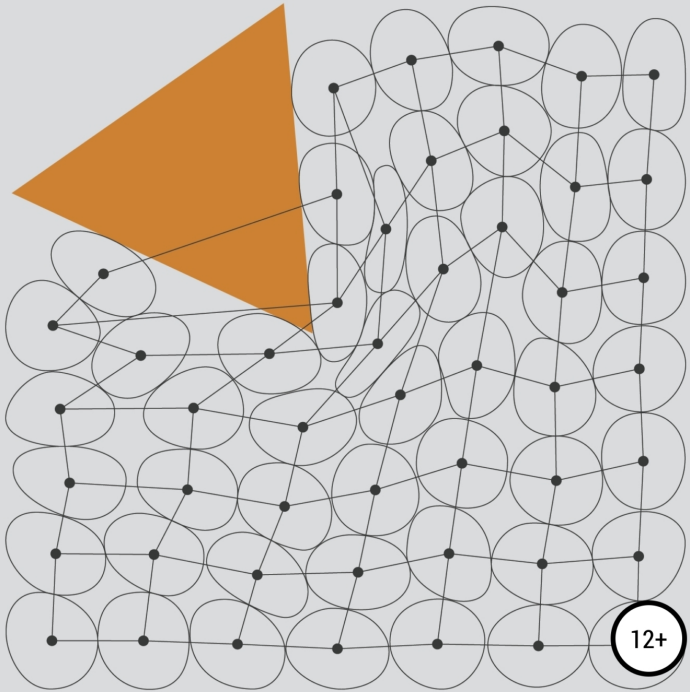
Александр Бындю

Антихрупкость в IT

Как достигать результатов в IT-проектах
в условиях неопределённости



Александр Васильевич Бындю

Антихрупкость в IT

http://www.litres.ru/pages/biblio_book/?art=68036792
SelfPub; 2024

Аннотация

Код пишется и макеты рисуются для того, чтобы компания быстрее и точнее конкурентов понимала и выполняла потребности своих клиентов. Для достижения этого результата следует понимать, какие инструменты работают, а какие мало применимы в мире постоянных перемен. В своей книге я рассказываю, как можно выстроить внутреннее качество IT-систем и процессы разработки таким образом, чтобы успевать вовремя подстроиться под любые изменения внутренней и внешней среды, а также изменяющиеся по ходу реализации проекта нужды заказчика. Одним из ключевых понятий данного исследования является понятие «антихрупкость». Мой собственный предпринимательский опыт и опыт моих партнёров, клиентов и друзей убедил меня в том, что при создании IT-продуктов важное внимание следует уделять их «антихрупкости» – прочности, работоспособности предлагаемых клиенту решений в условиях меняющегося мира.

Содержание

Александр Бындю
Антихрупкость в IT

Отзывы о книге

Александр – практик, создавший успешную ИТ-компанию, интересную своей культурой и решениями. Эта книга – результат жизненного опыта, пройденные грабли и ошибки. Позволяет с разных сторон посмотреть на задачи, решаемые с помощью ИТ. Послужит отличной точкой входа для понимания дальнейших направлений профессионального развития

Заостровцев Николай – коуч-консультант организаций и первых лиц

Книга является настольным руководством для заказчика, который решил воспользоваться услугами заказной разработки ПО, или для директора, который решил завести в своей компании айтишников. Книга вводит вас в необходимую терминологию и инструменты, необходимые для достижения успеха в IT-проектах.

Книга изобилует примерами из личного опыта автора, который, находясь по другую сторону контракта, объясняет

вам, как правильно стоит пользоваться услугами разработки. Десятки примеров из практики, засады и ловушки, а также немного юмора делают процесс чтения захватывающим и полезным.

Алексей Пименов – тренер и консультант по современным методам менеджмента, сооснователь компании Neogenda

В моих руках лежит Книга – плод долгих изысканий Александра Бындю в области практического использования IT-решений как в области организации собственного бизнеса, так и при построении подобных решений по заказам его клиентов. Автор показал отличное владение предметом, широкие знания зарубежных авторов и подходов. Мне, как финансисту, преподавателю и в прошлом проджект-менеджеру Мирового банка, особенно понравились следующие вещи. Умение донести сложные понятия и процессы простым языком, используя картинки, блок-схемы, аналогии и т. д. Также мне импонирует глубокое знание работ американского финансиста и трейдера ливанского происхождения Нассима Талеба, числе которых «Одураченные случайностью», «Антихрупкость» и др. Читателю, безусловно, понравится стиль изложения автора – с юмором, с примерами из практики. Чего только стоит раздел «Я бухгалтер, я так вижу» – сколько из нас сталкивались с этой буквально обезоружива-

ющей фразой! Автор явно знаком с книгой известного американского психолога Эрика Берна «Люди, которые играют в игры. Игры, в которые играют люди», – примеров из этого эпохального труда в данной книге немало.

Особый интерес у меня вызвало сравнение технических долгов с финансовыми, подтверждаю их сходство, исходя из собственного опыта финансового директора.

Уверен, что данная книга и последующие на её базе курсы послужат делу цифровой трансформации общества, о которой так много говорится с высоких трибун, но пока так мало делается. Книга А. Бындю была бы полезна в сфере образования государственных служащих в указанной сфере.

Трегубов Владимир Андреевич – доцент РАНХиГС при Президенте РФ, к.э.н., квалифицированный инвестор, финансовый консультант, журналист.

Я рекомендую эту книгу по трем причинам. Первая причина. Автор книги, Александр Бындю – практик, книга написана на основе многолетнего успешного опыта ведения бизнеса. Вторая причина. Александр самостоятельно и независимо мыслит. Смотрит на устоявшиеся в отрасли каноны свежим взглядом, не боится перерабатывать в котлеты "священных коров". Третья причина – книга читается легко, у автора хороший слог.

Павел Буков – консультант по управлению стрессом в ор-

ганизациях, спикер TEDx, врач психотерапевт.

Предисловие

Однажды я рассказывал топ-менеджеру крупной российской компании про микросервисы. Руководителям, подумал я, было бы полезно разбираться в том, что именно в IT работает на благо бизнеса, а что им пытаются продать как «серебряную пулю», которая не сработает. Я хотел рассказать об этом без лишних деталей и технических глубин, но при этом донести необходимые для понимания основы: как устроено создание IT-продуктов, в чем состоят плюсы и минусы разных подходов и каковы их границы применимости. Так и появилась эта книга.

В моём понимании IT занимается обслуживанием бизнеса. Код пишется и макеты рисуются для того, чтобы компания быстрее и точнее конкурентов попадала в потребности своих клиентов. Для этого следует понимать, какие инструменты работают, а какие мало применимы в мире постоянных перемен.

Название этой книги отсылает к близкой мне работе Нассима Талеба «Антихрупкость». Я часто буду использовать термины из его книги, поэтому надеюсь, что вы прочитали его книгу или хотя бы ознакомились с её основными тезисами.

Меня вдохновляют идеи Талеба: как в условиях неопределённости извлекать выгоду из хаоса и быть готовым к случайным внешним событиям. Главными инструментами «Антихрупкости» указаны *прилаживание*, основанное на рациональности, и метод проб и ошибок. В своей книге я рассказываю, как можно выстроить внутреннее качество IT-систем и процессы разработки таким образом, чтобы успевать *вовремя* подстроиться под любые изменения.

То, что я описываю, обычно обсуждается на узкоспециализированных профессиональных конференциях. Мои знания живут на уровне техлидов, техдиректоров, арт-директоров и тимлидов и почти не поднимаются выше: до инвесторов, стейкхолдеров, руководителей и владельцев бизнеса. С другой стороны, не опускаются ниже – до разработчиков, QA и дизайнеров. Между тем, в процессах цифровой трансформации и создания сложных IT-продуктов важно, чтобы все члены команды понимали основные концепции продукта и видели его движение.

Я хочу, чтобы эту книгу прочитали менеджеры и будущие ведущие разработчики, а также студенты, интересующиеся разработкой сложных продуктов.

Кому нужна эта книга

Вам – если вы хотите обогнать конкурентов за счёт развития IT внутри своего бизнеса. Владельцы компаний, топ-менеджеры, руководители подразделений и проектов узнают из книги, как крутятся шестерёнки IT-мира и как эффективно выстраивать развитие IT-продуктов.

Тимлидам и техлидам книга даст правильное представление о пользе IT в достижении бизнес-целей.

Особенно полезна книга будет тем, кто занимается цифровой трансформацией и вовлечён в переход от проектного подхода к продуктовому, а также всем IT-специалистам и инженерам, готовым взглянуть на свою работу под новым углом.

Об авторе

Александр Бындю
IT-архитектор, Agile и Lean эксперт

Я всю жизнь занимаюсь IT. Самостоятельно прошёл путь

от программиста до владельца IT-компании. Обожаю инженерное дело, создание IT-продуктов, разработку IT-архитектуры и участие в трансформации бизнеса.

1. С 2012 года владелец и IT-архитектор в компании Byndyusoft, выпустившей много критически важного софта для электронной коммерции, ритейла, логистики и других областей.

2. С 2009 года преподаю в университетах принципы разработки ПО и управлению IT-проектами.

3. Выступаю на профессиональных IT-конференциях, рассказываю об управлении IT-продуктами и современных подходах к IT-архитектуре.

Как я помогаю компаниям:

1. Трансформирую культуру крупных компаний, организую цифровые трансформации.

2. Перевожу монолитные системы на микросервисную архитектуру.

3. Провожу тренинги для программистов и менеджеров.

4. Организую работу от сбора требований до запуска IT-продукта.

5. Консультирую как внешний IT-архитектор и личностный коуч.

Личный сайт https://byndyu.ru

Благодарности

Коллегам за поддержку и такую же сильную любовь к инженерному делу, как у меня. В особенности Андрею Шапиро – за его глубину и постоянное желание докопаться до сути.

Моей жене Яне Мильбергер за её любовь и мудрость.

Хочу поблагодарить за обратную связь по книге, подсказки, уточнения и помощь Антона Степанова, Алексея Пименова, Михаила Пуляевского, Николая Заостровцева и Влада Зелинского.

Структура книги

В разделе I – инструменты для управления IT-продуктами в условиях неопределённости. В разделе II – IT-архитектура и работа с IT-системами с точки зрения эффективности достижения бизнес-целей.

В конце книги есть два приложения с дополнительными материалами к каждому разделу. В приложениях я собрал ссылки на свои статьи, интервью и выступления, чтобы вы могли глубже погрузиться в затрагиваемые в книге темы.

Необязательно читать последовательно. Для начала можно выбрать наиболее актуальную для вас тему и возвращаться к книге по мере возникновения новых вопросов. Каждая тема – это ключ к большой области знаний. Я даю основы понимания и ви́дения этой темы плюс множество ссылок для самостоятельного изучения и дальнейшего движения в правильном направлении.

В сносках довольно много ссылок. Если вы читаете книгу в бумажном виде, то у меня для вас хорошая новость: не нужно набирать длинные ссылки вручную. Перейти по ссылке в сноске можно, написав в строке браузера https://byndyu.ru/footnote/<номер сноски>, и вы перейдёте к статье

или видео, на которые я ссылаюсь. Кроме этого, полный список ссылок можно найти на сайте книги https://byndyu.ru/AntifragileIT.

Раздел I. Управление IT-продуктами

В этом разделе собраны подходы и принципы, которые помогут выстроить работу компании для создания IT-продуктов и пригодятся в цифровой трансформации.

Глава 1. Кнопочное мышление против целостного IT-продукта

Кнопочные решения портят жизнь разработчиков, пользователей и инвесторов

Когда мы общаемся с коллегами, заказчиками и пользователями, я использую фразу «кнопочное мышление». Что я имею в виду?

Синонимами кнопочного мышления можно назвать экранное мышление или преждевременную концептуализацию. Суть мышления кнопками можно раскрыть на десятке примеров из практики, но пока – история, которая наверняка случалась с каждым. Представьте: к вам приходит клиент и рассказывает о падении конверсии на сайте. А вы в ответ: «Давайте кнопку покупки сделаем побольше и поярче!» Что произошло? В бизнесе возникла проблема. Вместо погружения в детали, вместо исследования причин, вы играете с размерами кнопки. Это и есть кнопочное мышление.

Откуда берутся кнопочные идеи

Практика показала, что есть три типажа людей-генераторов кнопочных решений. Речь не о конкретных позициях в проекте, потому что лажать может каждый член команды.

Программисты, QA, дизайнеры, заказчики, инвесторы и конечные пользователи – потенциальные создатели кнопок.

Торопыги

Представьте ситуацию, в которой Заказчик приносит проблему. Он делится этой проблемой с Торопыгой. Что делает Торопыга в ответ? Чувствует давление, как будто Заказчик ждёт ответа, причём мгновенно. Пауза затягивается, а ответа в голове не появилось – напряжение возрастает. В своём воображении Торопыга уже видит, как его, беднягу, обвиняют в некомпетентности, и выдаёт первое пришедшее в голову решение. В лучшем случае бесполезное, в худшем – вредное.

Как быть, если вы заметили за собой недуг торопыжничества? Во-первых, осознайте, что невозможно знать все ответы. Придумывание налету – не признак профессионализма. Во-вторых, брать паузы в переговорах и на совещаниях – норма. Берёте паузу, идёте думать. Приносите с собой варианты решений, риски по каждому, плюсы и минусы[1].

Решалы

Решалы – ребята-профи, которые в IT собаку съели. Спроси – сразу ответят. Хотя и не факт, что за плечами у них серьёзные проекты и десятки лет опыта.

---

[1] Александр Бындю. Проблем быть не должно, https://byndyu.ru/footnote/1

Для Решалы входящие проблемы и задачи видятся типовыми, уже решёнными. По фреймворку Cynefin Framework[2] Решалы видят мир в квадратике Obvious, ну, максимум – в Complicated. Другими словами, у них всегда есть готовое решение, надо только выбрать категорию проблемы. Не понимая того, что они лишают себя шанса преподнести бизнесу проработанное решение и вырасти на новой задаче.

Бизнес-решала пострашнее Разработчика-решалы, потому что любит проталкивать Pet Features в продукт. Pet feature – это такие любимчики, как милые домашние питомцы, цель их существования в продукте неясна, но они почему-то по душе тому, кто платит за разработку.

Спасители

Неожиданно в голове у человека появляется чувство, что если он не поднимется с колен и не поведёт за собой команду, то конец проекту, продукту и даже компании. Он берёт на себя роль Спасителя, поднимает флаг и ведёт людей за собой. Спасители мыслят кнопочно с особым фанатизмом.

Если вы решили, что это ситуационное лидерство[3], о котором так много сказано в гибких подходах разработки, то будете правы. Только не всё ситуационное лидерство одинаково полезно. Проблема возникает, когда во время подъ-

---

[2] Cynefin framework, https://byndyu.ru/footnote/2

[3] Situational leadership theory, https://byndyu.ru/footnote/3

ёма человек перестаёт слышать других и адекватно оценивать ситуацию. Его решения становятся ультимативными: он на войне, он Спаситель, он вершит судьбы.

Если вы заметили Спасителя в проекте, постарайтесь вывести человека из этого состояния. Медленно, но непреклонно и жёстко. Чем раньше, тем лучше. Впереди его ждут сожаления о решениях, принятых в состоянии аффекта.

Я – могучий генератор кнопок

Не знаю, к какой части себя присоединить, но и я тоже великий генератор кнопок и быстрых решений. 15 лет опыта и скорость мозга не дают мне усомниться в своей правоте. Я решаю со страшной скоростью и наслаждением.

Наверняка в мире живут айтишники со стальной волей. Они способны держать себя в жёстких рамках, не вываливаться в роль Решалы или Спасителя. Я к таким не отношусь и периодически скатываюсь в одну из ролей, а иногда в их комбинацию.

Когда я это осознаю, бью себя по щекам, колю булавкой и торможу подобное решательство. Не всегда успешно, но я всё равно стараюсь. Кажется, со временем становится лучше.

Мимикрирующие User Story и хитрые Product Owner'ы

С источниками кнопочного мышления разобрались. Теперь узнаем, что с ними делать. Почему мы даём Решалам порулить? Почему не отбрасываем поверхностные идеи? Как предотвратить собственное решательство?

User Story как фильтр

Когда я думал о фильтрах, не пропускающих кнопочные идеи, вспомнились User Story[4]. Мы используем истории для формирования задач проекта в повседневной практике. Сила User Story в том, что они заставляют описывать ценность. Нет ценности в истории – нет лишней кнопки в интерфейсе.

Работа строится следующим образом. Вносишь в работу идею → опиши в виде User Story. Ответь на вопрос «Чтобы что?».

Хочешь кнопку выгрузки отчёта в Excel. Ок, чтобы что? Чтобы было на всякий случай? В мусорку, не берём в работу.

Бухгалтер Оля сказала, что ей так нравится? В мусорку, не берём в работу.

Вы посовещались внутри отдела экономистов, нарисовали кнопку в интерфейс и теперь хотите, чтобы мы её добавили? Чтобы что? Потому что это ваша идея и она вам нравится? В мусорку.

Вы заказчик и вы так видите? Другого «чтобы что» нет?

---

[4] Андрей Шапиро. Руководство по сбору требований в формате User Story Mapping, https://byndyu.ru/footnote/4

Увы, в работу не берём. Иногда можно сделать ставку на Pet Feature, но перед этим следует обозначить риски и чётко проговорить бесполезность затеи.

Хитрые Product Owner'ы

User Story отлично отсеивает кнопочные идеи. Проверено на практике. Однако здесь появляется оборотная сторона проблемы. Product Owner'ы и stakeholder'ы поняли, что через User Story не пройти, потому что приходится искать ответ на вопрос «Чтобы что?». А это сложно, если ты пришёл с Pet Feature. Сложно, но сильно хочется.

Product Owner'ы[5] подстроились под новую модель постановки задач. Они научились играть в эту игру.

Я, как корпоративный клиент,

Хочу скачивать отчёт о движениях денежных средств,

Чтобы видеть, что баланс стал отрицательным.

Неопытный разработчик или дизайнер примут такую историю за правильную. Но присмотритесь к ней внимательно. Перечитайте эту историю и попробуйте прикинуть, какие вопросы у вас возникнут к Product Owner'у.

Я бы для начала спросил о дальнейшей жизни скачанного отчёта. О жизни пользователя, который скачает этот отчёт. Что он найдёт в отчёте? С помощью чего найдёт нужное? Как

[5] Product Owner по сути – это предприниматель внутри компании. Он ещё не вырос до создания своей компании либо не хочет рисковать созданием собственного бизнеса, при этом он уже мыслит и действует как предприниматель.

он отделит нужное от ненужного?

Мимикрирующие User Story

Правило User Story соблюдено, но без погружения и дополнительных вопросов в работу оно не работает. Что делать? Копаем, ищем корни проблемы, задаём открытые вопросы, используем принцип 5 Why[6]. Со временем узнаём корень проблемы и записываем в User Story:

Я, как корпоративный клиент,

Не понимаю, в каком состоянии счёт, и из-за этого ухожу в минус.

Хочу…

Чтобы…

Уже лучше, потому что мы поняли, откуда пришло кнопочное решение со скачиванием отчёта. Теперь мы знаем, что клиент теряет деньги, если оперативно не получает информацию о счёте.

Следующий шаг – понять, как мы поменяем жизнь корпоративного клиента. Gojko Adzic в книге Quick Ideas To Improve Your User Stories[7] указывает на то, что лучше прописывать в User Story дельту между тем, как пользователь жил до релиза, и тем, как он заживёт после. Получаем такую

---

[6] Five whys, https://byndyu.ru/footnote/6

[7] Gojko Adzic. Fifty Quick Ideas To Improve Your User Stories, https://byndyu.ru/footnote/7

историю:

Я, как корпоративный клиент,

Не понимаю, в каком состоянии счёт, и из-за этого ухожу в минус.

Хочу **останавливать работу, если баланс стал критично низким,**

Чтобы не терять деньги.

Мы остановим работу пользователя и трату денег, когда баланс станет отрицательным. Когда мы озвучиваем предложение пользователям, они не верят, что можно автоматически остановить работу. Для пользователя боль потери денег настолько сильна, что они сами придумали скачивание отчёта, они готовы смотреть в отчёт, искать в нём ответ на вопрос об отрицательном балансе.

В последней версии User Story кнопочное решение убрано. Раскопана корневая проблема. Предложено решение, которое закроет корневую проблему. Появился шанс принести пользу после релиза, а не добавить ещё одну кнопку для скачивания ещё одного отчёта.

Преждевременные решения

Некоторым людям неймётся выпалить решение. Они как будто играют в «Свою Игру» или «Брейнринг». Ждут вопрос и на скорость предлагают решение.

В спешке между проблемой и идеей возникает слепая зо-

на. Цепочка рассуждений и выводов остаётся за кадром (рис. 1).

Рис. 1. Отсутствие связи между целью и решением

Мы не принимаем первое поспешное решение, копаем корень проблемы, определяем действующих лиц. После прокладывания пути из цели через действующих лиц до корня проблемы кнопочное решение теряет былую прочность (рис. 2).

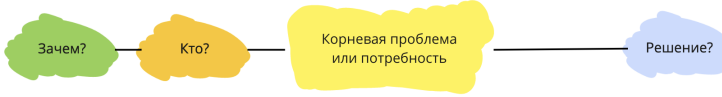


Рис. 2. Прорисовка связи между целью и решением

Увидев корневую проблему или потребность, накидываем много возможных решений (рис. 3).

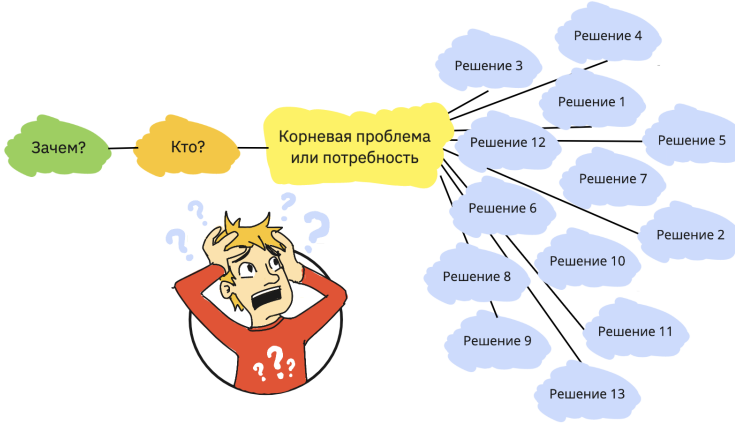


Рис. 3. Множество решений для одной цели

Обратите внимание, что теперь налицо выбор, но сделать его сложнее. У меня есть предположение, что люди намеренно останавливаются на первом решении, которое кажется подходящим. Ведь если идти дальше, то придётся выбирать, оценивать риски каждого решения, его плюсы и минусы. Работы прибавится. Кроме того, чем шире выбор, тем меньше будет радость от итогового решения.

Глубокое бурение проблемы затратно, никто не стремится

в это болото. Но если мы хотим создать полезное решение, то нужно пересилить себя и раскрыть слепую зону.

Истории из жизни

Кейс: Сужение видения

Идёт планирование релиза. Product Owner заканчивает фразу словами: «…можно отправить почтой». Я сразу останавливаю обсуждение, потому что одной фразой произошло сужение проблематики до одного решения. Остановились и раскопали корень потребности. Почему отправлять? Почему почтой? В мире ведь придумано много способов донесения информации до пользователей. В итоге предложили пять способов рассказать клиентам об обновлениях. Совет: не сводите решение к одному варианту.

Кейс: Решения без проблемы

Новый заказчик обсуждает с нами модернизацию IT-продукта. Пока рассказывает о продукте, вспоминает о проблеме – клиенты уходят в минус и перерасходуют ресурсы без оплаты. Сервис берёт деньги по мере выполнения операции, но предсказать расходы заранее нельзя. По мере рассказа заказчика посещает идея: обрубать доступ и оставлять клиента без результата.

Обсуждение прервали, раскопали проблему пользователей. Выяснилось, что пользователи не понимают, сколько де-

нег остаётся в каждый момент времени, поэтому не могут оперативно принимать решения. Мы предложили показывать им расходы и текущий баланс в режиме онлайн. Заказчик удивился и сказал: «А что, так можно?»

Сколько стоят 1000 строк кода?

Представьте сцену в магазине. Вы набрали продуктов в корзину и подошли к кассе. Кассир пробил товары, взвесил фрукты и овощи, назвал стоимость. Отлаженный механизм.

Та же история с созданием IT-продукта. Вы сидите на кассе, приходит бизнес с корзиной фич и решений. Вы оцениваете, взвешиваете, говорите ему стоимость.

Давайте вернёмся в магазин и переиграем ситуацию. Вы подходите к кассе, выкладываете покупки. Продавец вам говорит: «Зря вы взяли помидоры „шеди леди“ для рагу из кролика. Этот сорт слишком сладкий, для рагу не подойдёт. Возьмите сорт „маленький принц“, рагу с ними отменное».

Посмотрим, что изменилось. Почему вы благодарны кассиру и хотите его обнять? Он узнал вашу цель. После этого он предложил решение, которое двигает вас к цели, а не просто молча согласился с предложенным решением. Ценность покупки в этом магазине выросла, удовлетворение выросло,

вы захотите приходить в этот магазин снова.

Теперь вернёмся к IT. Для выявления целей, понимания пути достижения целей, формирования выбора из решений я рекомендую Impact Mapping[8]:

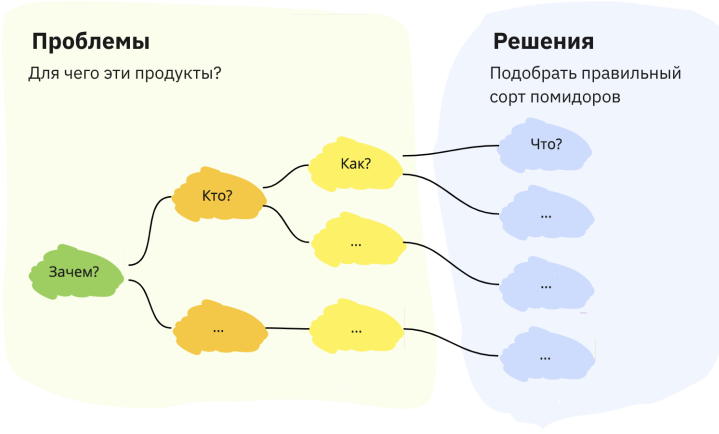


Рис. 4. Разделение проблем и решений

Техника изучается за пару дней. С помощью Impact Mapping мы прокладываем путь от решений до цели, расставляем приоритеты, отсекаем Pet Feature, которые идут со стороны бизнеса и со стороны команды. Единственная сложность – процесс создания Impact Mapping: он требует навы-

[8] Impact Mapping, https://byndyu.ru/footnote/8

ков эффективной коммуникации.

Истории из жизни

Кейс: Требуется больше всплывающих окон

Представьте IT-отдел внутри компании. Руководители отдела маркетинга, финансов и прочих ставят ему задачи. Приходит начальник, который отвечает за точки продаж, и требует добавить всплывающее сообщение раз в 10 минут на рабочем месте. Работников на местах обяжут нажимать «ОК» в модальном окне каждые 10 минут, чтобы понять, на месте работник или нет. Задача как задача; IT-отдел взял да и сделал. Прошло время. Работники на местах ужились с новшеством.

В IT-отдел пришёл начальник маркетинга и попросил добавить всплывающее сообщение, чтобы работник выходил на улицу и раздавал листовки. Сообщение по задумке всплывает каждые 30 минут, в результате должны повыситься продажи. Задача как задача; IT-отдел взял да и сделал.

На местах это вылилось в противоречивый сценарий. Работник видит сообщение о том, что надо идти на улицу и раздавать листовки. Он выходит и раздаёт, а в это время всплывает сообщение: «Ты на месте?»

Почему так произошло? Никто не контролировал целостность системы. Многоголовый Product Owner приносил задачу, разработчики брали задачу, не создав целостной кар-

тинки поставки ценности, поэтому они не увидели противоречий.

Кейс: Зачем делаем?

Заказчик пришёл с идеей сделать приложение для курьеров. Заказчик – федеральная компания, сотни филиалов по стране. Цель продукта – оптимизация работы курьеров.

До работы с нами у проекта накопилась полугодовая история. Заказчику сделали ТЗ, реализовали часть мобильного клиента, но не сделали серверную часть. С этой историей заказчик пришёл к нам. Мы начали проект по нашему процессу[9], и уже к концу Customer Journey Mapping заказчика осенило. Они поменяли бизнес-модель, запустили ряд экспериментов в бизнесе и обещали вернуться к нам через три месяца. В итоге вернулись через полгода с перестроенной компанией, которая стала готова для создания нового продукта. Продукт мы сделали и успешно запустили. Мы считаем это успехом, так как не позволили потратить время на что-то бесполезное.

Движение без цели

Если цели IT-отдела или IT-продукта не сформулированы, то это благодатная почва для кнопочных решений. Перефразируя фразу из монолога Жванецкого[10]: Если нет цели,

---

[9] Byndyusoft. Анализ IT-продукта, https://byndyu.ru/footnote/9

[10] Отрывок из выступления Жванецкого, https://byndyu.ru/footnote/10

то куда бы ты ни шёл – получается вперёд.

Когда мы берём задачу, то сопоставляем её стоимость с отдачей в достижении цели. А если цели нет? Значит, и соизмерять не с чем. Отсюда рождается стиль работы, при котором задачи реализуются, потому что прикольно эти задачи реализовывать. В таких отделах разработки кипит жизнь, фичи добавляются, идут непрерывные релизы. При этом бурном движении результат не просматривается.

Истории из жизни

Кейс: Покажем, потому что можем

Продукт – SaaS-инструмент для партнёров топовой e-commerce России. Диалог с IT-подразделением заказчика:

– Давайте выведем договоры в интерфейсе, – говорит разработчик со стороны заказчика.

– Чтобы что? – отвечаем мы.

– Они уже лежат в БД, можно легко вывести.

– Как это поможет достигнуть целей продукта?

– Без договоров невозможно заплатить!

– Чтобы заплатить, нужно начать пользоваться продуктом, а для пользователя в этот момент взаимодействия с системой никакого продукта еще нет.

В таких случаях помогает только возврат к целям проекта и фильтрация с помощью Impact Mapping (раздел l, глава 2).

Рефлексия и душевный покой

Чтобы уберечь ваши нервы, делюсь выработанной стратегией борьбы с кнопочным мышлением:

1. Когда к вам пришли с кнопочной идеей, спросите себя, почему они принесли такое решение, почему оно не нравится вам, какие вопросы выявят корень проблемы. Только после этого начинайте говорить.

2. Поймите, что коллеги не со зла лезут с кнопочными идеями. Никто не хочет навредить или саботировать. Все пытаются принести пользу.

3. Управляйте на уровне достижения бизнес-целей, а не задач.

4. Ставьте перед командой проблемы, а не приходите с решениями.

5. Делайте короткие итерации (одна неделя или короче), постоянно собирайте обратную связь от команды и от клиентов.

6. Проводите валидацию идей как можно раньше, убивайте идеи до этапа реализации.

Рекомендации одинаково банальные и действенные. Мне в работе помогает.

Здесь самое важное, что мы фильтруем «хрупкие» идеи, которые со временем разрушат целостность нашей системы. Мы требуем фильтровать входящие задачи вопросом «чтобы

что?», а это создаёт в системе запас прочности и позволяет ей оставаться гибкой по отношению к новым изменениям.

Замечайте кнопочное мышление за собой, замечайте за коллегами, рассказывайте бизнесу и учитесь работать с запросами пользователей. Помогайте друг другу в преодолении недуга.

Глава 2. Impact Mapping на практике

Трассировка от задач до целей

Когда читал книгу Impact Mapping[11] первый раз, было желание бросить её на середине, настолько в ней всё очевидно. Я нашёл в себе силы и дочитал, благо книга короткая и с большими картинками. Как впоследствии оказалось, вся соль была в применении советов на практике. Я не применял. В моей практике заказчик иногда писал бизнес-цели в официальных документах к проекту; иногда мне казалось, что я и так понимаю цели бизнеса – они абсолютно очевидны. К чему уточнять очевидное? Разницу я почувствовал, когда начал применять Impact Mapping в работе.

Важно! На данный момент я уже заменил этот метод на Карту гипотез – метод стратегического планирования. Если вы хотите узнать более углубленный вариант работы со стратегией, то можно пропустить главы об Impact Mapping и прочитать книгу о Карте гипотез.

История появления Impact Mapping

Раньше на старте проекта у нас были технические задания, схемы работы системы и в лучшем случае прототипы

[11] Gojko Adzic. Impact Mapping, https://byndyu.ru/footnote/11

интерфейса. В этих документах не хватало понимания динамики развития проекта и приоритетов в работе.

Мы начали писать User Story[12] и делать User Story Mapping. Эти практики добавили понимание логики развития проекта и наших приоритетов, дали возможность плодотворнее общаться с заказчиком. Чего не хватало? Продукты существуют не в вакууме: нужно уметь видеть более глобальные задачи, которые лежат где-то выше историй использования системы. Не хватало простой игровой практики по постановке целей проекта, из которых потом будут появляться User Story Map и список User Story на релиз.

Mijo Balic и Ingrid Ottersten в 2007 году написали статью «Effect Managing IT» (подробнее Agile product management using Effect Maps[13]). Через четыре года Gojko Adzic[14] выпустил книгу «Specification by Example»[15], где в главе «Deriving scope from goals» упоминает о технике под названием Effect mapping. Эта техника призвана помогать командам фокусироваться на бизнес-целях, выявлять заинтересованные стороны и их потребности.

Gojko Adzic со временем добавляет в Effect mapping несколько усовершенствований, таких как: приоритизация

---

[12] User Story, https://byndyu.ru/footnote/12

[13] Gojko Adzic. Agile product management using Effect Maps, https://byndyu.ru/footnote/13

[14] Gojko Adzic, https://byndyu.ru/footnote/14

[15] Gojko Adzic. Specification by Example, https://byndyu.ru/footnote/15

целей и воздействий, возможность уходить от технических деталей на уровне What, цикличность в предположениях и экспериментах. На мой взгляд, это действительно важные изменения, они помогают в реальной жизни. После этого техника стала называться по-новому – Impact Mapping.

Руки без головы

Представьте, что вы владелец магазина. К вам приходит заказчик. У него в голове уже есть набор фич на покупку, он знает чего хочет. Он берёт корзину для покупок, складывает в неё список технологий, десяток прототипов интерфейса, интеграцию с соцсетями и т. п. Подходит к кассе и просит всё взвесить, реализовать и выставить ему счёт.

Получается, что заказчик пришёл к вам с готовыми решениями каких-то своих проблем. В такой ситуации заказчик купит только руки разработчика, но не голову. Разработчик не сможет критически оценить предложенные решения. Будет ли успешным проект с подходом, где купили только «руки»? Шанс невысок.

Зато в наших силах увеличить шансы на успех за счёт того, что каждый в команде будет понимать и разделять цели бизнеса. Тогда любое решение – от именования переменной в коде до выбора архитектуры – будет приниматься с учётом реальных потребностей бизнеса.

Остаётся вопрос, как вытащить из бизнеса истинные це-

ли, которых мы хотим достичь? Как сделать так, чтобы команда услышала их, приняла и начала с ними работать? Как провести трассировку от любой задачи до бизнес-цели, чтобы была видна логика выбранного решения?

Составляем Impact Map

Impact Map (Карта воздействий) – это mind map по целям проекта с картой влияний, которые должны подтолкнуть бизнес к достижению целей (рис. 5).

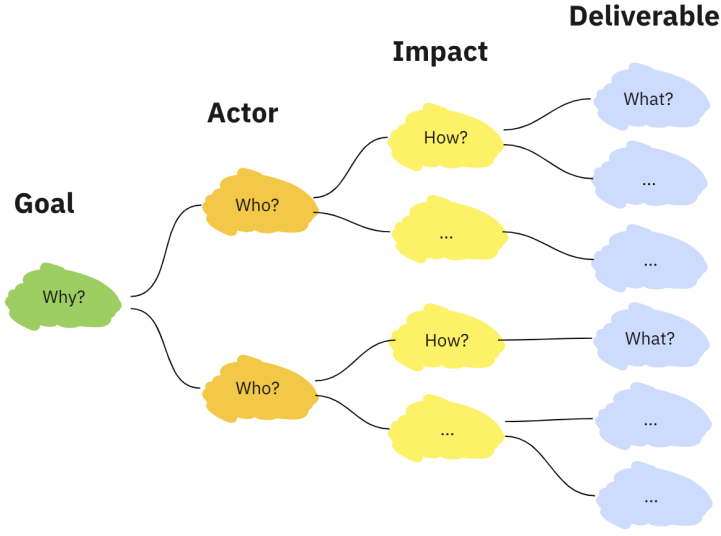


Рис. 5. Схема Impact Map

Why?

Центральный элемент нашей карты, который отвечает на ключевой вопрос: зачем мы это делаем? Это цель, которую бизнес пытается достичь.

Who?

На первом уровне мы отвечаем на вопросы: *Кто может помочь достичь желаемого результата? Кто может помешать? Кто пользователи нашего продукта?* Сюда войдут все заинтересованные стороны, способные повлиять на цели бизнеса.

How?

На втором уровне нужно описать действия заинтересованных сторон для достижения целей. Мы ищем ответ на вопросы: *Как они помогут бизнесу достичь целей? Как они могут помешать успеху продукта?*

What?

После ответа на основные вопросы можно обсудить кон-

кретные задачи. Третий уровень отвечает на вопросы: *Что мы можем сделать как организация или команда разработки, чтобы стимулировать действия?* Здесь будет описано что-то вроде технического задания.

Организация процесса

Приглашайте не больше 10–15 человек на это мероприятие, иначе будет сложно проводить. Оптимально позвать по три-четыре человека со стороны бизнеса и со стороны команды продукта.

Со стороны бизнеса обязательно присутствие тех, кто принимает решения, а не только технических специалистов со сложившимся мнением насчёт конкретных реализаций поставленных целей.

Подготовьте карту и доску (или стену) заранее. Impact Mapping для решения задачи с объемом работы в полгода месяцев умещается на доске стандартного размера.

Составление карты займёт от одного часа до двух дней. Сроки сильно зависят от состава участников и ваших навыков проведения.

Каждый блок карты можно рисовать маркером или делать

стикерами. Я предпочитаю стикеры, они более мобильны, а это важно, потому что Impact Map будет часто меняться по ходу погружения в продукт.

Перед началом обязательно проговорите правила и цели составления карты. Если есть время, то разошлите всем материал по теме.

Если есть возможность и обстоятельства позволяют, то сделайте несколько упражнений на знакомство друг с другом, потому что техника подразумевает открытое общение.

И самое главное – процесс должен проходить легко и весело. Не добавляйте в него бюрократии!

Пример из практики

Разберём пример, очень приближенный к реальному проекту, для которого мы сделали Impact Mapping. Остановимся на ключевых моментах при составлении Impact Mapping и фатальных ошибках.

Why?

Корневым элементом нашей карты будет список бизнес-целей. Например, это может быть *увеличение удовлетво-*

рённости пользователей в два раза. Важно, что *удовлетворённость пользователей* – это индекс, то есть конкретная цифра, которую можно взять из CRM, а не мнение/ощущение заказчика. Мы же хотим после поставки фич *измерить* достижение цели и понять, в том направлении мы идём или нет. Если бы *удовлетворённость пользователей* была не цифрой, то как бы мы узнали, что достигли цели? Важно, что мы написали именно *в два раза*, а не просто *увеличение*. Хорошие цели должны быть SMART[16] (рис. 6).

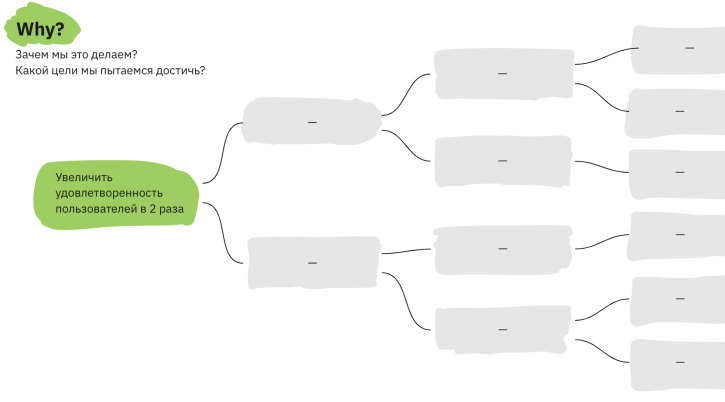


Рис. 6. Измеримая цель в Impact Map

Во время обсуждения головы кипят, потому что приходится много анализировать и рефлексировать. Первые пара

[16] Метод SMART, https://byndyu.ru/footnote/16

часов работы довольно сложны, но на старте закладывается правильный импульс для составления остальной карты и создаётся атмосфера доверия среди участников. Что я могу посоветовать, исходя из своей практики[17]:

1. Не торопитесь предлагать решения на этом этапе. Выслушайте заказчика, по-настоящему выслушайте. В ходе обсуждения вы успеете всё скорректировать и причесать, а пока запишите то, что есть у него в голове.

2. Самая распространённая проблема заключается в навязывании решений (этап What?) до того, как цели стали понятны. Инженерная мысль летит со скоростью света – заказчик только открыл рот, только начал говорить о своих целях, а мы уже создали в голове БД со всеми таблицами, придумали архитектуру и накидали куски кода. Зачем слушать дальше, если мы и так всё придумали? Будет ошибкой начать перебивать заказчика и предлагать решения. Запомните анекдот на тему: «Дима сказал „Привет“, а Даша мысленно сыграла свадьбу и родила троих детей».

---

[17] На HappyDev 2014 я проводил мастер-класс по составлению Impact Mapping и Story Mapping. Играть роль заказчика согласился руководитель проекта по обработке заявок на строительство. Все, кто пришёл на тренинг, были очень активны и сразу втянулись в процесс. Со временем мы осознали, что довольно сложно просто слушать заказчика и понять его проблему. Коллеги наперебой предлагали свои решения. В какой-то момент приходилось прерывать работу группы, напоминать, что мы должны больше слушать. Несколько раз из-за напряжённой атмосферы и давления участников заказчик принимал наши решения, отказываясь от своих. Я думаю, что все участники почувствовали важный баланс между тем, когда надо слушать заказчика, а когда надо предлагать решения.

3. Не переубеждайте заказчика на этом этапе. В самом начале вы не знаете его бизнес во всех тонкостях. Заказчики могут вам доверять как профессионалам в IT, и из-за этого быстро соглашаться с вашими предложениями. Вы сами не заметите, как на доске окажутся только те цели, которые вы навязали, а не те, с которыми заказчик жил всё это время.

4. Даже если цель трудноизмерима, то постарайтесь придумать критерий её достижения. Мысленно перенеситесь на финал проекта и подумайте, как вы узнаете, достигнута цель или нет.

5. Процесс выработки целей итерационный, не обязательно выжимать из заказчика все цели на первом круге.

6. Не надо вытягивать искусственные цели. Бывают проекты, которые просто есть, потому что инвесторам хочется поиграть в создателей ПО. С этим нужно смириться и свернуть работу по составлению Impact Mapping.

Who?

На этом этапе мы должны выявить всех, кто поможет оказать влияние на цель, кто поспособствует её достижению или помешает. В нашем примере это будут *Отдел маркетинга* и *Модератор форума*. По мнению заказчика, именно они могут изменить удовлетворённость пользователей (рис. 7).

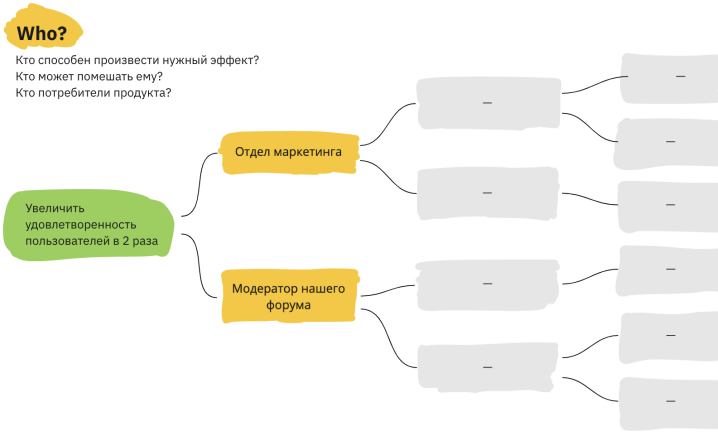


Рис. 7. Акторы в Impact Map, влияющие на цель

Здесь мы можем указывать конкретных людей, названия отделов, сегменты рынка и так далее. Выбирайте любой уровень абстракции, лишь бы он был адекватен вашему проекту.

How?

Теперь нам надо определить с набором действий. Например, модератор форума может попробовать давать ответы на вопросы в течение одной минуты. Как вы думаете, повысит это удовлетворённость пользователей? У нас есть *предположение*, что повысит, поэтому записываем этот «impact». То же самое делаем для остальных ролей (рис. 8).

Рис. 8. Гипотезы, которые помогут в достижении цели

Несколько рекомендаций:

1. Необязательно, но желательно, чтобы воздействия тоже были измеримыми. Мы написали не просто *Отвечать на форуме*, а *Отвечать на форуме в течение одной минуты* .

2. Не записывайте все возможные воздействия каждой роли. Нам нужны только те активности, которые приводят к достижению цели.

What?

Мы дошли до самого несущественного в Impact Mapping. В последнем узле нашей карты находится та самая корзина с

покупками, с которой обычно начинается работа над проектом. Разница в том, что теперь мы понимаем ценность каждой фичи, почему эта фича здесь и к чему приведёт её реализация (рис. 9).

Рис. 9. Финальная часть со списком задач

Несколько замечаний и лайфхаков:

1. В конечных узлах карты можно написать User Story или названия модулей/подсистем.

2. Эту часть карты можно подробно не расписывать, можно даже не заполнять, а лишь проговорить её основные моменты. Полный список всех User Story вы успеете создать на

User Story Mapping'e.

3. Здесь необязательно описывать IT-задачи. Вместо этого можно написать организационные преобразования и попросту любые необходимые вам действия на пути к цели.

4. Понимание целей даёт возможность создавать более дешёвые и быстрые решения. С помощью карты мы начинаем использовать не только руки разработчиков, но и голову – каждый член команды может принимать обоснованные решения.

Результаты создания Impact Mapping

Вот и готов наш Impact Mapping. Осталось приоритизировать каждую колонку. Не все цели одинаково важны, то же самое можно сказать про остальные узлы карты. Есть разные способы. Так как мы идём по пути простоты и визуализации, я могу рекомендовать ставить звёздочки. Каждому участнику даётся по пять звёзд, и он может ставить их куда считает нужным. Таким образом можно выявить самые приоритетные узлы.

Результат работы нужно повесить у всех на виду. Если команда распределённая, то следует выложить Impact Mapping в общую базу знаний или повесить перед экраном, который видят все участники разработки. Главная цель – обеспечить видимость и достижимость задач, ведь мы опираемся на них

при работе над проектом[18].

Фильтр входящих задач

Даже когда все согласились с целями проекта и способами их достижения, заказчик может добавить в проект фичу, которая ему очень нравится – pet feature. Мы можем отфильтровать её через цели, показать, что эта фича никаким образом не приведёт нас к достижению целей.

Аналогично мы будем фильтровать идеи по архитектуре и дизайну системы, которые исходят от команды разработки. Ведёт ли переделка архитектуры к более быстрому и дешёвому достижению цели? Если нет, то зачем нам это делать?

Обратите внимание, что мы двигаемся через прилаживание новых требований. Все входящие задачи должны пройти проверку на соответствие целям и должны «зацепиться» за одну из веток Impact Map. Такой подход даёт возможность обсуждать идеи: участники могут попробовать добавить свою задачу, видят проблему или ошибку в формули-

[18] Когда я рассказывал про Impact Mapping на AgileClub, коллеги заметили, что есть и другие способы понять стратегические цели. Например, можно использовать Lean Canvas, JTBD или собрать требования в проектной документации с описанием целей и заинтересованных сторон. На самом деле Impact Mapping не противоречит другим подходам и может использоваться вместе с ними. Лично мне он больше нравится, потому что:1. Это простая техника, которая способствует общению и взаимодействию, в ней нет бюрократии.2. Заказчикам, которые не разбираются в IT и производстве ПО, такой подход очень просто объяснить, хватает пары минут.3. Визуализация в виде mind map.

ровке или в месте, куда они хотели её добавить, пробуют переформулировать и раскрывают новые грани своей идеи. Именно это обеспечивает антихрупкость нашей системы, потому что какие бы неожиданные требования или изменения не пришли извне, мы знаем, как их обработать и приладить к текущему видению.

Модернизация Kanban-доски

Какая колонка идёт последней на вашей Kanban-доске? Могу поспорить, что это Release, Deploy, Done или что-то в этом духе. Последней колонкой на доске должна стать – «Проверка достижения цели». Недостаточно просто залить фичу на сервер, нам нужно проверить, как эта фича повлияла на продвижение к цели.

FAQ по Impact Mapping

Как продать создание Impact Mapping бизнесу перед началом проекта?

Лучше всего идти от проблемы. Попросите заказчика вспомнить случаи, когда было сделано много фич, а бизнес от этого только пострадал. Почему так произошло? Может, надо чётко описать цели?

Должна ли эта работа оплачиваться?

Да, обязательно. Составление Impact Mapping может занять несколько дней и имеет ценность для бизнеса.

Что если заказчик не хочет этого делать?

Вы как профессионал должны предоставить бизнесу прогноз на будущее. Расскажите о возможных проблемах и рисках. После этого дайте клиенту выбрать. Если вы донесли возможное проблемное будущее и клиент принял его, отказался от Impact Mapping'а при полной ясности последствий, то теперь это не ваша проблема; просто делайте ему фичу.

Всегда ли бизнес чётко знает свои цели?

Часто бизнес сам не до конца понимает свои истинные проблемы и цели. В ходе обсуждений представители бизнеса начинают спорить друг с другом: оказывается, что все поразному себе представляют цели. Это нормально. На этом этапе надо просто дать им выговориться, просто слушать и фасилитировать. В результате у всех проясняются мысли и синхронизируется понимание, что даёт толчок к развитию общих идей.

Impact Mapping является одной из активностей, которые сделают и заказчика, и разработчиков более счастливыми и эффективными. Как вы увидите в следующих главах, визуализация целей, трассировка от задач к целям и возможность постоянно изменять эту карту являются одними из основных инструментов создания антихрупкости вашей системы.

С помощью Impact Mapping и подобных практик вы только усилите ваш продукт от происходящих неожиданностей.

Глава 3. Углубление в Impact Map

Разбор нюансов создания Impact Map на примере притчи

Больше восьми лет я использую Impact Map для аналитики IT-продуктов. Я довольно активно делился знаниями об этой практике: писал статьи, выступал на конференциях с докладами и мастер-классами, рассказывал студентам в университетах и интернам в компании. Слушатели и участники мастер-классов легко улавливают, как создавать и использовать Impact Map – с теорией нет проблем. Тем не менее я вижу большие затруднения с применением этого подхода в реальной практике, когда нужно придумать и описать идеи для сложного IT-продукта.

Мы уже познакомились с этой техникой в прошлой главе, а сейчас я расскажу, каким образом формулируются идеи, которые являются самой сложной и самой ценной частью Impact Map. И ещё поделюсь своим ви́дением, как наиболее эффективно воспринимать каждую из частей Impact Map.

Нюансы Impact Mapping

В прошлой главе мы рассмотрели структуру Impact Map (рис. 5). Идея этого подхода в том, чтобы прорисовать связь от задач (Deliverable) к цели (Goal). По задумке, задачи ока-

зывают воздействие (Impact) на какую-то группу (Actor), и это воздействие приводит бизнес к цели. Таким образом, нет бесполезных задач, и чётко видно, что и *зачем* мы собираемся делать.

В этой, казалось бы, понятной схеме кроется много нюансов, которые вызывают ступор. Я приведу свою интерпретацию каждой части Impact Map и подскажу, что конкретно надо вписывать в каждую из них.

Goal – здесь нужно описать измеримый результат. Ключом к этой части будет ответ на вопрос, который вы зададите себе и заказчику: «По каким критериям мы поймём, что достигли успеха?» Такой вопрос помогает перенестись в будущее и подумать, как мы собираемся оценивать результат. Формулировка выбрана так, чтобы человек начал рефлексировать, и это не то же самое, что спросить «какая у вас цель?». Цели могут быть разные, а вот конечный результат в будущем оценят по каким-то критичным для бизнеса параметрам. Это и есть Goal.

Actor – на кого мы собираемся оказывать воздействие. Обычно сюда предлагают[19] писать тех, кто нам может помочь или помешать в достижении цели. Так можно делать, это неплохой совет, но он недостаточно нас фокусирует. Я предлагаю думать о том, чью жизнь мы хотим поменять с помо-

[19] ScrumTrek. Impact Mapping – инструкция по применению, https://byndyu.ru/footnote/19

щью воздействия, чтобы *это изменение* жизни привело *нас* к цели. Трюк в том, что мы не просто собираемся воздействовать, а надеемся, что сможем изменить поведение людей в нужную сторону. Такой взгляд заставляет точнее очерчивать границы для Actor и приходится тщательно обдумывать Impact.

Impact – воздействие, которое мы собираемся сделать. Это самая сложная часть, которая мало кому даётся. Я сравниваю её с созданием гипотезы в научном методе или рождением идеи в ТРИЗ, то есть техника понятна, но откуда берётся идея, как она рождается в голове, как научить человека эти идеи создавать, решительно неясно. Здесь ключом может стать понимание, что в этой колонке мы описываем ключевые идеи нашего продукта – то, что в итоге принесёт деньги. Если собрать все импакты из итоговой карты, то у нас должно получиться уникальное торговое предложение. Об этой части Impact Map будет вся остальная глава, где я расскажу на примере притчи, как описываются идеи.

Deliverable – это список задач или историй. В этой области все отлично умеют действовать.

Типичные ошибки Impact Mapping

Impact Map обычно не получается по следующим причи-

нам.

Цели неизмеримы или неясны. Это тот случай, когда хочется сделать много чего, но непонятно зачем. Антипаттерном будет закрыть глаза на отсутствие цели и накидать задач в работу, как делали во времена «плоских» ТЗ.

Описаны абстрактные юзеры, но неясно, как наши воздействия поменяют их поведение. Антипаттерном будет указать просто «пользователей», потому что потеряется связь с реальностью, в этих «пользователях» нет жизни и настоящих потребностей. Это приведёт к тому, что мы опишем идеи, но не будем понимать, как они повлияют на мир.

Вместо идей и гипотез (Impact), написаны user story или задачи. Эта самая частая проблема. Ещё раз: это очень (!) частая проблема, которая убивает всю идею Impact Map! В этом случае происходит подмена идей задачами, аналитики по привычке пишут to-do list вместо продуктовых гипотез.

Приведу пример, который поможет вам во всём разобраться.

Impact Mapping гончара

Я взял старую и известную притчу о гончаре. Гончар столкнётся с проблемой, сформулирует цель и попробует

придумать идею, как ему достигнуть цели.

Притча о гончаре и мальчишках

Жил на свете старый гончар. Он делал горшки, продавал их на базаре и на это жил. Но вот повадились соседские мальчишки бить его горшки. Он и просил их не делать этого, уговаривал, ругал, жаловался их родителям – ничего не помогало…

У гончара были три основные идеи и две группы людей, на которых он надеялся воздействовать, чтобы достичь своей цели. Изначально его Impact Map мог выглядеть как на рис. 10.

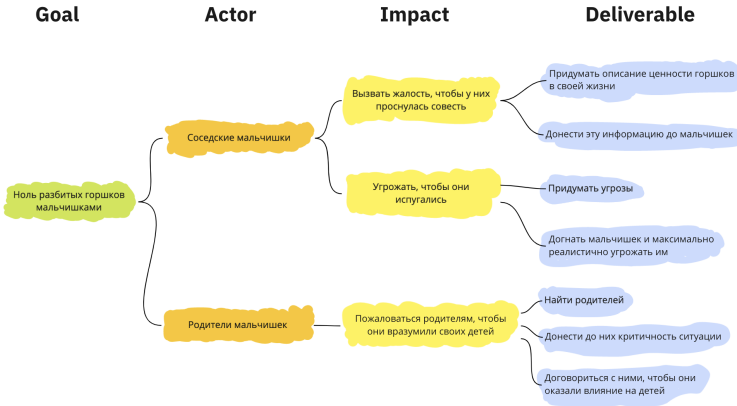


Рис. 10. Первые гипотезы и задачи гончара по достиже-

нию своей цели

Обратите внимание, что у каждой идеи есть блок «чтобы», который открывает практическую ценность идеи, показывает её основу.

Как следует из притчи, гончар проверил все три гипотезы и не добился результата. Не всегда гипотезы приводят к цели, это обычное дело, когда мы создаём продукты. Поэтому гончар продолжил поиск идей:

…Тогда он позвал мальчишек к себе во двор и сказал, что за каждый разбитый горшок будет платить им рубль. Мальчишки обрадовались, перебили все горшки, получили деньги и убежали. На следующий день гончар сказал, что денег у него мало, платить он может только 50 коп.

Мальчишки опять перебили все горшки, получили свои деньги и убежали. Следующий день опять повторилось то же самое, только за каждый разбитый горшок гончар заплатил всего 20 коп.

На следующее утро ребятня опять прибежала во двор. Старик вышел к ним и сказал: «Денег, ребятки, у меня почти совсем не осталось, потому что продавать мне было нечего. Теперь за каждый разбитый горшок я могу платить только одну копейку». – «Нашёл дураков бесплатно бить твои горшки!» – возмутились мальчишки. Больше горшков они не били.

Это очень важный момент! Я прошу вас не смотреть дальше, а самостоятельно сформулировать идею, которая помогла гончару достигнуть результата. Обязательно пропишите часть «чтобы». Опишите идею так, чтобы она была целиком и полностью понятна любому, кто её прочитает. Не оставляйте скрытых смыслов. К идее запишите набор задач, которые нужны для её реализации.

Ещё один абзац, пока вы думаете над формулировкой. Предлагаю вам собрать коллег и вместе попробовать сформулировать идею. Попробуйте записать несколько вариантов. Если у вас получится коротко и ясно записать идею, то можете считать, что вы готовы сделать Impact Map IT-продукта почти любой сложности.

Я надеюсь, что к этому моменту вы сформулировали минимум один вариант идеи, которая помогла гончару. На рис. 11 я приведу свой вариант, который кажется мне достаточно хорошим.

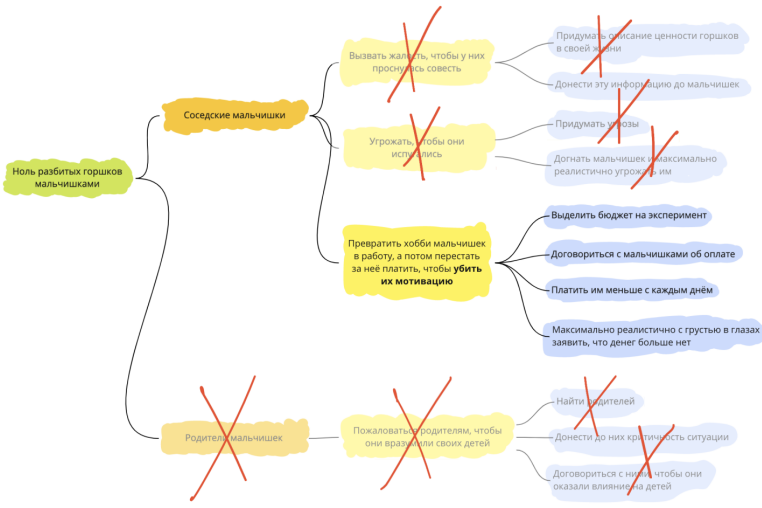


Рис. 11. Появление новой гипотезы у гончара

Когда гончар разочаровался в разговорах и угрозах (первые три гипотезы не сработали), ему пришлось искать идею, которая бы на самом деле воздействовала на мальчишек. Он понял, что бить горшки для них – это весело и забавно, поэтому нацелился на основу их мотивации. Он превратил весёлое хобби в работу, а потом перестал за неё платить.

Моя формулировка идеи звучит так: «Превратить хобби мальчишек в работу, а потом перестать за неё платить, чтобы убить их мотивацию». Здесь описано, что хочет сделать гончар и на чем основывается его надежда на достижение результата. Он надеется сильно снизить мотивацию мальчи-

шек бить его горшки. Похоже, что он отличный психолог!

Почему так сложно сформулировать идею?

При создании IT-продукта нам тоже нужно понимать внутреннее устройство мотивации пользователей, чтобы воздействовать на ключевые точки. Нам нужно понимать взаимосвязи внутри нашей системы, во внешних системах и их пересечении. По сути, формулируя идеи или гипотезы, мы работаем как профессиональные инженеры, которые из всего многообразия вариантов действий, умеют выбрать наиболее эффективное решение, основываясь на ограничениях и принципах работы системы.

Из практики я вижу, что формулировка идей даётся очень тяжело. Почему так происходит? Мне на ум приходит аналогия с решением прямых и косвенных задач. Оказывается, дошкольники довольно легко могут решить прямые задачи типа: *На ветке было три птицы, прилетело ещё шесть птиц. Сколько стало птиц на ветке?* Дети отвечают: девять. Если эту же задачу с этими же цифрами сделать косвенной, то есть проблемно-ориентированной, то дети теряются: *С ветки улетело три птицы, осталось шесть птиц. Сколько птиц изначально было на ветке?* Во второй задаче нужно как бы задом наперёд взглянуть на ситуацию. У взрослых с описанием идей возникает такая же проблема: они хорошо пишут

прямые задачи (Deliverable), но им тяжело даются косвенные/проблемные сценарии (Impact).

Рекомендую тренироваться в описании идей на простых жизненных сценариях и в повседневной жизни. Это очень помогает на реальной сессии Impact Map, когда нужно сформулировать идею достижения цели в сложном IT-продукте.

Глава 4. Пять самых важных составляющих процесса выпуска успешных продуктов

Практический подход к созданию IT-продуктов, которые достигают бизнес-целей

Опыт показывает, что нельзя просто взять и описать большой продукт в ТЗ, а потом реализовать его по описанию. Жизнь всегда оказывается шире, чем наше представление о ней. С другой стороны, эмпирический подход отражает постоянное углубление нашего понимания предметной области, бизнеса заказчика и изменений на рынке по мере создания и совершенствования продукта.

Нам посчастливилось создать успешные продукты, которые приносят прибыль заказчикам. Причём в самых разных областях: медицина, наружная реклама, госзакупки, налоги, логистика.

Я хочу рассказать про процесс создания таких продуктов и показать, что за магия приводит нашего заказчика к успеху. Не претендую на знание о самом совершенном процессе, который решает все проблемы, но конкретно этот подход работает, поэтому делюсь. После описания процесса мы возьмём реальный пример проекта, который прошёл по такому

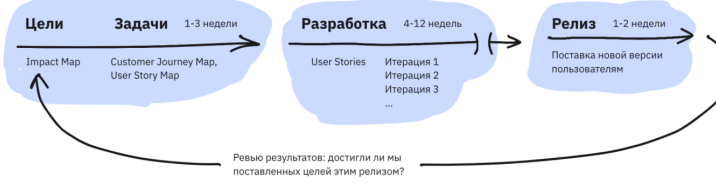
процессу.

Иногда я буду писать «проект», а иногда «продукт». Для себя мы не разделяем два этих понятия. В книге «Бережливое производство программного обеспечения. От идеи до прибыли»[20] есть интересная мысль о том, что любой проект можно рассматривать как продукт, поэтому при разработке ПО мы используем подходы по созданию продуктов.

Общий взгляд на процесс

Разработка ПО – это процесс создания знания. В начале работы неопределённость очень высока. Заказчик витает в облаках со своей идеей, разработчики неглубоко знают предметную область.

Мы много экспериментировали с разными методологиями – от жёстких до гибких, от Agile до Lean – и пришли вот к процессу, описанному на рис. 12.



[20] Мэри и Том Поппендик. Бережливое производство программного обеспечения. От идеи до прибыли, https://byndyu.ru/footnote/20

Рис. 12. Схема работы со всеми инструментами и обратной связью

Вся работа должна быть визуализирована: от процесса и целей до мелких пожеланий и требований. Мы исходим из простого принципа: «You cannot improve what you cannot see».

Первое знакомство с проектом всегда начинаем с целей. Спрашиваем: «Как вы поймёте, что достигли успеха?», «Что для вас является успешным продуктом?», «По каким критериям вы оцените его успешность?» и так далее, пока цели не материализуются. Приоритезируем цели и строим пути их достижения.

Следующий шаг понимания будущей системы – Customer Journey Mapping (CJM), но не в классическом варианте, а в том, как его описал мой коллега Андрей Шапиро в статье «Схематизация опыта с CJM и Service Blueprint. Практика гибридной нотации»[21]

[21] Андрей Шапиро. Схематизация опыта с CJM и Service Blueprint. Практика гибридной нотации, https://byndyu.ru/footnote/21

Конец ознакомительного фрагмента.