаться. В конечном счёте это может привести к:

- Субъективная оценка развития руководителем
  - Любая оценка априори субъективна, но в этом случае она может быть основана больше на ощущениях и меньше на фактах. Сложно вспомнить события с участием сотрудника за весь период.
- Нет точных целей – нет развития. Даже если направление движения есть, отсутствие цели может привести к стагнации, т.к. неясно к чему стремиться, «ну вроде развиваюсь».

На кого может быть делегирована?

Помощи в оценке сотрудников можно ожидать от отдела HR, а также на тимлидов ниже уровнем

Теория

Книги

- Работа рулит! Почему большинство людей в мире хотят работать именно в Google, Ласло Бок
- Измеряйте самое важное. Как Google, Intel и другие компании добиваются роста с помощью OKR, Джон Дорр

Карьерная линейка

Описание

Карьерная линейка – набор грейдов, присваиваемых сотрудникам по достижению определённого уровня компетенций и пользы для компании. Каждый грейд подкреплён описанием и уровнем компетенций, то есть для каждой должности присутствует должностная инструкция с описанием трудовых функций и действий.

Почему ветка важна?

Для руководителя:

- Продвижение по карьерной лестнице – один из способов материальной и нематериальной мотивации.
- Позволяет компании контролировать и прогнозировать ФОТ.
- Для руководителей других команд в организации это позволяет получить общее представление о возможностях сотрудника.
- Позволяет описать требования к сотрудникам каждой должности в рамках компании.
- Позволяет договориться о требованиях к сотрудниками одной должности разных команд в рамках компании.

Для сотрудника:

- Признание со стороны коллег и руководителя.
- Выходя на рынок, человек с более высокой должностью имеет возможность получить лучше предложение и не только в материальном плане.
- Понимание требований, которые предъявляются к сотруднику на определённой должности. Как следствие, сотрудник может определить, что необходимо выполнить для перехода к следующему грейду.

Что будет, если её не делать?

Отсутствие понятной карьерной линейки приводит к размыванию должностей, а для сотрудников теряется один из мотивационных факторов. Может проявляться следующим образом:

- Сотрудник хочет, чтобы его оценивали как старшего разработчика, но ему не могут ответить что для этого надо.

- У сотрудника может возникнуть ощущение, что он топчется на месте.
- Сотрудник не понимает, какие функции входят в обязанности текущей или следующей должности.

На кого может быть делегирована?

Карьерная линейка обычно едина по всей компании и для её изменений требуется участие руководителя отдела, CTO, CEO, HR. Составление должностных инструкций можно делегировать главным специалистам, тимлидам. Для принятия инструкции по всей компании необходимо согласование со всеми заинтересованными сторонами: тимлидами/главными специалистами смежных команд, руководителями отделов. Добавление новых обязательных в рамках существующего грейда может стать демотивирующим фактором для сотрудников, необходимо уделить таким ситуациям больше внимания. При проработке должностных инструкций обязательно необходима фокус-группа, которая сможет предоставить обратную связь по корректности, прозрачности и понятности формулировок в описании грейда.

Способы прокачки

Профессиональные стандарты

Профстандарт Программист - полезен с точки зрения структуры описания и наполнения

Полезные статьи

Как оценивать работу программистов: примеры системы грейдов в компаниях

Управление компетенциями

Описание

У команды разработки должны присутствовать все требуемые для выполнения рабочих задач компетенции. Под ними подразумеваются как широкие области: фронтенд, мобильная разработка, бэкенд, так и знание специфических технологий или конкретных участков бизнес-логики. Задача тимлида заключается в том, чтобы обеспечить наличие в команде всех этих компетенций и закрыть связанные с их потерей риски.

По сути управление компетенциями на уровне тимлида укладывается в несколько конкретных задач:

- Построение карты компетенций команды и приведение её в порядок через найм или развитие скиллов у существующих членов команды
- Повышение bus factor
- Привлечение компетенций в помощь команде со стороны – из других команд или аутсорса

Почему ветка важна?

Для менеджера:

- Что найм, что развитие компетенций внутри – это долгий и сложный процесс без гарантированного результата. Тимлид должен думать наперёд о том, какие знания и навыки могут понадобиться команде, чтобы работа не остановилась в критический момент.
- При высоком bus factor потеря любого сотрудника не будет для команды критической.

Для сотрудника:

- Когда у тимлида присутствует ясная картина требуемых компетенций, упрощается написание личных планов развития – там присутствуют не случайно выбранные технологии, а те, которые действительно понадобятся на практике.
- Если по всем вопросам сотрудника может подменить кто-то другой из команды, ему гораздо проще уйти в отпуск или заболеть.

Что будет, если её не делать?

- Будет расти количество кода, в котором разбирается либо никто, либо малое количество человек. Вносить туда изменения будет сложно и дорого.
- Повышается стоимость потери отдельных сотрудников.
- Команда не сможет полноценно автономно работать над своими задачами и будет зависеть от внешних ресурсов.

На кого может быть делегирована?

- На тимлида ниже уровнем, либо на ответственного члена команды.

Способы прокачки

Требуемые навыки

- Найм
- Удержание
- Оценка
- Развитие

Теория

Статьи

Матрица компетенций как инструмент тимлида: https://habr.com/ru/company/oleg-bunin/blog/437772/ Про bus factor: https://medium.com/@lananovikova/%D0%BA%D0%B0%D0%BA-%D1%81%D1%87%D0%B8%D1%82%D0%B0%D1%82%D1%8C-%D0%B2%D1%8B%D1%8F%D0%B2%D0%BB%D1%8F%D1%82%D1%8C-bus-factor-%D0%BD%D0%B0-%D0%BF%D1%80%D0%BE%D0%B5%D0%BA%D1%82%D0%B5-19f351301abc

Климат в команде

Описание

Эмоциональное состояние команды. Коллективная субъективная оценка участников команды повседневной работы, а также их собственных достижений.

Почему ветка важна?

Эмоциональное состояние команды влияет на эмоциональное состояние отдельных её членов, перфоманс команды. Для сбора субъективных оценок используют интервью, опросники или командную ретроспективу. Оценка и работа с этим аспектом команды влияет на:

- Перфоманс команды
- Удовлетворённость работой

Что будет, если её не делать?

Даже у зрелой команды есть эмоциональные спады, вызванные плохими результатами, конфликтами, уходом людей из команды. В случае, если не обращать на это внимание:

- На негативном эмоциональном фоне все негативные явления только усиливаются. Не закрываются спринты – команда перестаёт верить, что сможет их закрывать. Был конфликт – он только усугубится. Кажется, что все разбегаются – мне тоже пора бежать
- Команда неспособна самостоятельно выбраться из эмоциональной ямы, негативные эффекты будут только накапливаться

На кого может быть делегирована?

Scrum-мастер, тимлид ниже уровнем.

Способы прокачки

Консультации

- Чат TeamLead Bootcamp

Дизайн команды

Описание

Эта область ответственности актуальна в разных случаях – и для единой команды, работающей над одним бэклогом, и для команд, в которых разработчики участвуют одновременно в нескольких проектах. Задачей тимлида является учесть все ограничения и сформировать одну или несколько команд таким образом, чтобы работа шла без каких-либо проблем. Что нужно учесть:

- Требования проектов – количество людей, их уровень, навыки.
- Совместимость людей – насколько конкретные сотрудники могут ужиться друг с другом и продуктивно работать.
- Желания людей – над каким проектом кому интересно работать, задачи какого рода вдохновляют.
- Планы роста людей – работа над какими задачами поможет развитию требуемых навыков.

В случае работы над несколькими проектами у тимлида зачастую есть возможность на своё усмотрение перебрасывать людей. В случае единой команды такой переход обычно значит и перевод к другому руководителю.

Почему ветка важна?

Для менеджера:

- Замена людей – быстрый способ решения любых проблем с результатами, перфомансом или навыками команды. Но это не даётся бесплатно – любая перемена состава может сильно повлиять на состояние команды, поэтому любые изменения важно тщательно продумывать.
- Перекидывание людей между проектами и задачами позволяет точечно прокачивать требуемые компетенции и влиять на мотивацию.

Для сотрудника:

- Разнообразие рабочих задач и проектов – это отличный способ попробовать что-то новое и вырасти как профессионал.

Для компании:

- Переход в другую команду может быть альтернативой увольнению для сотрудника. Даже если одна команда потеряет ценного специалиста, в масштабах компании эта потеря будет не так

значима.

Что будет, если её не делать?

- Слишком частые и бессистемные перестановки не позволят команде глубоко узнать продукт и сработаться – уровень производительности будет стабильно низким.
- Если не держать под контролем требования по людям разных проектов, то где-то ресурсов будет не хватать, а где-то их будет избыток. В обоих случаях страдает и бизнес, и сотрудники.

На кого может быть делегирована?

Обычно не делегируется.

Примеры хорошего поведения

- Провести дизайн команды, опираясь на бэклог, над которым команда будет работать
- Учесть динамику личных отношений внутри коллектива
- Учесть личные интересы и планы развития потенциальных членов команды

Примеры плохого поведения

- Нанять "среднюю по рынку" команду: аналитик, дизайнер, разработчик, тестер
- Использовать такие правила, как "1 разработчик - 2 тестера"

Способы прокачки

Практика

Подход к дизайну команд, который используется в Large Scale Scrum (LeSS)

Ниже описанный подход призван лишь показать, как "качественно" оценить состав команды. Полученные данные нужно правильно интерпретировать.

1. Составление одного бэклога для проекта(ов)/продукта(ов), для которого необходима команда

В данном контексте *бэклог* - это список всей известной работы на текущий момент в рамках заданного направления

Пример развития онлайн-магазина:

№	Элемент бэклога
1	Лендинг-страница
2	Вывод в топ поисковиков
3	Форма обратной связи
4	Интеграция с ПО управления складом
5	Форма онлайн-оплаты

Размер элементов и их порядок не имеют принципиального значения.

2. Составление HitMap (матрица компетенций и элементов бэклога)

Дополните бэклог вторым измерением в виде компетенций, которые необходимы для выполнения данных элементов. На пересечении элементов и компетенций поставьте

- "0" - для реализации данного элемента данная компетенция не нужна
- "1" - для реализации данного элемента данная компетенция требуется

№	Элемент бэклога/ Компетенции	PHP	JS	SEO	Маркетинг	Бух. учёт	Админ. Linux
1	Лендинг-страница	1	1	0	0	0	1
2	Вывод в топ поисковиков	0	1	1	1	0	0
3	Форма обратной связи	1	1	0	0	0	0
4	Интеграция с ПО управления складом	1	0	0	1	1	1
5	Форма онлайн-оплаты	1	1	1	1	1	1

Можно не ограничиваться бинарной оценкой, экспериментировать с весами и коэффициентами. Если посчитать сумму по столбцам и нормировать их по общей сумме, то получим %-распределение компетенций, которыми должна обладать новая команда.

№	Элемент бэклога/ Компетенции	PHP	JS	SEO	Маркетинг	Бух. учёт	Админ. Linux
1	Лендинг-страница	1	1	0	0	0	1
2	Вывод в топ поисковиков	0	1	1	1	0	0
3	Форма обратной связи	1	1	0	0	0	0
4	Интеграция с ПО управления складом	1	0	0	1	1	1

№	Элемент бэклога/ Компетенции	PHP	JS	SEO	Маркетинг	Бух. учёт	Админ. Linux
5	Форма онлайн-оплаты	1	1	1	1	1	1
	Сумма	4	4	2	3	2	3
	% распределения	22%	22%	11%	17%	11%	17%

Т.е в нашем примере если мы бы хотели набрать команду из 10 человек, то состав примерно был бы следующим:

- 2 x PHP-разработчика
- 2 x JS-разработчика
- 1 x SEO-специалист
- 2 x eMarketing-специалист
- 1 x Специалист по бух.учёту
- 2 x Администратора Linux

Если учесть, что реальные люди владеют несколькими компетенциями, общая численность команды может быть снижена за счёт совместного перекрытия навыков.

3. Составление "Звёздной карты"

Если мы формируем команду в т.ч. из сотрудников компании, а не полностью набираем с рынка, то можем оценить, как они смогут потенциально перекрыть необходимые компетенции. Для этого возьмём список компетенций из п.2, но элементы бэклога заменим нашими сотрудниками. На пересечении в такой матрице нужно отметить уровень знаний сотрудника в данном направлении:

- "⭐ " - хорошо знаю, могу подсказать
- "👓 " - хочу изучить
- " " - не интересно

№	Сотрудник/ Компетенции	PHP	JS	SEO	Маркетинг	Бух. учёт	Админ. Linux
1	Петя	⭐		⭐		👓	👓
2	Вася	👓		⭐			⭐
3	Коля	👓		⭐	⭐		
4	Маша		⭐			👓	👓

Как и с прошлым упражнением, можно придумать любую градацию кроме предложенной. Сотрудникам она поможет найти экспертов в необходимой области, а тимлид "оцифрует её", чтобы

провести анализ. Это можно сделать поменяв пиктограммы на числа и сразу применив нормирование как в п.2:

- "⭐ " - 2
- "👓 " - 1
- " " - 0

№	Сотрудник/ Компетенции	PHP	JS	SEO	Маркетинг	Бух. учёт	Админ. Linux
1	Петя	2	0	2	0	1	1
2	Вася	1	0	2	0	0	2
3	Коля	1	0	2	2	0	0
4	Маша	0	2	0	0	1	1
	Сумма	4	2	6	2	2	4
	% распределения	20%	10%	30%	10%	10%	20%

4. Анализ распределений

Если наложить распределения из п.2 и п.3

Сотрудник/ Компетенции	PHP	JS	SEO	Маркетинг	Бух. учёт	Админ. Linux
Бэклог требует из п.2	22%	22%	11%	17%	11%	17%
Люди умеют из п.3	20%	10%	30%	10%	10%	20%

Как видно из сравнения в примере, что компетенции "Маркетинг" и "Бух. учёт" не могут быть закрыты полностью текущей командой, т.е. это возможные направления в усилении команды за счёт найма или обучения.

5. Воркшоп по объединению в команду

Как только вся потенциальная команда собрана, можно дать людям рассказать о себе, просто пообщаться или совместно выполнить какое-либо задание. Далее путём закрытого голосования определить готовность всех участников стартовать в таком составе.

Консультации

- Чат TeamLead Bootcamp

Теория

Ссылки

- HitMap и воркшоп по формированию команд

Запуск команды

Описание

Запуск команды является важной частью производственного цикла, в рамках него команда договаривается о большинстве аспектов своей работы.

Почему ветка важна?

Успешный запуск команды закладывает примерно до 20% успеха её будущих результатов, снижает риск конфликтов и снижение эффективности.

Что будет, если этого не делать?

Если не договориться на берегу, как мы и что мы делаем, то это может грозить в будущем снижением производительности, несбывшимися ожиданиями, конфликтами и общей апатией к происходящему.

На кого может быть делегирована?

Тимлид уровнем ниже

Способы прокачки

Можно использовать опросник ниже, чтобы во время старта новой команды не забыть каких-либо важных аспектов. Также такое упражнение можно провести с уже работающей командой или её новыми членами в рамках их онбординга.

О чем нужно договориться на старте команды

Продукт / Система / Домен

- Какой продукт / систему(ы) / направление деятельности компании мы развиваем?
- Какие глобальные проблемы акционеров / заказчиков / пользователей решаем?
- На чем мы зарабатываем?

- Кто наши пользователи?
- Кто является спонсором разработки?
- Какие ещё есть заинтересованные лица? (администраторы, инф. безопасность, аудиторы)
- Какие мы меряем наш успех?

Техническое направление

- Какой технологический стек используем?
- Кто принимает решения о его изменении?
- Какие есть ограничения? (что можем использовать, что нет)
- Какие правила написания кода?
- Какие правила ведения документации?

Ролевая модель

- Все ли в команде знакомы и знают друг друга по именам?
- Какие есть/планируются роли в команде? (тимлид, руководитель проекта, архитектор)
- Какие у этих ролей обязанности и зоны ответственности?
- Кто эти роли исполняет?

Процессы

- Какие у нас есть источники работ? Т.е. кто нам может ставить новые задачи?
- Кто принимает готовую работу?
- Какой методологии/подхода придерживаемся? (Scrum, RUP, Code&Fix, что-то своё)
- Какие есть периодические встречи команды? (Статус, Демо, Планирование)
- Какие из них обязательны и для каких ролей?
- Какие из них по желанию и для каких ролей?
- Как и кто оценивает задачи? (часы, сторипойнты, попугаи)
- Какой способы визуализации работы используем? (стикеры на стены, excel, переписка, jira, trello)
- Как отчитываемся о сделанной работе? (письмо руководителю в конце недели, таймшиты)

Рабочее пространство команды

- Где будет размешаться команда?
- Все ли необходимое оборудование и мебель имеются?
- Кто какое место займёт в комнате?
- Есть ли выделенные переговорные для совещаний?
- Если команда географически распределённая, настроены ли средства коммуникации? (телефоны, вкс, комнаты для конференций)

Более подробно рабочее пространство команды описано в отдельной ветке.

Каналы коммуникации

- Какие каналы коммуникации используем? (личные встречи, чаты, почта, документация)
- Как из них является первоочерёдным?
- Какие каналы для каких видов решений принимаем?

Командные соглашения

- Какое время прихода, ухода, обеда, перерывов?
- Как поведение приемлемо?
- Как поведение неприемлемо?
- Как принимаем решения? (большинство голосов, римское голосование, подбрасываем монету)

Консультации

- Чат TeamLead Bootcamp

Зрелость команды

Описание

Субъективная оценка отношения команды к повседневным задачам, способам их выполнения, особенностям командной динамики. Есть несколько моделей: Херси – Бланшара, пяти пороков Ленсиони. Эта оценка опирается:

- Как команда хочет получать задачи: инструкция vs конечная цель
- Какие есть отношения и ожидания друг от друга в команде

Почему ветка важна?

Команда большее, чем совокупность её членов, а значит необходимо работать и оценивать не только сотрудников в отдельности, но и команду в целом. Целенаправленная работа над зрелостью команды позволит:

- Увеличить продуктивность команды
- Получать результат лучше, чем было запланировано
- Работа в команде, которая любит своё дело – дополнительный фактор мотивации сотрудников

Что будет, если её не делать?

- Развитие команды остановилось. Например, делать ровно то, что написано в таске, не задумываясь о том какая проблема решается, считается нормой
  - Теряются возможности получить лучший результат, может получаться бессмысленный выхлоп не решающий проблему
- В команде отсутствует требовательность друг к другу
  - Делать спустя рукава становится нормой, потому что так делают все

На кого может быть делегирована?

HR, Product Owner, тимлид ниже уровнем

Способы прокачки

Консультации

- Чат TeamLead Bootcamp

Обеспечение прозрачности

Описание

Тимлид является интерфейсом между командой и окружающим миром – руководством, соседними отделами, всей компанией. Это не значит, что члены команды не должны самостоятельно общаться с кем-то из перечисленных, это значит, что на плечах тимлида лежит обязанность любым способом обеспечить нужную прозрачность. Все направление по обеспечению прозрачности можно разбить на три большие части:

- Отчётность перед руководством
- Обеспечение видимости достижений членов команды
- Информирование команды об изменениях и новостях

Предоставление отчётности может принимать очень разный вид: формальные письменные репорты, регулярные демо команды, быстрый статус-чек на one-to-one встречах, дэшборды в Jira. Самый простой способ определиться с форматом – обсудить его со своим руководителем, выделить волнующие его вопросы и определить удобную для всех периодичность.

Форматов обеспечения видимости достижений команды тоже может быть много – открытые демо, регулярные письма со списком достижений на всю компанию, статьи в Интранете. Главное – охватить максимальное количество релевантной аудитории и предоставить информацию в понятном для всех виде.

Информирование команды об изменениях и новостях проще всего проводить на регулярных встречах со всей командой – тимлид рассказывает о новостях и изменениях и даёт время задать волнующие вопросы.

Почему ветка важна?

Предоставление отчётности для руководства:

- Управление ожиданиями руководителя
- Дисциплинируешь себя регулярно смотреть на результаты команды со стороны заинтересованного лица

Обеспечение видимости достижений:

- Повышение ценности каждого члена команды внутри компании

- Получение сотрудниками признания их достижений
- Хорошие результаты – хороший тимлид

Информирование команды:

- Облегчает циркуляцию информации в компании
- Проще проводить организационные изменения
- Повышает лояльность сотрудников к компании

Предотвращение и разбор критических ситуаций:

- Снижает вероятность повторения технических или управленческих ошибок, пройденных другими командами. Кто-то извне может сообщать важную информацию
- Разбирать критические ситуации легче при высокой степени прозрачности и доверия. Понимать их причины и пути предотвращения

Что будет если её не делать:

- Неадекватная оценка руководством работы менеджера. Если нет хороших результатов, то сложно доказать, что причина проблем не в тебе.
- Команде сложнее помогать своему тимлиду, если она не понимает, чем он занят.
- Команда живёт в отрыве от реальности, что может привести к неадекватной оценке собственных результатов, статуса и места в системе.
- Любые изменения для команды будут происходить внезапно и не логично.

На кого может быть делегирована?

Все составляющие обеспечения прозрачности могут быть легко делегированы на ответственного участника команды. Хорошим вариантом может быть подключить к этому своего заместителя.

Примеры поведения

Примеры плохого поведения

- Не сообщать коллегам, подчинённым или руководству о новостях, планах, событиях, достижениях. Коллеги узнают об этом от третьих лиц.
- Придерживаться мнения, что видимость результатов работы подчинённого (visibility) есть исключительно его собственная проблема.
- Держать команду в информационном вакууме или доносить ей недостоверную информацию.

Примеры хорошего поведения

- Организовывать демонстрации (Show and Tell) внутри команды.
- Делать регулярные рассылки заинтересованным лицам с информацией о проделанной работе и планами.
- Демонстрировать прозрачность в своей работе, объяснять её плюсы. Не все люди понимают что такое прозрачность, не все осознают её преимущества, не все умеют быть прозрачными.
- Проводить регулярные встречи с командой, на которых рассказывать о последних новостях.

Способы прокачки

Консультации

- Чат TeamLead Bootcamp

Практика

Прозрачность наружу

1. Определите всех потенциальных стейкхолдеров вашей команды. В этом может помочь техника stakeholder mapping. Убедитесь, что в этом списке есть и ваш непосредственный руководитель.
2. Проведите опрос внутри технического отдела или всей компании о том, кому интересно следить за вашими новостями. Сразу приготовьтесь к тому, что в реальности заинтересована в этом будет лишь малая доля ответивших "да".
3. Пройдитесь по всем стейкхолдерам и проговорите, с какой частотой и в каком формате они хотят получать информацию о работе вашей команды и её результатах.
4. С учётом требований всех стейкхолдеров определитесь с форматом информирования их и других заинтересованных групп внутри компании. Это, например, могут быть демо, письменные отчёты, краткие дайджесты в Slack. Постарайтесь сделать так, чтобы выбранные вами форматы не задевали команду и не съедали её время.
5. Для каждого формата определитесь с тем, как будете контролировать его эффективность – это могут быть количество просмотров ваших отчётов, качественная оценка удовлетворённости заказчиков.

Прозрачность внутрь

1. Назначьте с командой регулярные встречи или встройтесь в текущую рутину.
2. Подготовьте небольшой рассказ о том, что происходило в компании последнее время – это могут быть новости из соседних отделов, стратегические планы, интересные слухи.
3. Соберите с команды запросы того, о чем ещё им хотелось бы узнать, и договоритесь насчёт периодичности в будущем.

Теория

Статьи

- О Visibility

Организация рабочего пространства

Описание

К рабочему пространству относится все, что позволяет команде эффективно выполнять свою работу. Задачей тимлида является такое пространство обеспечить и решить все возможные неудобства.

Какие вопросы должен закрывать тимлид:

- Актуальность железа, которым пользуется его команда.
- Наличие у всех нужного лицензионного софта и выстроенный процесс заказа новых лицензий.
- Прозрачный механизм получения прав ко всем нужным рабочим сервисам (трекер, база знаний, SCM).
- Для распределённой команды – построенный процесс и регламент взаимодействия, площадка для созвонов.
- Для сидящей вместе команды – организовать единое пространство с местом для стендапов, доской и другим оборудованием.

Работа тимлида заключается как в оперативном решении возникающих проблем и вопросов, так и в выстраивании процессов, которые позволяют не уделять этой области слишком много времени. Как пример, чтобы упростить закупку новым сотрудникам софта, тимлиду достаточно составить общий список доступных команде лицензий и договориться с отделом закупок о работе через трекер задач.

Комфортное рабочее пространство относится к гигиеническим факторам мотивации – если с ним все хорошо, команда этого не замечает, но если плохо – то это может сильно повлиять на их вовлеченность и производительность.

Почему ветка важна?

Для менеджера:

- Один раз потратив время на выстраивание процессов решения этих проблем, можно будет избавиться от большого объёма операционки.
- Некомфортное рабочее пространство сильно демотивирует команду и может создавать “эффект разбитых окон”.

Что будет, если её не делать?

- Разные отвлекающие и раздражающие факторы вроде шума или тормозящего железа мешают команде сосредотачиваться на своей задаче.
- Комфортное рабочее пространство относится к гигиеническим факторам мотивации. Проблемы в этой сфере могут легко свести на нет усилия во всех остальных.

На кого может быть делегирована?

На тимлида ниже уровнем или любого ответственного сотрудника в команде.

Способы прокачки

Консультации

- Чат TeamLead Bootcamp

Теория

Подкасты

- Podlodka – Работа в распределённой команде

Развитие технического бренда

Описание

Технический бренд компании – это совокупность эмоций, знаний, мнений и ассоциаций о технической стороне компании в головах потенциальных клиентов или соискателей. Чаще всего складывается из следующих составляющих:

- Публичный образ компании
- Отзывы текущих и бывших сотрудников
- Внешний вид и качество работы продукта компании
- Качество и содержание технологического контента, создаваемого компанией

На первые три пункта тимлид зачастую влияет лишь опосредованно, в то время как четвёртый находится целиком в его зоне ответственности. Можно выделить следующие этапы работы:

1. **Определение того, чем команда может и должна делиться.** Тимлид помогает команде понять, что из того, чем они занимаются, может быть интересно сообществу. Это могут быть как технологии, так и конкретные прикладные истории.
2. **Подготовка контента.** Тимлиду отводится роль того, кто отвечает за качество итогового содержания и подачи. Для этого он помогает своим сотрудникам подготовить план и тезисы, сам или с помощью специально обученных людей оказывает поддержку в дизайне или редактуре, проводит финальный контроль качества.
3. **Распространение контента.** Если контент хорош, но его никто не увидел, пользы для техбренда он принесёт мало. В качестве инструментов распространения и продвижения можно использовать социальные сети, профессиональные сообщества (чаты, митапы, тематические сайты).
4. **Анализ обратной связи.** Обратная связь по содержимому и форме подачи важна, так как помогает корректировать направление развития техбренда на будущее.

Инструменты развития технического бренда, которые может использовать тимлид:

- Статьи
- Выступления на внешних площадках
- Видеозаписи внутренних выступлений
- Open Source технологий или прикладного кода
- Playbook с информацией про команду

Почему ветка важна?

Для компании:

- Растёт поток людей, которые приходят на собеседования сами, без работы рекрутера.
- Упрощается онбординг и проверка на culture-fit к компании – собеседуемый уже сильно в контексте происходящего.
- Гордость за свою компанию и её технологический уровень – важный фактор в удержании.

Для тимлида:

- Участие в активностях по развитию технобренда помогает прокачивать различные soft skills сотрудников: навыки публичных выступлений, написания качественных текстов.

Для сотрудника:

- Развитие личного бренда и повышение своей стоимости на рынке.
- Твои скиллы оценит не только руководитель и команда, а широкое сообщество профессионалов.
- Нетворкинг с людьми, решающими схожие задачи.

Что будет, если её не делать?

- Останется полностью не закрытым один из мотивационных факторов. Сотрудника будет легко переманить в компанию, которая активно развивает технический бренд.
- Теряется канал обратной связи от сообщества, которое может вовремя сигнализировать о том, что используемые в компании технологии или подходы устарели для рынка.

На кого может быть делегирована?

- На активистов внутри команды, которым эта тема интересна.
- На выделенного DevRel-менеджера.

Примеры поведения

Примеры плохого поведения

- На подготовку статей, выступлений или open source не выделяется рабочее время.

- Тимлид выключается из процесса подготовки публичного выступления, не даёт свою обратную связь.

Примеры хорошего поведения

- Участники команды регулярно выступают на профильных мероприятиях.
- Внутри команды или компании налажена практика предварительных прогонов выступлений и дачи обратной связи.
- Участие в активностях по развитию технического бренда влияет на материальное поощрение сотрудника.
- У команды есть собственный технический блог на одной из площадок.
- Тимлид помогает команде понять, чем действительно интересно делиться наружу – у самих разработчиков глаз часто бывает замылен.

Способы прокачки

Консультации

- Чат TeamLead Bootcamp

Практика

1. Определите ваши цели – чего конкретно вы ждёте от развития технического бренда. Это может быть любой из пунктов, перечисленный в разделе "Почему ветка важна?".
2. В зависимости от цели определитесь с чувствительными и релевантными метриками ии способами их измерения. Например, если вы хотите повышать количество людей, которые приходят к вам на собеседования, попросите всех рекрутеров узнавать у кандидатов, слышали ли они ваши доклады.
3. Проведите анализ того, где вы находитесь сейчас относительно ваших целей, и того, что вы уже делаете для техпиара. Хороший вопрос, который стоит себе задать – "Насколько текущие активности помогают достигать поставленных целей?"
4. Обсудите с каждым членом команды, как ему интересно работать над техническим брендом. Кто-то хочет выступать, кому-то интереснее писать статьи или работать над open source проектом, кого-то все это вообще может не интересовать. Одновременно с обсуждением каналов, подумайте и над контентом. Наводящий вопрос: "А что интересного ты делал за последнее время, что мы могли бы пошарить?".
5. Используйте готовый механизм личного целеполагания, если он есть, чтобы поставить цели по техпиару сотрудникам.
6. Составьте общий план ваших активностей по развитию технического бренда. По каждой из них собирайте обратную связь и анализируйте, насколько она приближает вас к вашим целям.

7. Выделите низко висящие фрукты в получившемся плане – те активности, которые при наименьшем количестве затрачиваемых усилий дадут больше всего результата.

Теория

Книги

- Код публичности, Ана Мавричева
- Никогда не ешьте в одиночку и другие правила нетворкинга, Кит Феррацци

Статьи

- Про что выступать

Подкасты

- Про DevRel и технический бренд
- Про организацию конференций
- Про личный бренд разработчика

- Инструменты DevRel в арсенале тимлида, Михаил Клюев

Генерация элементов бэклога

Описание

Бэклог – это упорядоченный список задач, которые поступают на вход команде разработки. Менеджер отвечает за его наполнение обоснованными и ценными Product Backlog Items (PBI). Для каждого из запланированных улучшений он должен отвечать на два вопроса:

- Почему это улучшение ценно?
- Удобно ли это улучшение для пользователя?

Для генерации новых элементов бэклога менеджер может использовать следующие источники:

- Аналитика продукта
- Интервью с пользователями
- Результаты UX-исследований
- Анализ конкурентов
- Анализ рынка и трендов
- Стратегия бизнеса и продукта

В рамках этой ветки рассматривается только работа с улучшениями продукта. Задачи, связанные с техническим долгом или инженерной культурой, являются ответственностью другой роли – Technical Lead.

Почему ветка важна?

Для компании:

- В разработку уходят только идеи с подтверждённой гипотезой ценности.
- Любой желающий может ознакомиться с планами команды, изучив её бэклог.
- Вопросам удобства использования уделяется внимание до старта разработки задачи.

Для менеджера:

- Есть единый источник правды, определяющий загрузку и приоритеты команды разработки.
- Для каждого из улучшений, которое команда берет в разработку, можно чётко ответить, почему его надо делать.

Для команды:

- Возможность планировать следующие итерации разработки без стресса, если на входе у них есть готовый бэклог.

Что будет, если её не делать?

- В разработку будут поступать задачи, не прошедшие проверку ценности или удобства.
  - Фичи не принесут ожидаемой пользы для продукта и бизнеса.
- Команда будет хаотично прыгать между задачами из разных областей продукта.
  - Не будет системного продвижения к поставленным продуктовым целям.
- Любой стейкхолдер сможет свободно проталкивать свои задачи в разработку, исходя из принципа "кто громче кричит".

На кого может быть делегирована?

- Product Owner
- Тимлид ниже уровнем
- Разработчик, ответственный за конкретную фичу

Примеры поведения

Примеры плохого поведения

- Бэклога нет.
- В бэклоге появляются задачи, ценность которых не подтверждена ничем, кроме экспертного мнения тимлида или заказчика.
- Удобство новых фич проверяется только на конечных пользователях после релиза.
- PBI описаны хаотично или вообще не имеют описания.
- Задачи в бэклоге не связаны совсем, либо связаны лишь косвенно с целями команды.

Примеры хорошего поведения

- У команды есть продуктовый бэклог, который доступен на просмотр любому желающему.
- Каждый PBI описан по стандартному шаблону. Можно изучить изначальную гипотезу, её валидацию, посмотреть definition of done.
- Все члены команды знают, что за фича разрабатывается, почему она ценна, какая польза от неё ожидается на выходе.
- Бэклог прозрачен для стейкхолдеров.

Способы прокачки

Практика

Пройди отличный онлайн-курс управления продуктом на основе данных Go Practice. Он реализован в виде симулятора – всю получаемую теорию ты сразу же применяешь в специальной песочнице на практике.

Консультации

- Telegram-чат TL Bootcamp
- Продуктовый Telegram-чат No Flame no Game

Теория

Статьи

- What is the Product Backlog in Product Management and How to Maintain It
- How to Fill Up Your Product Backlog
- Как формировать бэклог и приоритезировать его

Книги

- Intercom on Product Management
- Inspired: How to Create Products Customers Love, Marty Cagan

Приоритизация бэклога

Описание

Бэклог – это упорядоченный список задач, которые поступают на вход команде разработки. В рамках этой ветки менеджер занимается тем, что определяет механизм приоритизации элементов бэклога (PBI) и применяет его на практике. Проще говоря, задачей менеджера является договориться о принципах, по которым определяется, какую задачу брать в разработку следующей.

Почему ветка важна?

Для компании:

- Есть уверенность в том, что команда всегда разрабатывает именно то, что важно в настоящий момент.
- При смене целей или курса движения можно не перетряхивать весь бэклог, а просто переработать алгоритм приоритизации.

Для менеджера:

- Можно чётко объяснить принципы, по которым та или иная задача поступает в работу.
- При наличии нескольких потоков поступления PBI, алгоритм приоритизации позволяет системно ими управлять.

Для команды:

- Уменьшает количество споров на этапах груминга и планирования.

Что будет, если её не делать?

- Команда будет делать не то, что нужно, а то, что просто/интересно/необычно.
- Сессии планирования и грумингов будут затягиваться из-за попыток выяснить на месте приоритет каждой задачи.
- Фокус команды разработки будет постоянно меняться из-за появления новых задач на разработку из других областей.
- Любой стейкхолдер сможет свободно проталкивать свои задачи в разработку, исходя из принципа "кто громче кричит".

На кого может быть делегирована?

- Product Owner
- Project Manager
- Тимлид ниже уровнем

Примеры поведения

Примеры плохого поведения

- Отсутствует чётко определённый механизм приоритизации бэклога.
- Алгоритмы приоритизации постоянно меняются.
- Используемый механизм приоритизации оторван от реалий компании, например, опирается на продуктовые метрики, которые не признаны ключевыми.

Примеры хорошего поведения

- Команда разработки вовлечена в оценку сроков задач и рисков их реализации.
- Используемый механизм приоритизации описан в командной Wiki.

Способы прокачки

Практика

1. Оцените условия, в которых сейчас находится команда. Должны ли решения о приоритетах приниматься только внутри команды (например, она целиком отвечает за повышение каких-то метрик), или нужно вовлекать стейкхолдеров со стороны.
2. В зависимости от полученных результатов, выберите одну из моделей приоритизации.
3. Соберите список задач, которые были реализованы командой за последние несколько итераций, и ретроспективно приоритизируйте их, используя выбранную модель. Сделайте вывод о том, насколько она вам подходит. Если не очень – попробуйте другой метод.
4. Вместе с несколькими членами команды устройте сессию первичной приоритизации существующего бэклога с использованием выбранной модели.
5. Во время груминга бэклога, либо в рамках отдельных сессий, все новые задачи пропускайте через выбранную модель.

Консультации

- Telegram-чат TL Bootcamp
- Продуктовый Telegram-чат No Flame no Game

Теория

Статьи

- 20 Product Prioritization Techniques
- Как формировать бэклог и приоритезировать его
- Оценка PBI по взвешенным факторам

Книги

- Intercom on Product Management

Жизненный цикл фичей

Описание

Все фичи, из которых состоит продукт, в разное время находятся в разных фазах своего жизненного цикла.

Этап разработки:

- поиск проблемы
- валидация проблемы
- поиск решения
- валидация решения

Этап раскатки:

- внутренний релиз
- релиз на процент пользователей
- полная раскатка)

Этап дальнейшего улучшения:

- этап роста
- этап зрелости

И этап вывода из использования. Вне зависимости от того, какая конкретно модель жизненного цикла используется в вашем продукте, смысл этой ветки заключается в контроле того, что каждая новая фича не выбивается из этой модели и соответствует ей.

Почему ветка важна?

Для продукта:

- Продукт не захламляется бесполезными фичами, которые никто не использует
- Внезапное добавление новых фичей в продукт не ломает устоявшиеся пользовательские сценарии
- Фичи не забрасываются после выкатки, а продолжают анализироваться и улучшаться
- Понимание текущей стадии жизненного цикла всего продукта или отдельных фич используется для координации маркетинговых усилий

Для тимлида:

- Стадии жизненного цикла могут использоваться для приоритизации задач
- Любая новая фича – это всего лишь гипотеза о её ценности. Постепенное продвижение по этапам жизненного цикла позволяет снимать неопределённость вокруг этой гипотезы постепенно

Что будет, если её не делать?

- В продукт будут добавляться все подряд фичи, не проходя предварительной проверки на их ценность
- Продукт будет постепенно обрастать огромным количеством малоиспользуемой функциональности
- Продукт может ощущаться сырым и не качественным, состоящим из большого количества недоделанных до конца фичей
- Пользователи не будут адаптироваться к новым возможностям и использовать их

На кого может быть делегирована?

- Product Owner
- Тимлид ниже уровнем
- Разработчик, ответственный за конкретную фичу

Примеры поведения

Примеры плохого поведения

- Нет цикла проверки продуктовых гипотез, либо какие-то фичи его обходят стороной
- Первые знания о том, как пользователь взаимодействует с фичей, приходят к команде только после её выкатки на 100% аудитории
- Не проводится периодический анализ того, сколько людей и как часто взаимодействуют со всеми фичами продукта
- Малоиспользуемая или бесполезная функциональность не убирается из продукта
- После релиза не анализируется то, как аудитория использует новую фичу
- Не происходит постепенного улучшения показателей фичей в продукте

Примеры хорошего поведения

- Есть налаженный процесс product discovery, который описывает то, как гипотеза о проблеме превращается в разработанную фичу

- Раскатка новых фичей происходит поэтапно, причём каждый этап добавляет команде знаний о пользователях
- Есть понятный для команды и пользователей процесс удаления фичей из продукта

Способы прокачки

Консультации

- Telegram-чат TL Bootcamp
- Продуктовый Telegram-чат No Flame no Game

Теория

Статьи

Раскрывают тему:

- The 6 Stages of the Product Life Cycle
- Product Discovery – детальное описание процесса
- Product Discovery: A Practical Guide for Agile Teams
- The End: Guide to Deprecating a Digital Product

Дополнительные материалы:

- Product Development Flow in GitLab
- Product Discovery в Avito
- On Killing It by Killing Features

Книги

- Intercom on Product Management

Знание рынка

Описание

Для того, чтобы развивать продукт, создавая новые фичи и улучшая существующие, менеджер должен хорошо понимать рынок, на котором работает. Все знание рынка можно поделить на четыре основные сегмента:

- **Конкуренты.** Какие продукты конкурируют с нами за пользователей как напрямую, предоставляя ту же функциональность, так и косвенно, закрывая потребность пользователя другими способами.
- **Государство.** Как на продукт влияет законодательство стран, выбранных в качестве рынка.
- **Особенности рынка.** Любые особенности выбранных рынков – география, культура, история.
- **Тренды.** Куда движется индустрия, какие глобальные веяния и тенденции могут изменить правила игры.

Почему ветка важна?

- Вместо того, чтобы совершать ошибки самому, можно учиться на ошибках других.
- Знание любого из перечисленных сегментов может служить источником инсайтов для наполнения продуктового бэклога.
- Аналитика рынка позволяет уменьшить неопределённость будущего.

Что будет, если её не делать?

- Все внешние воздействия на продукт будут сюрпризом.
- Повышаются риски совершения ошибок в процессе развития продукта.

На кого может быть делегирована?

- Product Owner
- Аналитик

Примеры поведения

Примеры плохого поведения

- Не знаешь, кто является конкурентом твоего продукта.
- Не следишь за трендами индустрии.
- Не изучаешь опыт схожих продуктов и рынков.

Примеры хорошего поведения

- На регулярной основе проводишь конкурентный анализ и отслеживаешь основные показатели конкурентов.
- Участвуешь в профильных конференциях, следишь за лидерами мнений.
- Включаешь особенности госрегулирования твоего сектора в оценку рисков.
- Проводишь custdev пользователей конкурентов.

Способы прокачки

Практика

Конкуренты

Выдели основных конкурентов своего продукта и проведи конкурентный анализ.

Государство

Если у тебя в компании есть юридический отдел, организуй с ними встречу. Узнай, какие конкретно аспекты законодательства могут повлиять на твой продукт. Зоны для вдохновения: обработка и хранение персональных данных, госрегулирование, возрастной контроль.

Если юридического отдела нет, найди подходящее консалтинговое агентство и привлеки их по тем же самым вопросам.

Особенности рынка

Подпишись на профильные медиа, связанные с твоим рынком и индустрией – Telegram-каналы, блоги, журналы. Бывает, что такие издания не находятся в открытом доступе. В таком случае стоит спросить совета коллег, работающих у конкурентов, либо у подходящего консалтингового агентства.

Тренды

Подпишись на "Стартап дня" и vc.ru. Смотри видео с конференций WebSummit, MWC, WWDC, Google I/O.

Консультации

- Telegram-чат TL Bootcamp
- Продуктовый Telegram-чат No Flame no Game

Теория

Статьи

- Конкурентный анализ

Книги

- "Революция платформ", Джеффри Паркер
- "Неизбежно: 12 технологических трендов, которые определяют наше будущее", Кевин Келли
- "Машина, платформа, толпа. Наше цифровое будущее", Эндрю Макафи
- "Сверхдержавы искусственного интеллекта", Кай Фу Ли

Знание пользователей

Описание

Product Owner является защитником интересов тех, кто использует его продукт. Если упростить, то эта ветка заключается в следующем:

- Определение целевой аудитории продукта и её сегментирование
- Сбор, анализ и решение болей пользователей

Почему ветка важна?

- Ресурсы команды ограничены, поэтому нужно заниматься приоритизацией фич. Понимание целевой аудитории и её сегментов даёт базис для приоритизации.
- Боли пользователей – источник инсайтов для генерации новых фич.

Что будет, если её не делать?

- Без фокуса на чётких пользовательских сегментах продукт будет просто сборником мало связанных друг с другом фичей.
- Теряется важный мотивационный фактор в управлении командой.

На кого может быть делегирована?

- Product Owner
- Аналитик

Примеры поведения

Примеры плохого поведения

- Ни ты, ни команда не можете описать пользователей своего продукта.
- Не понимаешь, зачем пользователи используют твой продукт.
- Не можешь назвать основные боли пользователей в текущий момент.
- Не общаешься с живыми пользователями.

Примеры хорошего поведения

- Команда регулярно общается с пользователями своего продукта.
- Команда в курсе пользовательских исследований и инсайтов.
- При разработке фичи всегда учитывается, для какого сегмента пользователей она разрабатывается.
- Регулярно проводятся маркетинговые исследования для обновления информации о целевой аудитории.

Способы прокачки

Практика

Определение целевой аудитории

1. Максимально широко определите, кто является вашей целевой аудиторией. Вариант "все люди", конечно, возможен, но лучше все-таки быть более конкретным.
2. Вместе с командой определите значимые критерии сегментации – те, разделение по которым влияет на боли, запросы и потребности пользователей. Например, для текстового редактора нет смысла сегментировать пользователей на основе их пола, но есть смысл на основе географии или целей использования.
3. Опишите ваши гипотезы о сегментах, их целях и объёмах. Максимально дешёвым способом провалидируйте их на существующих данных, с помощью пользовательских интервью или маркетинговых исследований.
4. Приоритизируйте ваши сегменты с использованием взвешенной оценки. Можно учитывать такие параметры, как размер сегмента, лёгкость выхода на его представителей, потенциальная прибыль.
5. Опишите полученные сегменты и используйте их в дальнейшем для планирования стратегии и приоритизации новых фич.

Общение с пользователями

1. Проведите брейншторм с командой способы выйти на релевантных для вас пользователей. Это может быть размещение объявления на Авито, сообщение в релевантном сообществе, профильное оффлайн-мероприятие.
2. Соберите контакты пользователей с использованием выбранных каналов.
3. Договоритесь о постоянном проведении пользовательских интервью, на которых вы сможете валидировать ваши гипотезы о проблемах, собирать обратную связь по фичам.

4. Позовите пользователей на демо команды – это даст отличную возможность сократить цикл получения обратной связи.

Анализ болей

1. Постройте канал сбора болей пользователей. Стоит посмотреть на социальные сети, данные от службы поддержки, отзывы в магазинах приложений.
2. Регулярно разбирайте этот канал и релевантные боли переводите в отдельный бэклог. В качестве критерия приоритизации можно использовать количество обращений.
3. Сохраняйте контакты тех, кого касалась боль, чтобы сообщить им о релевантном запуске.

Консультации

- Telegram-чат TL Bootcamp
- Продуктовый Telegram-чат No Flame no Game

Теория

Статьи

- Что такое Jobs-To-Be-Done и Job stories

Книги

- "Crossing the Chasm", Geoffrey A. Moore
- Intercom on Jobs-to-be-done
- "Спроси маму", Rob Fitzpatrick

Автоматизация цикла разработки/CI

Описание

CI (Continuous Integration или Непрерывная интеграция) – это автоматическая сборка программного обеспечения и его тестирование на работоспособность.

Почему ветка важна?

- Настроенный CI помогает быстрее доносить изменения кода на сервера.
- Тестировщики могут сразу узнавать в какой момент развернулась новая версия и приступать к её тестированию.
- Запуск автотестов в окружении приближенном к боевому.
- Гарантия стабильности основной ветки разработки. Это даёт возможность спокойного создания новых фич и релиз веток.
- Непрерывная интеграция означает, что мы на каждый коммит нашей ветки понимаем готовность нашей ветки сливаться в trunk.

Что будет если не настроить CI?

- Увеличивается время на тестирование
- Увеличивается время на сборку и развёртывание ПО, так как это происходит вручную
- Отнимает время команды на ту самую сборку и развёртывание, когда команда бы могла заниматься в это время исправлением багов или написанием новых фич.
- Возможно придётся изменять статусы тасок вручную

На кого может быть делегирована?

- Тимлид ниже уровнем
- DevOps - специалист
- Разработчик

Примеры поведения

Примеры плохого поведения

- Дёмонстрация заказчику без тестирования, так как у команды не было времени на ручное развёртывание кода или не было ответственного за это
- Программист во время разработки проверяет только локально, после выкатки на прод., он ломается. А разработчик говорит: "У меня локально то все работает". Как итог недовольные клиенты и заказчик. Решение: иметь два окружения с настроенным CI, одно для разработчиков, другое для тестировщиков.
- Автотесты проходят на локальном окружении, а на серверах падают
- Нет возможности быстро откатывать прод. к старой, рабочей версии

Примеры хорошего поведения

- В компании настроен стандартный процесс по настройке CI (есть стандартные конфигурации)
- В компании настроены системы для автоматического развёртывания (например: Jenkins, Kubernetes)
- Настроены оповещения о начале сборки и её окончании, удобным для команды способом (например: оповещения в чат telegram)
- Используемый task manager интегрирован в CI, что помогает автоматически менять статусы у задач
- Версионность билдов
- Настроена возможность создавать ветки релиза из протестированного билда

Способы прокачки

Практика

- Чтобы отработать навыки по настройке CI, вы можете на своём компьютере развернуть VM или Vagrant настроить как сервер, и попробовать самостоятельно установить необходимую систему для развёртывания и настроить её.
- Попросить у команды или у руководства возможность, настроить CI для следующего стартующего проекта
- Пройти различные курсы, например:
  1. Learn DevOps: CI/CD with Jenkins using Pipelines and Docker
  2. DevOps: CI/CD with Jenkins pipelines, Maven, Gradle
  3. DevOps Project: CI/CD with Jenkins Ansible Docker Kubernetes
  4. Docker and Kubernetes: The Complete Guide

Консультации

- Telegram-чат TL Bootcamp.
- DevOps - инженеры, разработчики в вашей компании, кто уже этим занимался.

Теория

Книги

- Джез Хамбл, Дэйвид Фарли Непрерывное развёртывание ПО. Автоматизация процессов сборки, тестирования и внедрения новых версий программ
- Философия DevOps. Искусство управления IT - лучше в оригинале
- Continuous delivery. Практика непрерывных апдейтов

Видео

- Илья Климов — Ламповый CI/CD. Как и с чего начать
- Лучшие практики CI/CD с Kubernetes и GitLab (Дмитрий Столяров, Флант, HighLoad++ 2017)
- DevOps-Projects

Статьи

- Феншуйная автоматизация CI & CD с помощью Jenkins и Jira
- CI/CD: принципы, внедрение, инструменты
- What is CI/CD and how to setup your CI pipeline for mobile?

Работа с системами контроля версий

Описание

VCS - система контроля версий. Стандартом в индустрии является Git, иногда встречается Mercurial и SVN, и очень редко другие. Умение грамотно работать с VCS позволяет:

- Оперативно исправлять критические баги
- Отслеживать вклад разработчиков в продукт
- Контролировать качество кода

Почему ветка важна?

Современные VCS позволяют:

- работать одновременно над одним или разными частями файлов
- оперативно синхронизировать кодовую базу
- иметь несколько вариантов состояний кодовой базы в разных ветках
- разрешать конфликты изменений
- просматривать историю изменений конкретных файлов или фрагментов кода
- собирать статистику по разработчикам

Что будет, если её не делать?

- Низкое качество кодовой базы
- Невозможность быстро решать критические проблемы

На кого может быть делегирована?

Умение работать с VCS на продвинутом уровне может быть делегировано на техлида или старших разработчиков.

Практика

Навыки работы с Git

Для комфортной разработки и взаимодействия в команде каждый разработчик должен уметь:

- Клонировать репозиторий
- Создавать ветки
- Переключаться между ветками
- Делать коммиты
- Делать fetch, pull и push
- Делать слияния веток
- Уметь разрешать конфликты при merge
- Уметь создавать Pull (Merge) Request

Один из опытных разработчиков, техлид или тимлид должны дополнительно уметь:

- Делать rebase
- Делать force push
- Работать с тегами
- Настраивать precommit хуки
- Настраивать конфиг git
- Работать со stash и знать для чего это необходимо

Наименования коммитов

1. В команде должно быть соглашение о наименовании коммитов
2. Все члены команды должны этому соглашению следовать

Пример соглашения

1. Все коммиты пишутся на русском языке
2. Все коммиты должны иметь следующий формат: <номер задачи> <глагол> <субъект> <детали>` Пример: JIRA-123 Изменил цвет кнопки "Сохранить" на красный
3. Допустимо писать подробности коммита в произвольной форме отступив 1 строку от заголовка
4. Недопустимо делать подряд несколько коммитов с одинаковой подписью
5. Название коммита должно показывать бизнес изменения, если они были, а не изменения в кодовой базе
6. В коммитах необходимо использовать термины из словаря проекта.

Хорошо

```
JIRA-123 Изменил цвет кнопки "Сохранить" на красный

JIRA-123 Исправил ошибку в сообщении при ошибке авторизации

JIRA-123 Сверстал модальное окно подписки на новости сервиса
```

```
JIRA-123 Сделал обводку в информационной модалке
```

Плохо

```
fix
```

```
Отрефакторил код
```

```
Исправил мелкие баги
```

```
Добавил border: 1px в MessagePopup
```

Почему это важно?

- Облегчается поиск по изменениям
- Появляется дополнительная документация
- Легко найти ответ на вопросы:
  - Зачем это сделано?
  - Когда это сделано?
  - Ошибка это или требование?
  - В рамках какой задачи это сделано?
  - Кем это сделано?
  - Можно ли это отрефакторить?
  - Устарел ли это код?
- Можно быстро формировать отчёты (ReleaseNotes)
- Легко понимать кто и что сделал за определённый промежуток времени

Консультации

- Telegram-чат TL Bootcamp

Теория

Способы прокачки

1. Pro Git book
2. Скринкаст по Git
3. Игра в гит

Методологии

GitHub-flow

GitHub-flow

Git-flow

Удачная модель ветвления для Git

Пожалуйста, перестаньте рекомендовать Git Flow

Trunk-Based Development

Trunk-Based Development

Статьи по теме

1. Как сделать мир (или хотя бы ваш проект) чуточку лучше
2. Как следует писать комментарии к коммитам

Техническая документация

Описание

Техническая документация подразделяется на пользовательскую и внутреннюю. **Пользовательская документация** включает в себя инструкции по использованию и установке продукта, описания API, системные требования и прочие документы, предназначенные для конечного пользователя. **Внутреннюю документацию** в свою очередь можно разделить на продуктовую (описания архитектуры проекта, мануалы по разворачиванию сред разработки, воспроизведению результатов экспериментов, UX документация) и процессуальную (гайдлайны по написанию кода и процедуре code review, документацию по командным процессам, отчёты для менеджмента, планы и roadmaps).

Почему ветка важна?

Для тимлида:

- Помогает всегда оставаться в курсе актуального состояния проекта и делиться этой информацией с бизнесом
- Облегчает процесс онбординга новых членов команды
- Упрощает коммуникацию с пользователями и stakeholders

Для команды

- Значительно снижается сложность вникания в новый проект
- Всегда есть актуальный референс, любой член команды может понять, что происходит в других частях проекта
- Увеличивается bus factor, происходит распространение знания внутри команды
- Зачастую сам процесс написания документации улучшает понимание архитектуры проекта и приводит к обнаружению багов или появлению новых идей

Для пользователей

- Легче пользоваться продуктом, возникает меньше вопросов и ошибок

Что будет, если её не делать?

- Внутренняя документация

    - Новые сотрудники будут тратить значительно больше времени на погружение в проект. Также они будут отнимать драгоценное время у членов команды, задавая вопросы, которые легко могли быть покрыты документацией
    - Проект будет обрастать незадокументированными фичами, знание о которых будет иметься лишь у нескольких людей, и даже эти люди в какой-то момент о них забудут
- Пользовательская документация
    - Возрастает нагрузка на техническую поддержку
    - Увеличивается недовольство пользователей

На кого может быть делегирована?

На любого разработчика или технического писателя

Примеры поведения

Примеры плохого поведения

- Обновление документации не включается в Definition of Done или не выносится отдельными PBI
- Затраты времени на написание документации не учитываются при планировании
- Документация не версионируется
- Важные изменения документации не проходят процедуру ревью
- Документация написана слишком длинно и нудно, содержит устаревшую и ненужную информацию
- Документация пишется не итеративно, а огромными пластами
- Документация пишется "для галочки", её ценность для команды и пользователей не определена, и её никто не читает

Примеры хорошего поведения

- У команд есть единые шаблоны, правила оформления и стиля для создания документации
- У каждого документа имеется понятная целевая аудитория (пользователи, другие команды, члены команды, менеджмент)
- Документация хранится в едином удобном месте с функцией поиска, а не раскидана по Google Docs, Confluence, Wiki, Redmine и бумажкам в офисе
- Используется методология Docs-as-code - документация пишется в текстовом редакторе (например, в формате Markdown), хранится в репозитории, версионируется, тестируется и проходит процедуру peer review, считаются метрики качества (например, readability), новые версии автоматически публикуются
- Документация хорошо оформлена, важные места выделены и бросаются в глаза
- Документация содержит ссылки на другие релевантные документы

- Документация время от времени тестируется на актуальность
- Обновление документации автоматизируется (насколько это возможно)
- Появление новой важной документации сопровождается туториалами и воркшопами для целевой аудитории
- Документация содержит средства визуализации (диаграммы, графики)

Способы прокачки

Практика

1. Полезно читать документацию опенсорсных проектов и других команд и заимствовать лучшие идеи
2. Регулярно структурируйте имеющуюся документацию и оценивайте её качество для разных частей проекта
3. В запущенных случаях может быть полезен внешний аудит документации
4. Для каждого типа документации нужно определить оптимальный способ написания (по ходу разработки vs document late)
5. Найдите правильный баланс между отсутствием документации и избыточной документацией. Соберите обратную связь от разных групп (разработчики, пользователи, тестировщики, другие команды)
6. Не бойтесь пробовать нестандартные формы документации - например, видео-дискуссии или подкасты

Консультации

- Telegram-чат TL Bootcamp

Теория

Подкасты

- SE Radio - Agile Documentation

Статьи

- Technical Documentation in Software Development
- A Roadmap to Agile Documentation
- Model Cards for Model Reporting

Распространение знаний

Описание

Помимо преобразования технических требований в работающую функциональность команда разработки постоянно генерирует новые знания. Это могут быть:

- Знания реализации и принципов работы отдельных компонентов системы
- Знания связи компонентов друг с другом или архитектуры всей системы
- Знания продукта и пользователей
- Знания лучших практик работы с кодом или инструментами
- Знания новых технологий
- Знания процессов и причин их изменений

Чаще всего эти знания оседают в головах конкретных людей, и незаменимыми сотрудников делают именно они, а не конкретные навыки. Именно поэтому в ветке "Управление компетенциями" придаётся такое значение понятию bus factor – с уходом человека исчезает не только член команды, но и вся накопленная им за время работы над проектом информация.

Роль тимлида в этой ветке – организовать такую систему обмена и распространения знаний, которая одновременно позволяет ценной информации распространяться по всей команде, отсекает лишний шум и не нагружает сотрудника дополнительной головной болью.

Почему ветка важна?

Для тимлида:

- Снижаются риски, связанные с потерей конкретных людей
- Упрощается найм и онбординг
- Повышается точность оценки задач, архитектурных и code ревью

Для команды:

- К работе над задачей или компонентом проще подключить коллег
- Проще разобраться в чужой части системы

Что будет, если её не делать?

- Знания об архитектуре продукта будут только у нескольких человек в команде
- С определённой периодичностью будут возникать споры по одним и тем же вопросам, и никто не будет способен вспомнить, как они решались раньше
- При смене команды или конкретных людей, работающих над проектом, придётся переписать целые компоненты
- Команда не будет учиться на своих ошибках
- Будут изобретаться велосипеды – ведь знания о том, что что-то уже реализовано легко могут быть потеряны

На кого может быть делегирована?

На любого члена команды, в том числе и на не разработчиков

Примеры поведения

Примеры плохого поведения

- На проекте полностью отсутствует документация
- Не ведутся логи встреч, где принимаются решения по процессам или архитектуре
- Люди в команде работают обособленно и не делятся своими мыслями с коллегами
- Разработчики не выходят за пределы подсистемы, разработкой которой занимаются
- Члены команды не делятся новостями и своим опытом использования новых технологий

Примеры хорошего поведения

- В базе знаний можно найти причины принятия всех значимых решений
- Членам команды не приходится сложно искать ответ на одни и те же вопросы чаще двух раз
- Всей командой обсуждается как положительный, так и негативный опыт, полученный в процессе разработки
- Вся основная продуктовая и техническая документация хранится где-то кроме головы тимлида
- В команде нет людей с уникальными знаниями о системе

Способы прокачки

Практика

1. Определите ваши текущие проблемы и боли, связанные с распространением знаний. В этом могут помочь такие инструменты:

- Интервью с членами вашей и других команд с вопросами: "Когда вы последний раз что-то пытались выяснить о нашем проекте и не могли этого найти?", "В каких областях нашего проекта разбирается слишком мало людей?", "Чему вы хотели бы научиться у кого-то из команды?", "Расскажите, как устроена архитектура нашего продукта".
- Изучение вопросов, которые задают в канале вашей команды или продукта в мессенджере. Если какие-то из них повторяются, это сигнал о том, что соответствующей информации нет, либо её тяжело найти.
- Пообщайтесь с новичками и узнайте, с какими сложностями они столкнулись в период онбординга.

2. Постепенно пробуйте разные способы распространения знаний, которые могут помочь вам в решении ваших проблем. Например:

- Ведение базы знаний в Confluence, Quip, Notion или другой системе
- Организация командного Stack Overflow
- Отдельный чат в мессенджере для Q&A
- Регулярные внутренние технические митапы
- Архитектурные и code ревью
- Технический радар
- Рассылки про изменения в технологиях и продукте
- Краткие дайджесты в командном чате
- Общие демо или стендапы

Консультации

- Telegram-чат TL Bootcamp
- Чат конференции KnowledgeConf "Я шарю"

Теория

Статьи

- Без управления знаниями больно
- Управление знаниями через модели компетенций

Подкасты

- Podlodka #103 – Управление знаниями

Видео

- Плейлист докладов KnowledgeConf 2019

Работа с багами

Описание

Это процесс наполнения, приоритизации и разбора багов. Процесс позволяет сделать работу с багами понятной и прозрачной для команды.

Почему ветка важна?

Баги в системе – это неправильное поведение системы. Слишком большое число критичных багов приводит к сильной деградации системы и в конечном счёте отражается на продуктовых метриках.

Что процесс даёт для разных ролей. Для лида: Контроль за состоянием системы. По динамике наполнения и разбора багов можно делать выводы как о проблемах в продукте, так и о работе команды разработчиков.

Для команды:

- Понятный план действий при обнаружении бага.
- Понимание, как неправильное поведение системы будет устраняться.

Для продукта и пользователей:

- Пользователи реже встречаются с неправильным поведением системы, а в критичных кейсах – почти никогда.
- Сообщения от пользователей становятся одной из входных точек процесса разбора багов.

Что будет, если её не делать?

- Непонятно кто и когда будет его разбирать. Разное представление об этом можем вызывать конфликты в команде.
- В процессе разработки системы неизбежно появляются новые баги. Без процесса происходит неконтролируемая деградация.

На кого может быть делегирована?

На старших QA-специалистов в команде или компании. На Scrum-мастеров, продактов.

Примеры поведения

Примеры плохого поведения

- Найденные баги никуда не заводятся и, как следствие, никогда не исправляются.
- Как и куда заводить баги знают только QA-специалисты.
- Все найденные баги должны исправляться немедленно.
- Бэклог растёт значительно быстрее, чем разбирается.
- Задачи от начальства автоматически получают максимальный приоритет вне зависимости от их реальной критичности.
- В бэклоге много старых багов.

Примеры хорошего поведения

- Каждый участник команды понимает, что делать при обнаружении бага.
- Регулярно выделяется время на разбор багов. Бэклог багов не пухнет.

Способы прокачки

Практика

У любой системы работы с багами есть несколько составляющих:

- Бэклог, куда баги заводятся и хранятся.
- Правила, по которым бэклог наполняется.
- Процесс, по которому бэклог разбирается.
- Люди, ответственные за наполнение и разбор багов.

Конфигурации могут быть разными и отличаться в зависимости от структуры команд. Например, в случае с матричной структурой компании, за систему сбора багов и наполнение отвечает отдел QA. Разбор может проходить за счёт периодического разбора багов силами разработчиков. Бывают интересные варианты с багоделью, когда баги разбираются в формате хакатона с элементами геймификации.

Zero Bug Policy

Основная критика многих стандартных подходов с наполнением и периодическим разбором багов в том, что он всё равно пухнет. Обычно так происходит из-за того, что создание новых фичей видится более перспективным для продактов, чем разбор старых багов. Просто выкидывать баги в

силу специфики работы тестировщикам тоже не хочется. В итоге бэклог багов превращается в «антресоль», куда баги попадают, но из которой уже не достаются.

Идея Zero Bug Policy в том, что баги не хранить. Фактически предлагается баги раскладывать на две группы:

- Критичный, потому дёргаем стоп-кран и правим его в этом же спринте.
- Не столь критичный, потому забываем о нём. Если каждый такой баг складировать, бэклог будет только расти.

Опять же, это не означает, что мы снижаем планку качества. Мы признаем, что в нашем продукте есть сколько-то багов и это нормально.

Критика данного метода основывается на следующем:

- Команда QA может начать деградировать. Зачем упираться, если твой баг всё равно могут выкинуть?
- Выброшенные баги могут давать кумулятивный эффект и выстрелить при нахождении бага, который команда решит править.
- Нет источника информации по багам. Каждый раз баги находятся как с чистого листа.

Консультации

- Telegram-чат TL Bootcamp.

Теория

Статьи

Ещё один взгляд на разбор багов Багодельня — марафон по убийству престарелых багов Багодельня: BUgHunting. Как найти 200 багов за день Zero Bug Policy. Нет багов — нет проблем?

Code review

Описание

Code review – это проверка исходного кода на ошибки, проблемы архитектуры.

Почему code review важен?

Помогает:

- Найти баги
- Выявить проблемы в архитектуре
- Сделать код единообразным

А также, что **более важно** в долгосрочной перспективе:

- Это работающий инструмент для обратной связи
- Участники code review будут учиться на своих и чужих ошибках
- Для оценки hard skills разработчика
- Code review поможет делиться знаниями о технологиях, вариантах решения проблем, возможных проблемах и самом проекте в команде
- Даёт приток новых идей для улучшений в процессах, подходах и автоматизации
- Децентрализация знаний

Что будет, если не делать code review?

Будет плохо продукту:

- Баги на production. Тестирование не найдёт все баги
- Технический долг снежным ~~комом~~лавиной накрывает проект. Время разработки новых фич увеличивается экспоненциально
- Подходы и архитектура будут несогласованны. Получится Франкенштейн.

Будет плохо lead-у:

- Плохо знает hard skills разработчиков по отдельности
- Не может оценить производительность

Будет плохо разработчикам:

- Без адекватной обратной связи будет ощущение работы "в стол". Демотивация, депрессия, поиск новой работы
- Не будет притока новых знаний о:
  - Проблемах, с которыми уже столкнулись другие. Учиться на чужих ошибках – это быстро и дёшево
  - Технологиях
  - Вариантах решения проблем
  - Самом проекте

На кого можно делегировать code review?

В code review желательно участвовать всем разработчикам проекта.

Примеры поведения

Примеры плохого поведения

- Токсичное поведение
  - Переход на личности
  - Сарказм
  - Раздражение
- Плохо настроенные процессы
  - Неизвестно, кто должен делать code review
  - Неизвестны критерии прохождения. Процесс может продолжаться бесконечно
  - Список правок не разбит на группы по приоритетам
  - Разработчик долго ожидает code review
  - В merge request приходят огромные фичи
- Software используется недостаточно эффективно
  - Не настроены linter-ы и/или автоматические тесты
  - Разработчики пытаются запомнить список правок
- Разработчику непонятно, зачем нужно вносить правки
- Ревьювер не читал задачу в Jira

Примеры хорошего поведения

- Адекватная обратная связь
  - Уточняющие вопросы вместо прямого указания на ошибки
  - Нейтральность или доброжелательность
- Процессы помогают ускорить и упростить code review
  - Разработчики понимают, что в их интересах делать code review

    - Все знают список требований для прохождения code review
    - Список правок удобным образом приоритизирован. Например, с помощью emoji 🔥 , 💬
    - Обратная связь – быстрая, в идеале – в течение дня
    - Merge requests атомарные
- Для ускорения используются библиотеки и программы
    - Используются linter-ы и автоматические тесты
    - Разработчики не пытаются запоминать список правок в уме
- Участники code review согласны и понимают причины почему нужно внести правки
- Ревьювер знает бизнес-логику решаемой задачи

Способы прокачки

Практика и способы прокачки

Code review выглядит просто. Проверяете merge request на ошибки и пишете о них, но есть нюанс. Важно понять и принять, что это долгосрочный процесс. Настоящие причины ошибок – пробелы в знаниях, сниженная мотивация. Lead должен включать soft skills, чтобы не стрелять в ногу себе и команде.

Полезно проводить "review" до написания кода, особенно для junior devs. Lead должен убедиться, что разработчик напишет задачу верным способом. Выяснять нужно с помощью уточняющих вопросов.

Умение критиковать

Умение критиковать – это ключевой навык.

Как смягчить критику вербально:

- Задавать уточняющие вопросы вместо прямого указания на ошибки
- Сперва похвалить, затем – критиковать
- Хвалите, если всё хорошо
- Feedback от разработчика о том как прошло review. Да, взять и спросить
- Не говорите "ТЫ сделал плохо"

Практиковаться можно "в уме" на merge request. Когда поняли что научились – давать настоящий feedback.

Как смягчить критику невербально:

- Следите за эмоциональными состояниями: своим и разработчика

Практикуйтесь "в уме" на косячных merge request. Вы не должны злиться и раздражаться на "эти тупые ошибки 😡 ". Для настоящих review начинайте с хороших merge requests и постепенно переходите к косячным.

Чтобы не злиться самому помогут:

- Понимание себя и истинных причин раздражения
- Понимание что цель не найти баги, а обучить разработчика не делать их снова; или помочь разработчику сменить позицию или компанию на более подходящую
- Психолог
- ~~Успокоительное~~
- Просто будьте проще, от ваших багов навряд ли кто-то пострадает (c) Капитан Очевидность

Понять что разработчик устал, ненавидит вас, **могут** помочь:

- Практические книги по психологии
- Психологические тренинги

Дополнительно

Без невербального общения (переписка в комментариях github) разработчик может додумать ваши эмоции. Старайтесь быть ближе, как минимум – созвонитесь.

Процесс review фичи

1. Выяснить, какую бизнес-логику писал разработчик
2. Рассмотреть ключевые элементы бизнес логики, архитектуру решения
3. Углубиться в детали

Нужно выяснить какую бизнес-логику писал разработчик

Об этом могут рассказать разработчик, менеджер или таска в jira. Feedback выдаётся разработчику и/или менеджеру при проблемах с бизнес-логикой.

Чтобы прокачать этот скилл можно пройтись по задачам с описанием.

Чтобы получить косячные задачи, можно:

- Зайти в "achieved"
- Попросить придумать косячные задачи
- Или попросить придумать задачи косячного менеджера

Результат – понимание задачи или аргументы что не так с задачей. Возможно – список на исправление.

Рассмотреть ключевые элементы бизнес логики, архитектуру решения

Об архитектуре решения и ключевых моментах расскажет разработчик. Feedback нужно выдать ему же.

Практиковаться можно на merge request вместе с разработчиком.

Разработчик самостоятельно расскажет о проблемах, если его спросить. Также он сможет аргументировать, почему он выбрал определённое решение.

Результат – понимание и принятие или аргументы:

- Почему это нужно исправить
- Что будет, если не исправить
- Соглашение с разработчиком как исправить

Углубиться в детали

На проекте должен быть linter. Он проверяет отступы, использование необъявленных переменных и другую головную боль.

Если это необходимо – пройдитесь по деталям реализации.

Результат – статус задачи "review закончено" или аргументированный список правок.

Что ревьюить

В зависимости от целей ревью и времени на него:

- Наиболее критичные задачи
- Практики безопасности
- Архитектуру
- Задачи junior devs
- Все задачи
- Выбранное вами

Вы должны чётко осознавать, что именно и зачем вы ревьюите.

Настройка и использование software

Автоматическая проверка кода

Для ускорения процесса нужно настроить проверку кода на:

- Синтаксические ошибки
- Стилистические неточности
- Предполагаемые ошибки во время исполнения (использование необъявленной переменной и тд)
- Возможные уязвимости в коде и используемых библиотеках

Это поможет сократить время на поиск проблем вручную.

Комментарии

Не полагайтесь на память. Git хостинги дают комментировать участки кода, вести дискуссии. Критичные правки можно записывать например в комментарии к merge request.

Можно настроить автоматическую синхронизацию комментариев с мессенджером.

Интеграции

- Git хостинги умеют слать на почту сообщения, связанные с code review
- Можно добавить интеграцию через API, чтобы слать сообщения в мессенджер
  - О новых merge requests
  - Об утверждённых merge requests
  - О merge requests которые долго никто не проверял
  - Комментариях к коду

Работа с другими процессами

- **CI/CD**. Если встроить code review в процесс CI/CD, то на выходе можно будет получать более надёжный продукт

Вопросы для собеседования

- Blitz
  - Что такое code review?
  - Организован ли у вас в команде code review?
  - Может ли code review помочь найти баги, которые не может найти тестировщик?
- Вопросы "на подумать"
  - Какие плюсы даёт code review в общем и вашей команде?
  - Можно ли не делать code review? В каких случаях?
  - Сколько времени можно тратить на code review?
  - С какими процессами в команде можно интегрировать code review?
  - Во время code review вы увидели токсичное поведение одного из ревьюеров; например, сарказм, грубость. Нужно ли с этим что-то делать? Если да, то что?

Консультации

- Telegram-чат TL Bootcamp
- Психологи на b17.ru

Теория

Статьи

Раскрывают тему:

- Code review по-человечески (часть 1)
- Code review по-человечески (часть 2)
- Pull Requests and Code Review
- Unlearning toxic behaviors in a code review culture

Управление инцидентами

Описание

Управление инцидентами – минимизация негативного воздействия внезапных перерывов в обслуживании или снижений качества продукта путём восстановления нормальной работы продукта в кратчайшие сроки.

Почему важно управление инцидентами?

Управление инцидентами помогает:

- Влиять на восприятие продукта, его качества и формировать ожидания.
- Рационально использовать ресурсы при решении, ранжируя инциденты по степени влияния на продукт:
  - Инцидентами с незначительным воздействием нужно управлять рационально, чтобы они не потребляли слишком много ресурсов.
  - Инциденты с более серьёзным влиянием на продукт требуют большего объёма ресурсов и более сложного управления.
  - Для управления крупными инцидентами, а также для управления инцидентами информационной безопасности часто используют отдельные процессы.
- Информировать заинтересованные команды и специалистов о состоянии затруднений при эксплуатации продукта.
- Обеспечить эффективность взаимодействия команд при работе над инцидентами с помощью инструментов общения.
- Предотвращать повторение инцидентов, благодаря информации накопленной в ходе решения предыдущих инцидентов.
- Обеспечить своевременный выбор методики решения инцидента:
  - Последовательные процедуры восстановления по готовой методике.
  - Привлечение команды со знаниями и опытом в конкретной предметной области.
  - По методу коллективного анализа под названием "Рой", когда на начальном этапе подразумевается совместная работа нескольких команд разных компетенций, пока не выяснится, кому лучше всего продолжить работу над решением.

Что будет, если не управлять инцидентами?

- Негативное восприятие продукта:
  - Повторяющиеся проблемы при эксплуатации продукта.
  - Длительные сроки восстановления после сбоев.
  - Отсутствие заготовленного временного решения-костыля для минимизации негативного влияния на продукт во время поиска окончательного решения.
- Нерациональное использование ресурсов при решении:
  - Многократное повторение процесса поиска решения, которое уже было ранее найдено при решении предыдущих инцидентов из-за отсутствия накопленной структурированной информации по ранее решённым инцидентам.
  - Использование недостаточных ресурсов для попытки решения крупных инцидентов.
  - Задействование массивных и дорогих ресурсов при решении незначительных инцидентов.
- Невозможность решения некоторых инцидентов вовсе:
  - Отсутствует индивидуальная для конкретного продукта методика решения.
  - Нет связующей информации о компетенциях команд подходящих для решения сложного инцидента.

На кого может быть делегировано управление инцидентами?

Управление инцидентами подразумевает привлечение широкого круга участников процесса, включая команды специалистов, пользователей, руководство, юристов, поставщиков и сторонние организации, если в части продукта задействованы их услуги.

Примеры поведения

Примеры плохого поведения

- Несвоевременное и недостаточно полное внесение информации о ходе решения инцидента или не внесение такой информации вовсе приведёт к нерациональному решению аналогичных инцидентов в будущем или даже к невозможности решения
- Отсутствие ранжирования инцидентов по тяжести и степени влияния на продукт
- Выбор неэффективного метода решения сложного инцидента, если не подошла ни одна из ранее описанных процедур, без привлечения команды с соответствующей компетенцией, поддержки поставщика или коллективного анализа, при необходимости.
- Фокус на поиске виновного.

Примеры хорошего поведения

- Использовать подходящий инструментарий для управления инцидентами: с содержанием в записях об инцидентах блоков и ссылок с описанием влияния на компоненты конфигурации

продукта, связанных проблем, известных ошибок и другой информации, чтобы обеспечить быструю и эффективную диагностику и восстановление.
- Подключить к работе над инцидентами специалистов поддержки поставщика, в случае необходимости. Заранее подготовить соответствующие пункты контрактов.
- Инициировать создание, использование, постоянное и качественное заполнение справочной системы для возможности решения инцидентов самим пользователями в момент возникновения.
- Формализовать процесс регистрации и управления инцидентами, чтобы обеспечить повышение эффективности исследования и диагностики инцидентов.

Практика

Начать внедрение управления инцидентами следует с обязательной фиксации каждого инцидента и внесения подробной информации с описанием хода решения, так появится возможность накапливать базу знаний и связывать части информации и быстро находить нужное в ранее решённых инцидентах, тем самым увеличивая скорость решения и снижая негативное влияние на продукт. С ростом продукта следует формализовать процедуры работы с инцидентами для одинакового восприятия информации и предпринимаемых действий всеми вовлечёнными командами — так возможно объединение совершенно разных команд с различными компетенциями для слаженной работы над инцидентами.

Теория

Книги

- ITIL 4 Foundation, 2019
- Digital Swarming “The Next Model for Distributed Collaboration and Decision Making”, Cisco Internet Business Solutions Group (IBSG), 2008
- Intelligent Swarming: A Framework for Collaboration, 2019
- Site Reliability Engineering: How Google Runs Production Systems (главы 14 и 15)

Нефункциональные требования

Описание

При разработке новой функциональности часто приходится делать выбор – сделать ли её более производительной или более поддерживаемой, более надёжной или более безопасной. При поддержке уже написанной системы тоже встают подобные вопросы, например, что именно кроется под понятием "высокая доступность". Разобраться в этом помогают описанные нефункциональные требования.

Нефункциональные требования – это чёткие критерии того, **как** система должна работать, в отличие от функциональных, которые описывают, **что** система должна делать. Давайте посмотрим на пример.

> API метод должен возвращать список ресторанов в короткой форме: id, название, адрес

Это функциональное требование, оно описывает поведение системы.

> API метод должен отдавать данные не более чем за 200ms на 95 перцентиле и не более чем за 500ms на 99 перцентиле.

А это уже нефункциональное требование, которое описывает определённый атрибут качества – performance.

Почему ветка важна?

Для тимлида:

- Возможность описать поведение системы, за которую он отвечает, в чётких параметрах, и понимать, соответствует ли она им

Для команды:

- Неопределённые желания заказчиков вроде "работает быстро" преобразованы в чёткий список требований
- Проще достигать согласия при спорных вопросах проектирования – вместо субъективных домыслов для аргументации используются конкретные требования

Что будет, если её не делать?

- Если обоснованные нефункциональные требования отсутствуют на этапе проектирования, то в будущем может быть невозможно начать им соответствовать без полного переписывания
- Система может получиться медленной, небезопасной и не соответствующий всем прочим атрибутам качества
- Архитектурные решения будут приниматься не на основе того, как должен работать продукт, а на основе личных предпочтений команды
- Сложно настроить разумный мониторинг, и понять, что является желаемым поведением

На кого может быть делегирована?

На специалиста по контролю качества, архитектора.

Примеры поведения

Примеры плохого поведения

- Атрибуты качества не описаны в виде нефункциональных требований
- Нефункциональные требования взяты из головы и не подтверждены реальным смыслом и необходимостью
- Нефункциональные требования выписаны на бумажку, но никак не проверяются на работающей системе
- Нефункциональные требования не учитываются при тестировании

Примеры хорошего поведения

- Для разрабатываемой системы определены требуемые атрибуты качества
- На основе сравнения с конкурентами, изучения поведения пользователей и требований соседних подсистем определены конкретные нефункциональные требования
- Проверка на соответствие нефункциональным требованиям проводится с помощью синтетических тестов в CI
- Production мониторинг ориентируется на список нефункциональных требований
- На всех архитектурных ревью система нефункциональных требований используется для принятия решений

Способы прокачки

Консультации

- Telegram-чат TL Bootcamp

Теория

Статьи

- Software Quality Attributes, Non-Functional Requirements and Better Software Architecture
- Systems and software engineering — Systems and software Quality Requirements and Evaluation
- 12 software architecture quality attributes

Пирамида тестирования

Описание

Пирамида тестирования — один из способов обеспечения качества ПО, визуализация, которая помогает группировать тесты по типу их назначения. Так же, позволяет согласовать правила написания тестов, разделения их на типы, обозначить основной фокус тестирования в каждой из групп.

Цель использования данного процесса:

- Экономия времени и ресурсов в процессах обеспечения качества
- Сократить количество сложных кейсов тестирования
- Группировка и типизация тестов
- Снижение рисков возникновения критических дефектов ПО
- Формализация тех. долга в части QA
- Формализация правил написания тестов
- Формализация требований к кодовой базе: code coverage, тестовые планы
- Автоматизация рутинных и частых операций
- Чётко-регламентированная изоляция ресурсов в зависимости от вида тестирования

Почему ветка важна?

Для менеджера:

- Увеличение скорости разработки в среднесрочной перспективе
- Сдвиг этапа обнаружения дефектов на более ранние стадии разработки
- Позволяет варьировать затраты на инфраструктуру, время разработки в зависимости от требуемого качества
- Возможность формализовать требования к тестированию ПО, соотношения ручных/автоматических тестов

Для разработчика:

- Безболезненный red/green refactoring
- Совместим практически со всеми практиками разработки ПО: TDD/BDD/DDD etc.
- Позволяет экономить время на запуск тестов
- В связке с практиками CI/CD позволяет быстро определять работоспособность критически важных фич

- Помогает предотвратить каскадные ошибки
- Снижает уровень связанности кода, улучшает модульность, помогает формализовать API модульного/сервисного взаимодействия

Для тестировщика:

- Небольшие изменения могут попадать в production без ручного тестирования
  - экономия времени тестировщиков
  - минимизация человеческого фактора на рутинных операциях
  - оптимизация time to market
- задачи в тестирование поступают хорошего качества
  - реже возникают ошибки из-за невнимательности
  - меньше циклов доработки-тестирования
  - тестировщик не вынужден тестировать одну и ту же фичу несколько раз
- простые ошибки выявляются тестами на ранних стадиях
  - отдел QA занимается более сложными кейсами тестирования
  - меньше тестировщиков могут обеспечивать такое же качество ПО
- количество GUI-тестов, как правило меньше
  - их проще поддерживать
  - тестируется только критическая функциональность, которая не был протестирована другими группами автотестов

Что будет, если её не делать?

- Увеличится среднее время обнаружения дефекта
  - Больше дефектов будет попадать в контуры промышленной эксплуатации (production)
  - Нестабильные релизы
  - Неконтролируемый тех. долг/лавинообразный рост багов даже при минорных изменениях
- Возрастут затраты на ручное тестирование
  - Рутина составления тест. планов
  - Возрастёт штат ручного тестирования
  - Нет чёткой структуры тестов
- Каждое изменение кодовой базы может требовать полный регресс тестирования всего модуля

На кого может быть делегирована?

Распространённая практика: разные группы тестов делегировать разным группам людей. Например модульные и интеграционные тесты могут быть отданы разработчикам, а end-to-end тесты и ручные тесты в отдел Quality Assurance.

Примеры поведения

Примеры плохого поведения

- Пользоваться моделью рожка мороженого (перевёрнутая пирамида тестирования). Модель, когда в основу вашего процесса тестирования закладывается большой ресурс на ручное тестирование
  - так же, вместо ручного тестирования используются библиотеки автоматизации GUI тестирования (пр. Selenium/Puppeteer), которые составляют основную базу тестовых кейсов. Данный подход считается антипаттерном
- Не писать авто-тесты в целом - проверка качества ПО смещается в сторону ручного тестирования
- Писать тесты, не опираясь на "хорошие практики". Антипаттерны:
  - unit тест в том числе тестирует смежные модули
  - интеграционный тест тестирует данные, а не бизнес логику
  - тесты не проверяют исключительные ситуации (exceptions) в ответах сервисов
  - тест-кейсы, который тестирует все, в том числе внутренние процессы реализации и бизнес логику одновременно
  - тесты, которые не выводят понятный assertion message
  - нестабильные тесты
- Не проводить внутренний аудит актуальности тестов
- Не запускать тесты
  - или запускать тесты в не изолированном окружении
  - или запускать все группы тестов на каждое изменение кода

Примеры хорошего поведения

- Писать unit-тесты перед реализацией - подход TDD
- Покрывать тестами ранее непокрытый код - снижение рисков при его модификации и рефакторинге
- Включать в Definition Of Done и Definition Of Ready требования по написанию автотестов
- Использовать механизмы статического и динамического анализа кодовой базы
- Производить автоматический расчёт Test Code Coverage и настраивать Quality Gates для автоматических проверок на соответствие минимальным критериям к кодовой базе

Способы прокачки

Практика

Если вы начинающий тимлид и теперь в ваши обязанности в том числе входит обеспечение качества продукта, то у вас возможно несколько ситуаций:

- код разрабатывается с нуля
- код достался вам по наследству
  - в нем были тесты
  - в нем не было тестов

С чего начать

1. Прежде чем советовать своим подчинённым, как именно следует писать тесты, вы должны в совершенстве овладеть навыком написания Unit-тестов самостоятельно. Это базовый навык, без которого сложно двигаться дальше. Таких тестов нужно будет писать много и не задумываясь.
2. Разберитесь в различиях между терминами mock/stub/spy, посмотрите различные варианты структуризации тестов: Arrange Act Assert, GivenWhenThen и другие
3. Разберитесь в возможностях библиотек для тестирования, как они генерируют отчёты, какие возможны результаты выполнения теста Success/Error/Broken, есть ли возможность группировать тесты и каким образом. Данные знания пригодятся вам во время дальнейшей структуризации
4. Старайтесь изолировать окружение теста, чтобы тестируемые функция/сервис не зависели от окружения (которое не используется в тестах), времени суток и прочих сторонних факторов
5. Подготовьте несколько простых и средней сложности примеров с тестами, обсудите эти тесты с командой, убедитесь, что все понимают правила написания хороших тестов
6. Когда практика написания тестирования начнёт применятся, необходимо настроить автоматический запуск этих тестов. Например в рамках CI или commit hooks. Таким образом вы сможете отслеживать ситуации, когда автоматические тесты спасли ваш продукт от дефекта. Старайтесь акцентировать внимание команды на том, что тесты полезны
7. Настройте автоматический расчёт Code Coverage, настройте Linters и другие механизмы статического и динамического анализа кода. Если у вас будет порядок на данном этапе, то можно двигаться в сторону более сложных тестов

С другими видами тестирования логика обстоит аналогичным образом, однако часто используются различные DSL, которые упрощают читаемость этих тестов. Старайтесь тестировать бизнес логику отдельно взятого сервиса/группы сервисов. Ознакомьтесь со статьями внизу страницы, там можно найти различные подходы к тестированию бизнес логики.

Для каждого уровня тестирования уже существует множество библиотек, которые упрощают разработку. Ознакомьтесь с существующими решениями, они сэкономят вам много времени, тем не менее, рано или поздно, вам все же придётся писать свои библиотеки и обёртки для тестирования.

####Примеры готовых решений для различных уровней тестирования:

- Генератор stub сервисов, и rest клиентов, для задокументированных openAPI
- Автотесты вашего API при помощи PostMan
- Тестирование layout вашего web приложения

Что делать если проект разрабатывался без тестов

Ознакомьтесь с рекомендациями выше и найдите способ мотивации сотрудников писать тесты. Мотивация может быть разной, например вы можете предложить покрывать тестами только изменённый код.

Т.о. вы начнёте тестировать в первую очередь тот код, который активно дорабатывается, и, как следствие, более подвержен к появлению новых ошибок. Договоритесь о минимальном code coverage для изменённого и нового кода, современные static analysis tools позволяют рассчитывать данные метрики автоматически (пр. SonarQube)

Консультации

- Telegram-чат TL Bootcamp.

Теория

Принцип пирамиды тестирования

Пирамида тестирования, в том числе, помогает наглядно объяснить причины, почему количество Unit тестов должно быть больше чем интеграционных. Части треугольника закрашенные разными цветами подразумевают количество необходимых тестов данной категории, чем больше площадь, тем больше тестов. Чем ниже находятся на пирамиде тесты, тем:

- проще и быстрее они разрабатываются
- ниже затраты на поддержку тестов
- быстрее скорость запуска атомарного теста
- выше уровень изоляции компонент между собой
- меньше нужно денег на содержание инфраструктуры для запуска этих тестов
- ниже уровень необходимой квалификации того, кто эти тесты может разрабатывать

Статьи

Раскрывают тему:

- Пирамида тестирования, как категоризовать и писать авто-тесты

- Антипаттерны тестирования ПО

Дополнительные материалы:

- Исправление недостатков пирамиды: Модель круглой земли (рус)
- Обзор различных групп тестов, библиотек и практик их написания

Написание кода

Описание

Тимлид участвует в решении рабочих задач команды какую-то часть своего времени. Это может быть написание кода, выполнение задач по администрированию, подготовка тест-кейсов – короче говоря те действия, которые являются рутинными для других членов его команды. Это помогает следующим целям:

- Вносить свой вклад в конкретный результат команды, выраженный закрытыми ими задачами
- Поддерживать связь с реальностью, не отрываясь от суровой реальности своей команды

Почему ветка важна?

Для тимлида:

- Не теряет уважение и доверие коллектива
- Самостоятельно встречается с теми же проблемами, что и его команда
- Видит простор для улучшений и оптимизаций процессов
- Учит своих сотрудников на практике

Для сотрудника:

- Вовлечённый в работу руководитель мотивирует своим примером
- С руководителем проще найти общий язык

Для компании:

- Играющий тренер – удобный способ закрытия рисков в команде в случае чьего-то ухода

Что будет, если её не делать?

- Тимлид может начать восприниматься членами команды как "эффективный менеджер", который оторван от реальности
  - Просядет доверие в команде, что повлияет на её эффективность
  - Тимлиду будет сложно управлять подчинёнными
  - Могут возникнуть серьёзные конфликты между тимлидом и командой

- Тимлид потеряет возможность делать независимые выводы о ситуации в команде, и ему придётся во всем полагаться на мнения подчинённых
  - Качество решений упадёт
- Технические знания тимлида постепенно потеряют актуальность
  - Сложнее будет развивать и собеседовать своих сотрудников
  - Просядет качество принимаемых технических решений

На кого может быть делегирована?

Не делегируется.

Примеры поведения

Примеры плохого поведения

- Тимлид никогда не пишет код
- Тимлид берет на себя задачи, находящиеся на критическом пути, и из-за его занятости в других ролях команда проваливает свои цели
- Тимлид большую часть своего времени занимается написанием кода

Примеры хорошего поведения

- Тимлид регулярно берет в работу не критичные для целей команды задачи
- Тимлид ориентируется во всех составляющих частях системы, за которую он отвечает, и способен взять задачу из любой области

Способы прокачки

Практика

1. Подумайте, какие цели перед вами стоят. Это может быть повышение bus factor в команде, обучение членов команды, изучение ранее неизвестных вам частей системы.
2. При планировании выбирайте себе такие задачи, которые:

- Отвечают поставленным вами целям
- Полезны для команды
- Не являются блокирующими или критичными – если вы их не сделаете в срок, никаких серьёзных последствий не будет.

Подходящими вариантами могут быть автоматизация каких-то рутинных действий, решение старого технического долга.

Консультации

- Telegram-чат TL Bootcamp

Теория

Статьи

- Должен ли тимлид писать код?

Выбор и контроль технологий

Описание

Периодически команда разработки начинает использовать новые технологии. Это может происходить как по рациональным причинам – начинаете решать новый тип задач, модернизируете архитектуру, так и не очень – разработчикам хочется попробовать на практике то, о чем они услышали на конференции. Роль тимлида заключается в том, чтобы отвечать за рациональность используемого стека технологий и построить процесс их ввода и вывода из использования.

Под технологиями здесь понимается всё – языки, фреймворки, библиотеки, инструменты.

Почему ветка важна?

Для тимлида:

- Без контроля за тем, какие технологии используются, область ответственности может быстро выйти из под контроля Тимлид обладает более широким контекстом, чем разработчики, чтобы оценить риски от внедрения новых технологий

Для команды:

- Если есть понятный процесс тестирования и внедрения новых технологий, это может быть хорошим кандидатом в качестве личных целей

Для компании:

- Упрощается унификация знаний между разными командами разработки и появляется возможность перебрасывать людей между командами
- Понижается риск того, что важный компонент системы будет написан с использованием чего-то экзотичного

Что будет, если её не делать?

- Команда разработки будет внедрять новые технологии исходя только из своей заинтересованности
  - Усложняется найм людей, так как на входе требуется больше знаний

    - Повышаются риски безопасности, так как больше мест для нахождения уязвимостей
    - Сложнее перекидывать людей между проектами и компаниями, так как навыки не будут универсальными
    - Система в целом становится менее поддерживаемой
- Внедрение любых новых технологий может быть под запретом
    - Усложняется найм людей, так как им не будет интересно копаться в старых технологиях
    - Понижается мотивация существующей команды
    - Не используются более эффективные способы решения старых задач

На кого может быть делегирована?

На самых опытных разработчиков отдельных платформ или направлений.

Примеры поведения

Примеры плохого поведения

- Разработчики с порога могут втаскивать в production любую технологию
- Процесс принятия решения о внедрении новых технологий слишком бюрократизирован
- Когда принимается решение об отказе от использования какой-то технологии, нет плана по полному избавлению от неё
- При принятии решения о внедрении технологии учитывается мнение только одной группы – разработчиков, архитекторов, менеджеров или кого-то ещё

Примеры хорошего поведения

- Есть чек-лист вопросов, на которые надо ответить при вводе или выводе технологии из стека
- Есть готовые подходы по тому, как можно аккуратно попробовать новую технологию и дёшево от неё отказаться
- Технический стек компании актуален на рынке
- Если появляется новая технология, которая может принести проекту пользу, её внедрение не является сложной проблемой
- Вся команда знает, как в проект можно внедрить новую технологию

Способы прокачки

Практика

Технологический радар — это набор практик, описывающих жизненный цикл технологии, и инструмент визуализации текущего состояния технологического стека. Радар помогает ответить

на ряд вопросов. Вот примеры:

- Почему мы не используем технологию X?
- Как мы относимся к модной технологии Y?
- Что стоит использовать в разработке нового сервиса?
- На какие технологии мне стоит сделать упор в саморазвитии?
- Какие технологии и почему не востребованы в моей компании?

По сути, это диаграмма состояний технологий. которая отображает:

- Текущий статус использования технологии в компании
  - ASSESS – присматриваемся, но пока не используем в production, аккуратно тестируем на маленьких проектах
  - TRIAL – тестируем, но не на критичном компоненте
  - ADOPT – смело используем в production
  - HOLD – по какой-то причине перестали использовать
- Принадлежность к категории
  - Языки
  - Фреймворки
  - Инструменты для работы с данными
  - Процессы

Создание радара

1. Соберите группу экспертов в определённой платформе или направлении (iOS, Android, backend, frontend)
2. Попросите каждого на стикерах перечислить все технологии, которые сейчас используются в вашем продукте, от которых когда-то отказались, или к которым хочется присмотреться
3. Объедините получившиеся стикеры по категориям. Это могут быть языки, фреймворки, библиотеки – что угодно, что имеет для вас смысл.
4. Внутри категории разбейте все стикеры по статусам – Assess, Trial, Hold, Adopt.
5. Для каждой из технологий постарайтесь собрать краткую историю того, как она оказалась в этом статусе.
6. Соберите с помощью конструктора свой собственный радар и внесите туда всю информацию.
7. Повторите этот процесс для других платформ.

Использование радара

1. Кратко опишите процессы перехода технологии между статусами. Это удобно делать с помощью чек-листов.
2. Договоритесь о том, какие люди в компании принимают решение о переходе технологии между статусами. Это может быть CTO или комитет из экспертов.

3. Периодически пересматривайте текущее состояние радара и проверяйте, что он соответствует действительности.

Пример требований для перехода между статусами

Из состояния неопределённости в ASSESS

Тот, кто хочет внедрить новую технологию, отвечает на вопросы:

- Задачи, которая должна решать указанная технология в компании
- Action Plan внедрения технологии: шаги, сроки, критерии готовности
- Конкуренты технологии внутри компании и снаружи, преимущества и недостатки по сравнению с конкурентами
- Оценочный уровень зрелости технологии
- Оценочное количество людей, знакомых с технологией внутри компании
- Оценочное количество людей, знакомых с технологией на рынке труда в России

Решение принимает тимлид команды, в которой планируется внедрение.

Из ASSESS в TRIAL

Технология должна быть опробована на каком-то тестовом проекте, который позволит понять её плюсы и минусы. За ревью результатов тестирования отвечает тимлид команды, в которой проводилось внедрение, плюс один или два эксперта из соседних команд.

Из TRIAL в ADOPT

Технология должна быть опробована в production той командой, которая начала её внедрение. За ревью результатов этого этапа отвечают руководители направлений и CTO.

Из любого статуса в HOLD

Тот, кто хочет избавиться от технологии, должен ответить на вопросы:

- Почему надо прекратить использовать технологию
- Кого в компании касается это решение
- На что технология будет заменена

Консультации

- Telegram-чат TL Bootcamp

Теория

Статьи

Раскрывают тему:

- Build Your Own Technology Radar

Дополнительные материалы:

- Радар технологий: перечень языков, инструментов и платформ, которые прошли через руки Lamoda
- Avito Tech Radar

Прочие материалы

- StackShare – база данных технологических стеков разных компаний

Знание технологического стека команды

Описание

Тимлид может руководить разработчиками как одного профиля, как обычно встречается в функциональной структуре организации, так и разных, как бывает в кроссфункциональных командах. Для того, чтобы управлять командой, тимлиду нужно ориентироваться в технологическом стеке этой команды – языках, фреймворках, инструментах, экосистеме, архитектуре, лучших практиках.

В случае функциональной команды обычно не возникает много вопросов, так как руководителем её часто становится опытный разработчик того же технологического стека. Все обстоит гораздо сложнее для кроссфункциональных команд, где одновременно нужно ориентироваться в iOS, Android, Frontend, Backennd, тестировании, базах данных и часто в чем-то ещё.

Почему ветка важна?

Для тимлида:

- Можешь оценивать качество технических решений
- При необходимости можешь закопаться в работу конкретного сотрудника и оценить качество его работы
- Получаешь возможность спокойно общаться со стейкхолдерами команды без необходимости привлечения кого-то из разработчиков
- Понимаешь обоснованность сроков разработки или опасность технического долга
- Находишь общий язык со всеми разработчиками и не выглядишь для них бесполезной прослойкой

Для команды:

- С технологически грамотным тимлидом проще договориться по вопросам сроков, внедрения новых технологий или закрытия технического долга
- Тимлид, обладающий более широким контекстом в вопросах понимания стратегии бизнеса и продукта, даёт полезный взгляд на архитектурные компромиссы

Для отдельного сотрудника:

- Тимлид может оценить твой уровень не только по внешнему эффекту, но и по сути сделанной работы

Что будет, если её не делать?

- Тимлид не может оценить оправданность сроков разработки
  - В команде могут проявиться конфликты из-за сроков
  - Разработчикам придётся идти на неприемлемые компромиссы, из-за чего резко вырастет количество технического долга и упадёт качество проекта
- Тимлид не может оценить долгосрочное влияние принимаемых решений
  - Решения принимаются неоптимальные и выстреливают команде в ногу через какое-то время
  - Команда будет заниматься не тем, что действительно полезно, а тем, что интересно или весело
- Тимлид не может работать автономно и для ответа на любые технические вопросы привлекает кого-то из команды
  - Команда не будет считать его своим лидером
  - Стейкхолдерам тимлид будет казаться некомпетентным
- Тимлид не может оценивать качество работы своей команды
  - Повышения раздаются несправедливо

На кого может быть делегирована?

Может быть делегирована на отдельных разработчиков в команде, которые становятся "чемпионами" своей платформы или функции.

Примеры поведения

Примеры плохого поведения

- Тимлид не пытается вникнуть в суть принимаемых в его зоне ответственности технических решений
- Тимлид не пытается разобраться в базовых особенностях разработки всех компонентов системы, за которую он отвечает
- При обсуждении любых технических вопросов тимлид не справляется и вызывает кого-то из разработчиков
- Все вопросы, касающиеся архитектуры или планирования незнакомой тимлиду платформы, проходят мимо внимания тимлида
- Тимлид пытается стать экспертом во всех областях, за которые он отвечает

Примеры хорошего поведения

- Тимлид не стесняется своей некомпетентности и со временем пытается его уменьшить
- Тимлид задаёт простые вопросы, чтобы разобраться в проблеме на незнакомом стеке хотя бы на базовом уровне
- Тимлид регулярно решает простые задачи из разных функциональных областей
- Тимлид одинаково погружён во все технологические области, включая и ту, в которой у него есть собственный background

Способы прокачки

Практика

1. Составьте карту знаний, которых вам не хватает. В этом может помочь инвентаризация всей системы, за которую вы отвечаете, или открытое обсуждение со всеми разработчиками в команде. Чаще всего они спокойно отвечают на вопрос "Чему я должен научиться, чтобы разобраться в твоей системе?" и могут посоветовать способы эти навыки прокачать.
2. Постепенно обучайтесь по этой карте. Ваша задача – не стать экспертом во всех технологиях, а разобраться в них на базовом уровне, понимать плюсы и минусы.
3. Берите простые задачи из бэклога команды, которые помогут самостоятельно что-то сделать в каждой из вверенных вам технологических областей. Это поможет попробовать всё на практике, разобраться в инструментах. А как бонус – ещё и доверие команды поднимется.
4. Не стесняйтесь задавать очень глупые вопросы на обсуждениях сложных для вас областей. Такие вопросы часто помогают увидеть какие-то фундаментальные просчёты или новые возможности.
5. При общении с разработчиками незнакомой вам области выписывайте все незнакомые термины, а после этого пытайтесь в них самостоятельно разобраться.

Консультации

- Telegram-чат TL Bootcamp

Теория

Подкасты

- Podlodka #57 — Head of Mobile

Рефакторинг

Описание

Контролируемый процесс улучшения кода, без написания новой функциональности. Обычно выделяют следующие цели:

- Чистый код
- Простой дизайн
- Уменьшение технического долга

Почему ветка важна?

Рефакторинг – это прежде всего процесс, которым необходимо заниматься постоянно. Это позволяет не накапливать технический долг до такой степени, что часть системы перестаёт быть управляемой, что в свою очередь приводит к необходимости переписывания с нуля. Чистый код позволяет расширять функциональность максимально быстро и качественно.

Рефакторинг приводит к позитивным изменениям как для членов команды, так и для лида. Для члена команды:

- Упрощение последующего расширения системы.
- Простота в понимании подробностей реализации.

Для лида:

- Снижение временных рисков при последующем расширении соответствующей части системы.
- Снижение негатива членов команды при работе с кодом.

Что будет, если её не делать?

Рефакторинг – один из основных способов уменьшения технического долга. Если регулярно не проводить рефакторинг:

- Накопится слишком много технического долга.
  - Как следствие, возможна потеря контроля над частью системы.
- Ощущение постоянной работы в легаси.
  - Влияет на найм и удержание сотрудников.

На кого может быть делегирована?

Идеальная ситуация: каждый разработчик проводит рефакторинг по ходу решения текущих задач.

Примеры поведения

Примеры плохого поведения

- Рефакторинг рассматривается как проект и проводится раз в большой промежуток времени.
- Рефакторинг затрагивает функциональность системы.
- Подходами к рефакторингу обладают не все члены команды.
- Правку багов относят к рефакторингу и используют те же инструменты.

Примеры хорошего поведения

- По ходу ежедневной работы каждый разработчик переключается между режимами: разработка/рефакторинг.
- Рефакторинг делается при помощи тестов.

Консультации

- Telegram-чат TL Bootcamp

Теория

Книги

- Refactoring: Improving the Design of Existing Code, Martin Fowler

Управление техническим долгом

Описание

Технический долг – это несделанная в проекте работа, которая будет мешать его развитию в будущем, если так и не будет выполнена. В технический долг не включаются баги или отложенные низкоприоритетные фичи. Технический долг – это, например, плохо спроектированная архитектура или запутанный код.

Управление техническим долгом – это его постоянный поиск, подсчёт стоимости и постепенное устранение.

Почему ветка важна?

- В будущем команде проще будет поддерживать разрабатываемый ею проект
- Появляется возможность прогнозировать стоимость технических компромиссов
- Весь текущий технический долг переводится в разряд видимого и известного

Что будет, если её не делать?

- Будет расти время на разработку и стоимость поддержки системы
  - Усложняется анализ кода, требуется много времени, чтобы разобраться в нём
  - Тяжелее вносить изменения – системы с техническим долгом отличаются хрупкостью
- В какой-то момент станет невозможной дальнейшая поддержка системы
  - Её придётся выводить из использования или переписывать с нуля

На кого может быть делегирована?

На ответственного за проект разработчика или архитектора

Примеры поведения

Примеры плохого поведения

- Программист делает ошибки при разработке проекта, которые не отлавливаются на code review или статическим анализом

- Когда ситуация вынуждает осознанно писать код быстро и некачественно, к нему в будущем не возвращаются для рефакторинга
- Объем технического долга не известен руководству
- В команде не выделяется время на периодическое исправление технического долга
- Разработчики игнорируют мелкие дефекты качества и не пытаются их исправить на месте по правилу бойскаута

Примеры хорошего поведения

- В каждом спринте выделяется определённый процент времени на решение технического долга
- Весь крупный технический долг инвентаризирован
- Неумышленный технический долг отлавливается ручными и автоматизированными проверками качества

Способы прокачки

Практика

Контроль качества внешним аудитом

Для того, чтобы оценить текущее состояние технического долга на проекте, можно привлечь внешних консультантов. Этот способ часто применяется при передаче проекта от одной компании к другой или при полной смене команды разработки.

Минусы:

- Внешние аудиторы не погружены в специфику проекта
- Это разовая проверка, которая позволяет получить только текущий срез состояния проекта
- Это довольно дорого
- Результат аудита сильно зависит от компетентности людей, которую трудно проверить

Плюсы:

- Глаз внешних людей не замылен

Code Review

Процесс code review детально описан в отдельной ветке.

Автоматизация оценки качества кода

Лучше всего бороться с кратковременным техническим долгом позволяет автоматизация проверки качества кода в CI. Для этого можно использовать подходящий для вашего языка

статический анализатор. Точно стоит посмотреть на SonarQube – скорее всего там уже есть подходящий вам плагин.

Минусы:

- Статические анализаторы не спасут от того, чтобы инвентаризировать крупный технический долг самостоятельно – о пробелах в вашей архитектуре и выбранном стеке технологий они не скажут
- Чтобы автоматическая оценка приносила пользу, нужно делать так, чтобы её показатели со временем улучшались, а обозначенный ею технический долг решался
- Численный показатель, который выдаётся наружу такими системами, скорее всего, очень далёк от реальности

Плюсы:

- Отлавливается много мелких ошибок и проблем, которые можно пропустить на code review
- Не отнимает много времени, не считая первичной настройки
- Можно задавать гибкие правила для своей команды
- Можно отслеживать увеличение или понижение технического долга

Консультации

- Telegram-чат TL Bootcamp

Теория

Статьи

- Технические долги
- Wikipedia on Technical Debt
- Martin Fowler on Technical Debt
- TechnicalDebtQuadrant
- Управление техническим долгом - Концепция Continuous Inspection

Подкасты

- Podlodka #77 – Технический долг

Видео

- Debt Metaphor

Unit-тестирование

Описание

Unit-тестирование (модульное тестирование) — процесс в программировании, позволяющий проверить на корректность отдельные модули исходного кода программы, функции или методы.

Цель использования данного процесса:

- Сформулировать требования к поведению конкретного модуля
- Повысить доверие к исходному коду
- Обеспечить качество путём быстрого обнаружения ошибок
- Поддержать хороший дизайн системы
- Дать возможность непрерывно интегрироваться

Почему ветка важна?

Для менеджера:

- Увеличение скорости разработки в среднесрочной перспективе
- Снижение стоимости разработки в среднесрочной перспективе

Для разработчика:

- Доверие своему и чужому коду
- Возможность правильно спроектировать систему

Что будет, если её не делать?

- Увеличится цикл обратной связи - дефекты находим позже, например, только на ручном регрессе, что затягивает проект в целом
- Вырастет количество дефектов из-за невозможности постоянно запускать тесты после изменения исходного кода
- Команда разработки потеряет контроль над кодовой базой - проще сделать новый функционал "рядом", чем менять существующий
- Система быстро устареет и потребуется переписать её с нуля

На кого может быть делегирована?

- Тимлид уровнем ниже
- Разработчик

Примеры поведения

Примеры плохого поведения

- Не писать unit-тесты в целом - проверка качества ПО смещается в сторону более дорогих способов, например, интеграционного тестирования
- Писать unit-тесты, не опираясь на "хорошие практики" - снижается их эффективность, что часто ведёт к полному отказу от данной практики в команде/проекте
- Не запускать unit-тесты, в том числе автоматически - практически то же самое, что их не писать или не актуализировать
- Пропускать при запуске или отключать неуспешные unit-тесты - ведёт к снижению доверия к ним

Примеры хорошего поведения

- Писать unit-тесты перед реализацией - подход TDD
- Покрывать тестами ранее непокрытый код - снижение рисков при его модификации и рефакторинге
- Использовать Заглушки и Тестовые дублёры - изоляция тестируемого кода
- Использовать DSL Domain Specific Language - повышение читаемости тестового кода
- Актуализировать/дополнять unit-тесты при исправлении новых дефектов, что позволяет учесть ранее пропущенное требование и предотвращает появление дефекта вновь

Способы прокачки

Практика

Чтобы отработать навыки написания unit-тестов, можно использовать KataCatalogue - сборник простых примеров, на которых можно потренироваться

Консультации

- Telegram-чат TL Bootcamp.

Теория

Тест считается по-настоящему модульным, если обладает следующими свойствами:

- Сфокусированный - проверяет только одно утверждение
- Ценный - отражает актуальное требование
- Независимый - от других тестов или окружения, на котором выполняется
- Быстрый - выполняется достаточно быстро, чтобы запускаться при каждом изменении кода
- Понятный - соблюдается структура, конвенция именования, имеет маленький размер
- Поддерживаемый - меняется с течением времени

Хорошей структурой unit-теста является подход AAA: **Arrange**(условие), **Act** (действие), **Assert** (утверждение):

java

```
class CalculatorShould
{
  @Test
    public void returnSumOfTwoIntegerNumbers()
  {
    // Arrange
    Calulator calculator = new Calculator();

    // Act
    Inreger result = calculator.sum(1,1);

    // Assert
    assertEquals(2, result);
  }
}
```

Статьи

- Юнит-тестирование для чайников
- Software Testing Anti-patterns

Книги

- xUnit Test Patterns: Refactoring Test Code, Gerard Meszaros, 2007
- The Art of Unit Testing: with examples in C#, Roy Osherove, 2013