O'REILLY®

# ПОЛЬЗОВАТЕЛЬСКИЕ ИСТОРИИ

## Искусство гибкой разработки ПО

ПИТЕР®

Джефф Паттон

## Annotation

Пользовательские истории – это метод описания требований к разрабатываемому продукту. В книге рассказано, как правильно использовать данную технику, чтобы сфокусироваться на поставленной задаче и пожеланиях клиента, а не распыляться на реализации второстепенных функций. Автор книги показывает, как данный подход не только ускоряет и систематизирует разработку, но и улучшает взаимопонимание в команде.

---

# Джефф Паттон
# Пользовательские истории. Искусство гибкой разработки ПО

*Посвящается Стейси, Грейс и Зоэ. Без вашей поддержки у меня ничего бы не вышло.*

*В память Люка Барретта, дорогого коллеги и учителя. Люк оказал огромное влияние на мою жизнь, а также на судьбы многих других людей.*

# Предисловие Мартина Фаулера

Одно из самых выгодных последствий популярности разработки программного обеспечения (ПО) по методологии Agile – распространение идеи разбиения больших, объемных требований на компактные фрагменты. Благодаря этим фрагментам – историям – отслеживать прогресс разработки проекта намного проще. Когда истории реализуют постепенно, каждый раз полностью интегрируя их в проект, всем очевидно, что проект понемногу растет. Рассматривая истории, которые приносят пользователям очевидную выгоду, разработчики могут планировать развитие проекта и определять, над чем нужно работать в следующую очередь. К тому же такая прозрачность подталкивает пользователей к активному участию в разработке – они больше не гадают месяцами и годами, чем занята команда разработки.

Тем не менее такое разбиение может иметь и негативные последствия. В частности, очень легко перестать понимать, в чем заключается общее предназначение ПО – что и как должна делать система. В итоге у вас в руках может оказаться множество кусочков, которые никак не складываются в единую картину. Или вы можете создать бессмысленную и бесполезную систему, так как утонули в деталях и забыли, что в действительности нужно пользователям.

Построение карт историй (story mapping) – это техника, позволяющая увидеть цельную картину, чего не удастся сделать с помощью простого набора историй.

Вот, собственно, и все – описание книги уместилось в одном предложении, но этого вполне достаточно, чтобы оценить преимущества метода. Обзор цельной картины облегчает взаимодействие с пользователями, позволяет избежать разработки ненужных функций, а также ориентирует на релевантный опыт использования. Когда я обсуждаю с коллегами по Thought-Works применяемый ими процесс разработки пользовательских историй, построение карт регулярно упоминается в качестве основной техники. Часто оказывается, что коллеги изучили эту технику как раз на семинарах Джеффа, поскольку именно он разработал ее и лучше всего может ей обучить. С помощью данной книги еще больше людей смогут узнать об этой технике непосредственно из первых уст.

Но эта книга не только для тех, у кого на бедже или в профиле написано что-нибудь вроде «бизнес-аналитик». Наверное, самым большим разочарованием для меня за 10 лет внедрения методологии Agile стало то, что множество программистов рассматривают истории как некие односторонние указания со стороны аналитиков. С самого начала предполагалось, что истории будут вызывать *обсуждение*. Если вы и в самом деле хотите получить эффективное ПО, которое может органично встроиться в человеческую деятельность, то тех, кто создает программы, необходимо рассматривать как живой источник идей о возможностях, ведь именно программисты лучше всех знают, что могут делать эти программы. Программисты должны хорошо понимать, что хотят получить их пользователи, и взаимодействовать с ними, создавая карты историй, где полностью учитываются пользовательские цели. Программист, умеющий составлять карты историй, может видеть пользовательскую среду куда более широко, чем тот, кто этого не умеет, и, следовательно, принимать участие в проектировании ПО, что улучшит качество работы.

Когда Кент Бек, впервые предложивший термин «история», воплотил свои идеи в разработке ПО, он назвал коммуникацию ключевым моментом эффективности команды.

Истории – строительные блоки коммуникации между разработчиками и теми, кто использует результаты их труда. Карты историй организуют и структурируют эти строительные блоки и тем самым стимулируют процесс коммуникации, крайне важный для разработки ПО в целом.

*Мартин Фаулер, 18 июня 2014 года*

# Предисловие Алана Купера

В научно-фантастическом романе Мэри Шелли «Франкенштейн» безумный доктор Франкенштейн создает чудовище из фрагментов тел мертвых людей, а затем оживляет его с помощью диковинной на тот момент силы электричества. Конечно, мы знаем, что на самом деле это невозможно. Вы не можете создать что-то живое, просто сшив вместе случайные части тел.

Тем не менее разработчики программного обеспечения все время пытаются сделать именно это. Они разрабатывают прекрасные новые функции для программ, одну за другой, а потом удивляются, почему лишь немногие пользователи восхищаются их продуктом. Ключ к загадке в том, что в качестве инструмента для проектирования и дизайна программисты используют свои методы разработки ПО, но эти средства совсем не взаимозаменяемы.

Более чем разумно *программировать* только одну функциональность ПО в каждый момент времени. Это идеальная стратегия, проверенная временем. Кроме того, многолетним опытом разработки было доказано, что использование такого подхода при проектировании цифровых продуктов, как одна функциональность в каждый момент времени, порождает монстров, подобных Франкенштейну, а не качественные программы.

Хотя процессы проектирования программного обеспечения и его непосредственной разработки тесно связаны, по своей сути они совершенно различны и, как правило, их выполняют разные люди с разным набором навыков. Если заставить программистов, подобно дизайнерам интерфейсов, проводить многие часы за наблюдением работы пользователей и выделением поведенческих паттернов, они просто на стену полезут. В то же время дизайнер, погрузившись в код и алгоритмы, почувствует себя выброшенным на необитаемый остров.

Но когда две составляющие одного процесса – проектирование и разработка – выполняются одновременно, работа идет искрометно, а у продукта есть все шансы родиться живым и дышащим. Именно командная работа вдыхает в монстра жизнь и заставляет людей полюбить его.

Хотя идея командной работы не является ни новой, ни особенно революционной, эффективно воплотить ее в жизнь нелегко. Особенности работы программистов – темп, ритм, привычный язык – сильно отличаются от методов, присущих дизайнерам.

Представители каждой из сторон могут быть решительными, способными, дисциплинированными, но у них есть одно общее слабое место: очень трудно описать дизайнерскую проблему в терминах программирования и так же нелегко сделать обратное. Две родственные дисциплины не имеют общего языка. Именно там, на стыке этих двух дисциплин, и работает Джефф Паттон.

Метод построения карт историй, созданный Джеффом, находят полезным разработчики, и точно так же его оценивают дизайнеры. Карты историй – Розеттский камень нашего цифрового века.

На самом деле гибкие методологии – не самая лучшая среда для дизайна приложений, несмотря на популярность противоположного мнения. Да, такая философия разработки дружественна дизайну, и это очень хорошо, но сама по себе она не поможет вам создать продукт, который понравится пользователям. В то же время мы столько раз видели, как отличный, хорошо документированный дизайн передается разработчикам – работающим по

Agile или нет, – а они в процессе реализации ухитряются загубить самую его суть.

Метод построения карт историй Паттона перекидывает мост через эту пропасть. Основа дизайна взаимодействия – это выяснение мельчайших потребностей пользователей и верная их интерпретация. Дизайнерская история, являющаяся формальной версией пользовательской истории, остается неизменной на протяжении всего периода разработки.

Современный мир бизнеса доказал, что команде из 200–300 человек почти невозможно создать продукт, который понравится пользователям. В то же время сообществу стартапов известно, что команда из четырех-пяти человек *способна* создать небольшой продукт, который люди полюбят, но даже эти маленькие продукты со временем растут и теряют свой блеск. Большие программы используются большой аудиторией и решают сложные коммерчески успешные задачи. Ждать, что вы легко их изучите и в процессе работы станете получать удовольствие, сложно да и просто смешно.

Единственный путь создать большую программу, не похожую на ужасного Франкенштейна, – научиться объединять дисциплины проектирования и разработки программного обеспечения. Никто не умеет делать это лучше, чем Джефф Паттон.

*Алан Купер, 17 июня 2014 года*

# Предисловие Марти Когана

Мне невероятно повезло – я имел возможность работать с представителями лучших в мире компаний и групп разработки разных технологий. Эти люди создают программы, которые вы любите и которыми пользуетесь каждый день. Люди, которые буквально меняют мир.

Кроме этого, мне часто приходилось помогать компаниям, у которых дела идут не так здорово. Это были стартапы, пытающиеся запустить хоть что-то работающее, прежде чем кончатся деньги. Компании покрупнее, выбивающиеся из сил в попытке воплотить в жизнь свои последние разработки. Команды, безуспешно пытающиеся повысить эффективность бизнеса. Лидеры, раздраженные тем, как много времени занимает переход от идеи к воплощению. Инженеры, конфликтующие с владельцами своих продуктов.

Из этого всего я вынес в первую очередь понимание того, насколько по-разному создают технологические продукты самые популярные компании на рынке и все остальные. И я не говорю сейчас о каких-то мелких различиях. Я имею в виду решительно все: подход руководителей к делегированию полномочий командам, способ взаимодействия команд, отношение организации к финансированию, комплектованию штата и выпуску продуктов, культуру, а также то, каким образом объединяют продукт, дизайн и технологии, чтобы разрабатывать самые эффективные решения для клиентов.

Эта книга называется «Пользовательские истории. Искусство гибкой разработки ПО», но очень скоро вы заметите, что она повествует о чем-то большем, чем такая простая, но мощная техника, как построение пользовательских карт историй. С помощью книги можно проникнуть в самую суть того, как команды сотрудничают, общаются и в конце концов приходят к созданию великолепных продуктов.

У многих из вас никогда не было возможности с близкого расстояния наблюдать за сильной командой в процессе работы над проектом. Возможно, опыт работы в компаниях, где вы трудились раньше или трудитесь сейчас, – все, что у вас есть. Поэтому я попытаюсь рассказать о том, насколько самые лучшие команды отличаются от всех остальных.

С благодарностью в адрес Бена Хоровица и его книги «Хороший менеджер продукта, плохой менеджер продукта» я приведу здесь лишь важнейшие различия между сильными и слабыми командами.

• У хороших команд есть четкое видение своего продукта, а каждый член команды страстно заинтересован в успехе. Плохие команды – просто наемники.

• Хорошие команды черпают идеи и вдохновение из системы ключевых показателей эффективности, наблюдения за клиентами, анализа полученных от клиентов сведений о результатах использования их продукта, а также из стремления постоянно применять новейшие технологии для эффективного решения проблем. Плохие команды получают требования из запросов заказчиков и отдела продаж.

• Хорошие команды понимают, кто их ключевые партнеры, им известны ограничения, которые вынужден учитывать бизнес клиентов, и поэтому они стараются находить решения, не только работающие для пользователей и заказчиков, но и учитывающие условия среды. Плохие команды просто выполняют требования партнеров.

• Хорошие команды компетентны во множестве техник, позволяющих быстро опробовать новые идеи для развития продукта и определить, какие из них следует воплощать

в первую очередь. Плохие команды тратят часы на совещания, где пытаются составить списки приоритетов.

• В хороших командах обожают мозговые штурмы с участием лучших умов всей компании. Плохие команды ощетиниваются, если кто-то извне осмеливается внести какое-то предложение.

• В хороших командах инженеры, дизайнеры и менеджеры работают бок о бок, все время обмениваясь опытом и информацией о функционале, пользовательском опыте и технологических возможностях. В плохих командах эти специалисты разделены согласно своим обязанностям, а запросы одних к другим передаются через служебные записки и совещания, проводимые по расписанию.

• Хорошие команды постоянно пробуют новые идеи и вводят различные усовершенствования, но делают это осторожно, чтобы не навредить эффективности бизнеса. Плохие команды ждут разрешения что-то попробовать.

• У участников хороших команд непременно есть полный набор навыков для создания сильных продуктов, например, с хорошим дизайном взаимодействия. Плохие команды даже не знают, кто такие дизайнеры интерфейсов.

• В хороших командах заботятся о том, чтобы у инженеров ежедневно находилось время поработать с прототипом продукта для поиска идей по его улучшению. В плохих командах инженерам показывают прототипы на планировании спринта при оценке объема работы.

• Хорошие команды еженедельно напрямую общаются с конечными пользователями и заказчиками, чтобы лучше понять их и узнать их мнение о последних изменениях и идеях. Плохие команды считают, что достаточно собственного мнения.

• Хорошие команды знают, что не все их любимые идеи будут работать для заказчиков, но даже те, что будут, потребуют нескольких доработок, прежде чем приведут к получению желаемого результата. В плохих командах просто делают то, что записано в плане, довольствуясь датами совещаний и показателями качества.

• Хорошие команды понимают важность быстродействия и регулярных прогонов для успешного внедрения инноваций; им известно, что скорость обеспечивается правильной организацией работы, а вовсе не напряженным трудом. В плохих командах все жалуются на медленную работу, обвиняя в этом недостаточно усердно трудящихся коллег.

• После оценивания затрат на реализацию запроса хорошие команды берут на себя жесткие обязательства и стараются убедиться, что они трудятся над жизнеспособным решением, которое будет эффективно работать как для заказчиков, так и для бизнеса. Плохие команды жалуются, что им приходится работать на эффективность продаж.

• Хорошие команды выстраивают свою работу так, что могут немедленно оценить, как их продукт используется, и сделать выводы, базирующиеся на этих данных. Плохие команды считают аналитику чем-то, что хорошо бы иметь.

• Хорошие команды постепенно и непрерывно обновляют продукт, зная, что постоянный поток небольших обновлений означает стабильное и надежное решение для заказчиков. Плохие команды проводят ручное тестирование в конце огромной фазы разработки, а затем выкатывают сразу все обновления.

• Хорошие команды концентрируются на своей целевой аудитории. Плохие команды концентрируются на конкурентах.

• Хорошие команды устраивают вечеринку, когда достигают значительного улучшения ключевых показателей эффективности. Плохие команды празднуют финальный релиз чего-

нибудь.

Я понимаю: вы, вероятно, хотите знать, что общего со всем этим имеют карты историй. Я уверен, вы удивитесь, поняв, в чем дело. По той же причине я преданный поклонник построения карт историй.

На моем пути встретилось не так уж много экспертов Agile, по моим меркам достаточно квалифицированных для того, чтобы оказать реальную помощь серьезной команде, разрабатывающей продукт, и поднять ее работу на тот уровень, в котором нуждается компания и которого она заслуживает. Джефф Паттон – один из них. Я наблюдал, как он в разгар разработки засучив рукава трудится вместе со всей командой. Я представлял его в компаниях, потому что он эффективен. Команды обожают его, так как при всей своей компетентности он совершенно лишен высокомерия.

Время, когда менеджеры день-деньской собирали и документировали требования, дизайнеры концентрировались на косметических улучшениях, а инженеры тонули в коде, для самых лучших команд давно ушло в прошлое. Настало время стремиться к будущему и вам.

*Марти Коган, 18 июня 2014 года*

# Об авторе

За 20 лет практической работы Джефф Паттон убедился, что не существует единственно правильного способа проектирования и разработки программного обеспечения, а вот неправильных путей существует великое множество.

Для помощи организациям в улучшении их работы Джефф использует более чем 15-летний опыт работы с широким спектром продуктов – от системы онлайн-заказа запасных частей для самолетов до электронных медицинских карт. В то время как многие процессы разработки концентрируются на скорости и продуктивности, Джефф уравновешивает эти факторы созданием продуктов, которые обеспечивают полезность для бизнеса и успех на рынке.

Джефф решил специализироваться на подходах Agile с тех пор, как работал в команде экстремального программирования в 2000 году. В частности, он специализируется на интеграции эффективного дизайна пользовательского взаимодействия и менеджмента продуктов в мощные инженерные методы.

В настоящее время Джефф работает как независимый консультант, тренер процессов Agile, тренер процессов дизайна продуктов и инструктор. Множество статей, эссе и презентаций, посвященных различным аспектам разработки продуктов Agile, можно найти на сайте agileproductdesign.com и в Crystal Clear Алистера Коберна. Джефф – основатель и модератор дискуссионной группы Yahoo! по теме юзабилити в Agile, колумнист в StickyMinds.com и IEEE Software, сертифицированный тренер Scrum, а также обладатель премии Agile Alliance’s 2007 Gordon Pask за вклад в развитие Agile.

# Вступление

*Живи в этом, плавай в этом, смейся в этом, люби в этом,*
*А еще оно уберет подозрительные пятна с простыней,*
*Развлечет во время визита к родным*
*И превратит сэндвич в банкет.*

*Том Уэйтс. Проходите, не стесняйтесь*

На самом деле эта книга должна была быть совсем небольшой… чем-то вроде памфлета. Вообще-то я собирался всего лишь описать простую технику, которую назвал *построением карт историй*. Я вместе с другими создаю простые карты, чтобы облегчить совместную работу, а также представить себе ощущения, возникающие при использовании нашего продукта.

*Карта историй помогает нам держать фокус на пользователях и их опыте, в результате чего взаимодействие улучшается и продукт становится несравнимо лучше.*


Составлять карты до смешного просто. Работая вместе с другими, я озвучиваю историю работы с продуктом, записывая каждый шаг, предпринимаемый пользователем, на листочках-стикерах и наклеивая их слева направо. Затем мы возвращаемся к началу и обсуждаем каждый шаг в деталях, записывая подробности на листочках и наклеивая их сверху вниз под соответствующим шагом. В результате получается простая, напоминающая таблицу структура, излагающая историю слева направо и раскрывающая детали сверху вниз. Быстро и очень интересно. А эти детали образуют *бэклог (backlog)*[1] историй для наших проектов, разрабатываемых по Agile.

Сложно ли написать об этом книгу?

Оказывается, и в самых простых вещах могут скрываться сложности. Поэтому описание того, зачем вообще строить карту историй и что с ней делать после того, как она построена, а также различных способов ее использования заняло немало страниц. Многовато для простой методики, какой я ее считал.

Если вы используете процесс разработки, описанный в методологии Agile, то ваши бэклоги и так, наверное, заполнены пользовательскими историями (user story). Я думал: раз создание историй является настолько распространенной практикой, писать о них книгу будет напрасной тратой времени. Но, как оказалось, я ошибался. Через полтора десятилетия после того, как истории впервые были описаны Кентом Беком, они стали наиболее популярны, а также наименее правильно понимаемы и используемы, чем когда-либо. Это меня огорчает. А главное, это сводит на нет все выгоды, которые мы получаем от составления карт историй.

Поэтому в этой книге я хотел бы скорректировать как можно больше недоразумений, связанных с использованием историй в разработке программного обеспечения по методологиям Agile и Lean. Вот почему, говоря словами Тома Уэйтса, я «превращаю этот сэндвич в банкет».

# Почему я?

Я люблю создавать. Когда я разрабатываю какую-нибудь функциональность для программного обеспечения, а люди с удовольствием ею пользуются, это очень радует и мотивирует меня. И я не слишком люблю методологии. Я бы сказал, мне надо разобраться в принципе какой-то методики или практики, чтобы как следует ею овладеть. Только сейчас, имея более чем 20-летний опыт разработки программного обеспечения, я начинаю понимать, как учить других тому, что умею сам. Я также понимаю, что то, чему я учу, постоянно меняется. На этой неделе я что-то изучил, а на следующей оно уже изменилось. Способы объяснить это другим людям меняются почти так же часто. Все это многие годы удерживало меня от написания книги.

Но время наконец пришло.

Несомненно, истории и построение карт – вещь прекрасная. Множество людей извлекли из них пользу. Использование историй благотворно повлияло как на рабочий процесс, так и на продукт, который создавался. Но в то время, как одни люди замечали улучшения, другие, которых было больше, мучились во время работы с историями больше, чем когда-либо прежде. Вот это мне и хотелось бы прекратить.

Изменить ситуацию я надеюсь с помощью этой книги. Если мне удастся добиться хотя бы небольших улучшений, я буду считать это успехом.

# Если вы используете истории и страдаете, эта книга – для вас

Уже довольно много организаций внедрило у себя методологии Agile и Lean, поэтому, вполне возможно, вы уже успели угодить в одну из ловушек, возникающих из-за неверного понимания концепции историй. Вот некоторые из них.

• Поскольку истории позволяют вам сконцентрироваться на создании небольших фрагментов ПО, легко *перестать видеть цельную картину*. В результате получается типичный продукт-франкенштейн, каждому пользователю которого очевидно, что он состоит из разрозненных, не связанных друг с другом частей.

• Когда вы работаете над продуктом значительных размеров, создание маленьких частичек одна за другой *заставляет людей задумываться, когда же вы наконец закончите и что же получится в результате*. Как будто вы строитель.

• Поскольку главное в концепции историй – это обсуждение, *люди часто забывают вести записи*. В результате предмет обсуждения и достигнутые соглашения забываются.

• Поскольку в хороших историях предполагается наличие критериев приемки, мы концентрируемся на определении этих критериев. Но этот процесс и описание создаваемого продукта – не одно и то же. В результате *команда не может закончить запланированную работу в запланированные сроки*.

• Поскольку хорошие истории должны быть написаны с позиции пользователя, но существует множество аспектов, которых пользователь просто не видит, члены команды утверждают: «У этого продукта нет пользователей, так что здесь пользовательские истории не подходят».

Если вы уже угодили в одну из этих ловушек, я постараюсь прояснить все вызвавшие их недоразумения. Вы узнаете, как оценить полную картину, продуктивно обсуждать цели и задачи пользователей и создавать хорошее ПО.

# Для кого еще эта книга

Для вас, конечно. Особенно если вы ее уже купили. Я вот считаю, что вы сделали умную инвестицию. Если же просто взяли у кого-то книгу почитать, лучше закажите свой экземпляр, а эту верните, как только получите собственную.

Во всяком случае, чтение книги будет особенно полезным для специалистов следующих областей.

*Продукт-менеджеры и UX-специалисты в коммерческих компаниях, производящих продукты,* должны прочесть эту книгу, чтобы перекинуть мостик от мира пользовательского опыта и работы продукта к тактическим планам и элементам бэклога. Если вы испытываете трудности, пытаясь перейти от представления о продукте к отдельным деталям, которые должна создать ваша команда, истории вам помогут. Если вам сложно заставить других людей поставить себя на место пользователей, карта историй вам поможет. Если вы никак не можете увязать вместе хороший дизайн взаимодействия и практическое проектирование продукта, вам поможет эта книга. И если пытаетесь провести эксперимент в стиле стартапа Lean, она тоже будет вам полезна.

*Представители заказчиков, бизнес-аналитики, а также продукт-менеджеры в организациях, занятых в сфере информационных технологий,* должны прочесть эту книгу, чтобы возвести мосты между пользователями, разработчиками и другими заинтересованными сторонами. Если вы тратите множество усилий, чтобы все заинтересованные лица в вашей компании пришли наконец к какому-либо соглашению, карты историй вам помогут. А если разработчики затрудняются, пытаясь нарисовать цельную картину, истории будут полезны и здесь.

*Тренеры процессов Agile и Lean*, если они хотят помогать командам и отдельным людям действовать эффективнее, должны прочесть эту книгу. Кроме того, подумайте только, насколько неверное представление об историях сформировалось у сотрудников вашей организации! Применяйте истории, простые упражнения и практики, описанные в этой книге, чтобы помочь вашим командам развиваться.

*И наконец, все остальные*. При использовании процессов Agile мы чаще всего ожидаем плодотворной работы с историями от людей, исполняющих обязанности бизнес-аналитиков или представителей заказчиков, но по-настоящему эффективной работа станет, если основы будут известны *всем*. Если люди не понимают самых простых вещей, вы часто будете слышать жалобы, что истории плохо расписаны, или слишком длинные, или недостаточно детализированы. Эта книга поможет, но не так, как вы ожидаете. Вы вместе с другими читателями узнаете, что создание историй – это не способ лучше писать требования, а путь к более продуктивным и организованным обсуждениям. Эта книга поможет вам сформулировать, какие виды обсуждений необходимы, чтобы в любой момент у вас имелась нужная информация.

Надеюсь, вы отнесли себя к одной или нескольким из описанных здесь групп. Если нет, подарите эту книгу кому-нибудь, кто им соответствует.

А если это все-таки вы, давайте начнем.

# Используемые соглашения

Как я полагаю, это не первая книга о разработке программного обеспечения, которую вы читаете, поэтому ничто не должно вас особенно удивить.

## Подзаголовки внутри каждой главы будут подсказывать вам направление темы

Обращайте на них внимание, чтобы найти что-либо нужное или пропустить материал, неинтересный вам в данный момент.

> *Ключевые моменты оформлены примерно так. Представляйте, что это нужно прочитать чуть более громко, чем остальной текст.*

Если вы читаете по диагонали, двигайтесь от одной ключевой точки к другой. Если вас что-то заинтересовало или просто вы не находите сказанное банальным, прочтите предыдущий и последующий текст. Таким образом вы разберетесь в деталях.

> Врезки используются для описания:
> • интересных, но не критически важных аспектов. Возможно, они развлекут вас, по крайней мере я на это надеюсь;
> • конкретных упражнений. Вы сможете использовать их, чтобы начать практиковаться в чем-либо специфическом;
> • историй и примеров, предоставленных другими. Возможно, вы почерпнете из них хорошие идеи и сможете применить их в своей компании.

Эта книга разбита на разделы. Вы можете читать по разделу за раз или обращаться к ним, чтобы найти какие-то идеи для решения конкретных задач, занимающих вас в данный момент.

# Как устроена эта книга

Однажды я купил замечательный лазерный принтер. Первое, что я увидел, открыв коробку, был бумажный буклет, на котором красовалась надпись большими яркими буквами: «СНАЧАЛА ПРОЧТИТЕ ЭТО». Я задумался, стоит ли так поступать на самом деле, ведь, как правило, я пренебрегаю инструкциями. Но позже очень радовался, что послушался этого предупреждения, потому что, как оказалось, в разных местах внутри принтера было закреплено много пластиковых деталей, чтобы уберечь его от повреждений при транспортировке, и, если бы я включил принтер, не убрав их, скорее всего, он был бы безнадежно испорчен.

Эта история, наверное, кажется здесь неуместной, но это не так.

Книга тоже содержит главу «Сначала прочтите это», где описаны две критически важные концепции, а также приводятся термины, которые я буду использовать в тексте. Мне хотелось бы, чтобы вы ознакомились с этим материалом, прежде чем приступите к дальнейшему чтению. Если вы начнете работать с картами историй прежде, чем хорошенько усвоите эту главу, я не могу гарантировать вашу безопасность.

## Построение карт историй с высоты птичьего полета

Главы 1–4 дадут вам общее представление о построении карт историй. Если вы уже используете их и имеете некоторый опыт, эта часть книги обеспечит достаточно материала, чтобы немедленно приступить к делу.

Глава 5 предоставит вам отличную возможность попрактиковаться в ключевых концептах, используемых для составления наилучшей карты историй. Попробуйте проделать это в своем офисе с группой коллег – в результате каждый участник получит полезный опыт. Я обещаю: созданные вами карты со временем будут становиться все лучше и лучше.

## Интуитивное понимание пользовательских историй

В главах 6–12 рассказано многое о пользовательских историях: как они работают в реальности, как организовать их использование в проектах Agile или Lean наилучшим образом. В дополнение к картам историй там приведены несколько маленьких примеров, которые могут оказаться полезными в ежедневной практике разработки. Даже если вы ветеран Agile, обещаю, вы узнаете об историях что-то новое для себя. А если вы в историях новичок, то узнаете достаточно, чтобы поразить самого заносчивого Agile-всезнайку в офисе.

## Лучшие бэклоги

Главы 13–15 раскроют перед вами подробности жизненного цикла историй. Мы обсудим конкретные практические приемы, которые помогут использовать истории и карты историй. Постепенно мы пройдем от открывающихся перспектив к составлению бэклога, заполненного историями, описывающими жизнеспособный продукт. Вы узнаете, как построение карт историй и другие приемы могут помочь вам на каждом этапе работы.

## Лучшая разработка

Главы 16–18 шаг за шагом посвятят вас в тонкости тактики использования историй. Вы научитесь доводить истории до конца, не упускать их из виду в процессе разработки, добиваться их точного исполнения, а также извлекать опыт из каждой истории, трансформированной вами в рабочее ПО.

Я обнаружил, что последние несколько глав в большинстве книг по разработке ПО – просто бесполезный перевод бумаги. Обычно их можно безболезненно пропустить. К сожалению, в моей книге я ничего такого не написал и вам придется прочесть ее целиком. Могу утешить вас лишь тем, что в каждой главе вы найдете какие-то полезные приемы и уроки, которые сможете немедленно применить на практике.

Перейдем к делу.

# Сначала прочтите это

В этой книге нет введения.

Да, вы все прочитали правильно. И сейчас, наверное, задаетесь вопросом: «Почему же в книге Джеффа нет введения? Может быть, он забыл его написать? Не пора ли ему на покой? Или введение съела его собака?»

Нет, я не забыл написать введение. И мне не пора на покой (по крайней мере я туда не собираюсь). И моя собака не съедала введения, хотя насчет морской свинки своей дочери я не был бы так уверен. Это просто потому, что за долгое время я убедился: авторы тратят кучу времени, пытаясь убедить меня прочесть их книгу, и большая часть этого времени приходится на введение. Самое интересное во всех книгах начинается примерно с третьей главы. Кроме того, я уверен, что не я один всегда пропускаю введение.

Так что эта книга начинается прямо здесь.

И вам ни в коем случае нельзя пропускать эту часть, так как на самом деле она самая важная. Я буду доволен, если вы вынесете из книги всего две мысли. Вот эти две мысли.

• Цель работы с историями не написание идеальных историй.

• Цель разработки продуктов не создание продуктов.

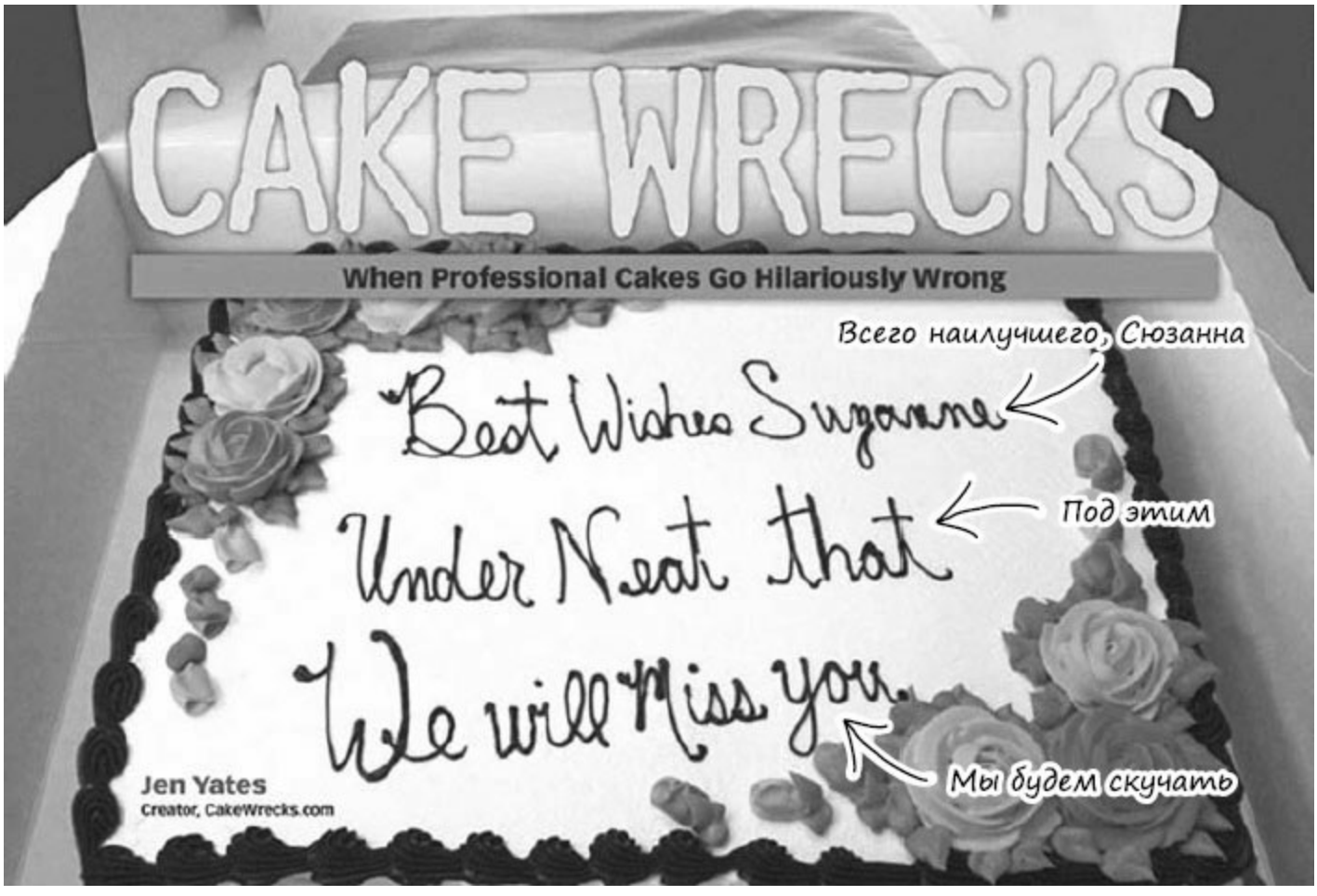
Сейчас я все объясню.

# Игра в испорченный телефон

Вы наверняка помните, как в детстве играли в дурацкую игру «Испорченный телефон». Вы говорили кому-то шепотом слово или фразу, тот озвучивал это следующему, и так повторялось с каждым играющим, а затем последний участник провозглашал совершенно искаженное сообщение, и все дружно смеялись. Сегодня мы часто играем в эту игру всей семьей за обедом. Кстати, родителям на заметку: это отличный способ занять детей, скучающих, когда взрослые беседуют.

В мире взрослых мы также продолжаем играть в эту игру, разве что больше не шепчем. Мы пишем длиннющие документы и проводим торжественные презентации, чтобы дать поручение кому-то другому, а затем получить от него совершенно не то, чего ожидали. А этот человек использует эти документы, чтобы написать еще больше документов и передать их еще большему количеству людей. Только вот, в отличие от детской игры, в конце концов нам становится не до смеха.

Когда люди читают письменные инструкции, они истолковывают их совершенно по-разному. Если вам сложно в это поверить (в конце концов, все это тоже написано!), читайте дальше – вот несколько примеров того, как инструкции действуют абсолютно неверно.




Перед вами обложка книги Джен Ятис «Торты, которым не повезло», опубликованной в издательстве Эндрю Мак-Милла (спасибо Джен и Джону Ятис за предоставление иллюстрации). Книга получилась по мотивам ее очень забавного сайта cakewrecks.com (только не ходите туда, если у вас нет по меньшей мере часа свободного времени). На сайте собрана коллекция по-идиотски украшенных тортов, логика их создателей не поддается

объяснению, хотя Джен и предпринимает попытки сделать это. Так, одна из самых часто встречающихся и в книге, и на сайте тем – неверно понятые требования. Джен, конечно, не называет их *требованиями*, ведь это слишком формальный термин, вместо этого она употребляет слово *«записи»*, так как исполнитель слушает и записывает, понимая буквально то, что слышит. Глядя на эти фото, я могу вообразить сотрудника кондитерской, который слушает заказчика и записывает его пожелания, а потом передает их кому-то, кто будет украшать торт.

*Заказчик:* «Здравствуйте, я хотел бы заказать торт».

*Работник:* «Конечно, что бы вы хотели на нем написать?»

*З.:* «Вы могли бы написать “Всего хорошего, Алиса” фиолетовым цветом?»

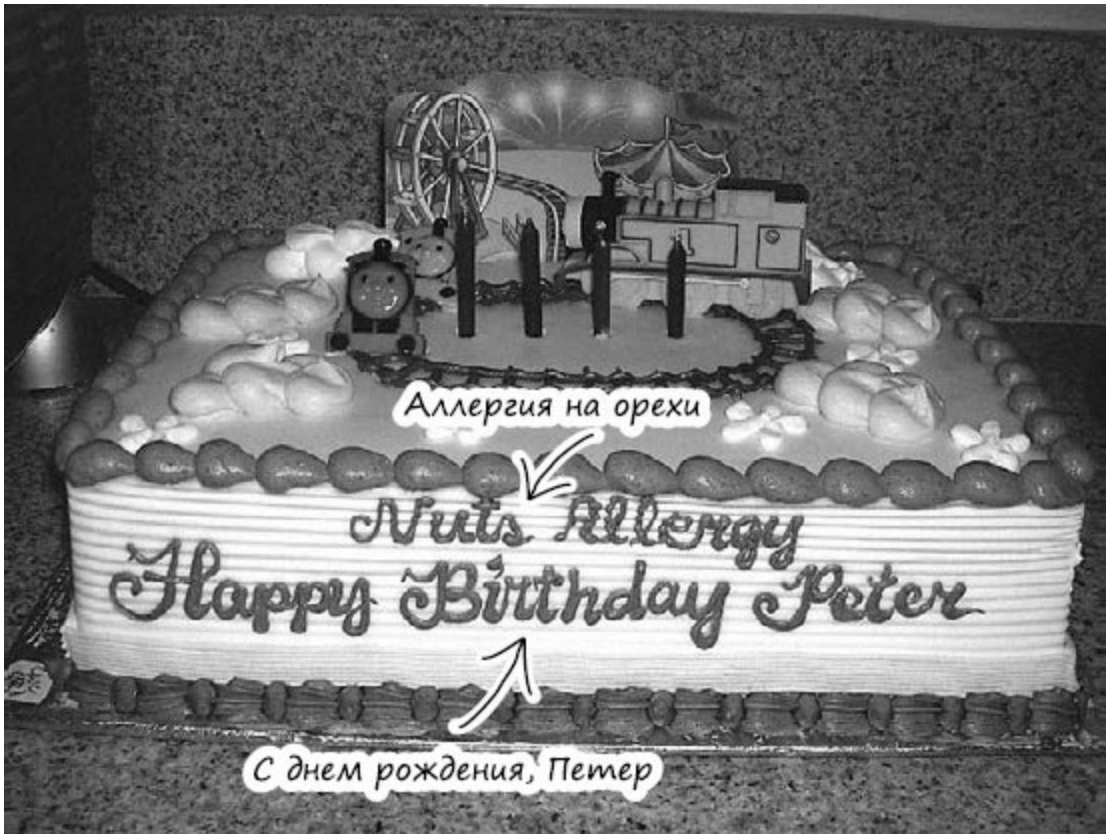
*Р.:* «Конечно».

*З.:* «А вокруг надписи пусть будут звезды».

*Р.:* «Без проблем. Я записал ваши пожелания и прямо сейчас передам их кондитеру-декоратору. Торт будет готов к завтрашнему утру».

Вот что получилось в результате.


Вот еще один пример. В разработке программного обеспечения такие вещи мы называем *нефункциональными требованиями*.




Все это очень весело, и можно посмеяться над 20 долларами, пропавшими зря. Но часто понесенные потери бывают куда более серьезными.

Может быть, вы слышали о неудачной попытке отправить в 1999 году на Марс орбитальный климатический зонд, в результате аварии которого NASA понесло убытки в размере 125 млн долларов[2]? Если не слышали, вот суть случившегося. Если когда-нибудь какой-либо проект по самые уши тонул в бумажной документации, это был, несомненно, проект NASA. Но несмотря на огромное количество требований и других документов, зонд вышел из строя по той простой причине, что NASA пользовалось метрической системой измерений, а инженеры партнерской компании Lockheed Martin, которые разрабатывали навигационные команды для двигателей аппарата, – британской. В результате никто не знает, где сейчас находится зонд, и некоторые надеются, что он нашел свое место на солнечной орбите где-то за Марсом.

Какая ирония – мы вкладываем огромные усилия в написание документов, чтобы общаться более ясно и избегать недоразумений, а в итоге получаем прямо противоположный результат.

*Общие документы не дают единого понимания.*

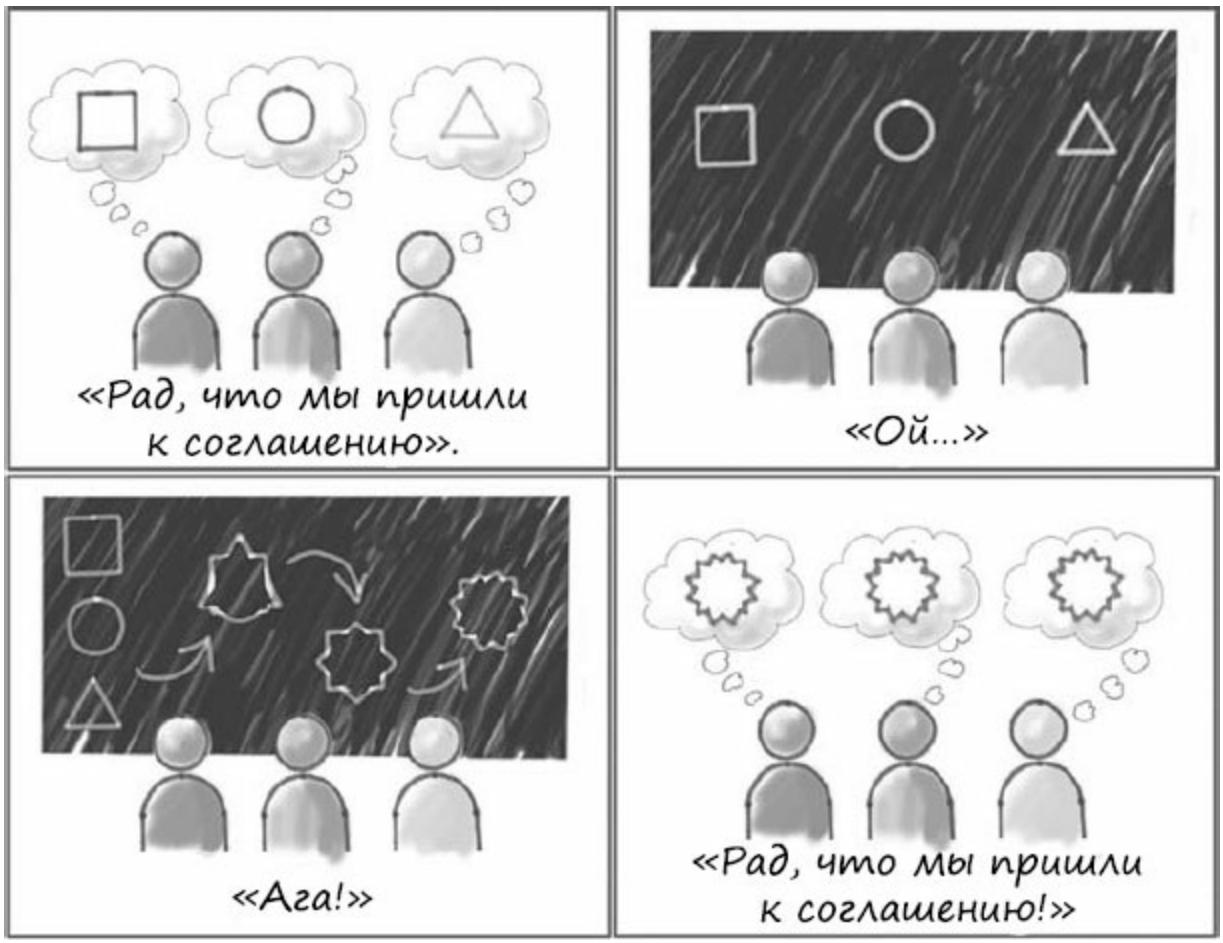
Притормозите на минутку и запишите это. Запишите на стикере и положите себе в карман. Представьте, что эти слова вытатуированы где-то на вашем теле и вы можете их видеть, когда утром приступаете к работе. Когда вы прочитаете их снова, вспомните истории, которые я вам сейчас рассказываю.

*Единое (общее) понимание* – это когда мы оба хорошо понимаем, что и почему представляет себе каждый. Очевидно, что между кондитерами и людьми, дававшими им инструкции, такого понимания не было. И в NASA какой-то важный начальник не обеспечил одинакового понимания того, как работает система управления, всеми участниками

процесса. Уверен, если вы трудитесь в сфере разработки ПО, то быстро вспомните аналогичную ситуацию: два человека были уверены, что пришли к соглашению относительно функции, которую хотели добавить в проект, но оказалось, что они представляли себе совершенно разные вещи.

# Единое понимание – это невероятно просто

Мой бывший коллега Люк Баррет первым показал мне этот комикс, чтобы описать проблему. Я спросил, где он его видел, но он так и не вспомнил (кто-то, возможно, не получает авторских отчислений). Годами я наблюдал, как Люк демонстрирует эти четыре слайда в презентации PowerPoint, и всегда воспринимал их как нечто забавное, но довольно банальное. Видимо, я туговато соображаю: лишь много лет спустя осознал, что этот комикс иллюстрирует, наверное, самую важную особенность работы с историями при разработке ПО.




Суть в том, что если у меня есть какая-то идея, я ее записываю на бумаге, а *вы* затем это читаете, то вполне можете представить что-то совершенно иное, чем я. При этом можно спросить каждого: «Вы согласны с содержанием этого документа?», и все дружно ответят: «Да! Да, мы согласны!» Но если мы соберемся вместе и начнем обсуждение, вы можете описать мне, что думаете, а я смогу задать вопросы. Беседа пойдет живей, если мы можем описать наши мысли, рисуя картинки либо фиксируя идеи на стикерах или карточках. Если у каждого из нас будет возможность объяснить свои мысли с помощью слов и картинок, мы придем к общему мнению. Однако здесь мы осознаем, что все понимают всё по-разному, и это очень неприятно. Но по крайней мере теперь проблема известна.

Это не значит, что один из нас прав, а другой – нет, напротив, мы оба видим разные и важные аспекты. Комбинируя и уточняя наши различающиеся представления, мы в конце концов придем к общему мнению, включающему все лучшие идеи. Вот почему вынесение идей вовне так важно. Мы можем перерисовывать эскизы или передвигать с места на место

стикеры – таким образом мы в буквальном смысле держим в руках свои идеи, и это замечательно. В этот момент мы формируем общее мнение, а это очень тяжело сделать, оперируя только словами.

По окончании обсуждения мы по-прежнему можем упоминать те же самые функции или предлагать улучшения, но сейчас определенно говорим об одних и тех же вещах. Мы синхронизированы и точно знаем, что движемся вместе в одном направлении. Вот качественный уровень, к которому мы стремимся. И, к сожалению, неосязаемый. Вы не можете увидеть или потрогать общее мнение, но можете его почувствовать.

# Перестаньте пытаться написать идеальную документацию

Огромное количество людей до сих пор верит, что есть некий идеальный способ составлять документы. Поэтому они думают, что, когда люди читают документ и у них возникают некие разногласия, это происходит лишь из-за небрежности автора или читателя. На самом деле ни то ни другое не верно.

Правильный ответ: просто не занимайтесь документами.

Перестаньте пытаться составлять идеальные документы.

Возьмите и напишите что-нибудь, неважно как. Затем проведите эффективное обсуждение с использованием и слов, и картинок, чтобы прийти к общему мнению.

*Истинная цель использования историй – достижение единого понимания.*

Если вы применяете в разработке истории, но не обсуждаете их с командой с помощью слов и рисунков, то вы неправильно используете истории.

Если при чтении этой книги вы поставили перед собой цель научиться лучше писать истории, то это плохая цель.

# Хорошие документы похожи на фотографии из отпуска

Если я покажу вам одну из фотографий, сделанных во время отпуска, вы увидите моих дочерей на пляже и вежливо скажете что-то вроде: «Очень красиво». Но когда я сам смотрю на это фото, то вспоминаю определенный пляж на Гавайях, куда нам пришлось больше часа ехать на машине по ухабистой грунтовой дороге, а затем еще полчаса идти по лавовым полям. Я помню, как дети ныли и жаловались, уверяя, что ничего не может быть хуже, и я был с ними вполне согласен. Но, придя наконец на место, мы провели великолепный день на потрясающем, почти безлюдном пляже, ради чего мы и решились на эту поездку. Кульминацией этого чудесного дня стали черепахи, пришедшие погреться на песке у воды.

Конечно, когда вы смотрите на фотографию, вы не можете знать всего этого, так как не были там. А я помню, потому что я там был.

Хорошо это или плохо, но именно так работают документы.

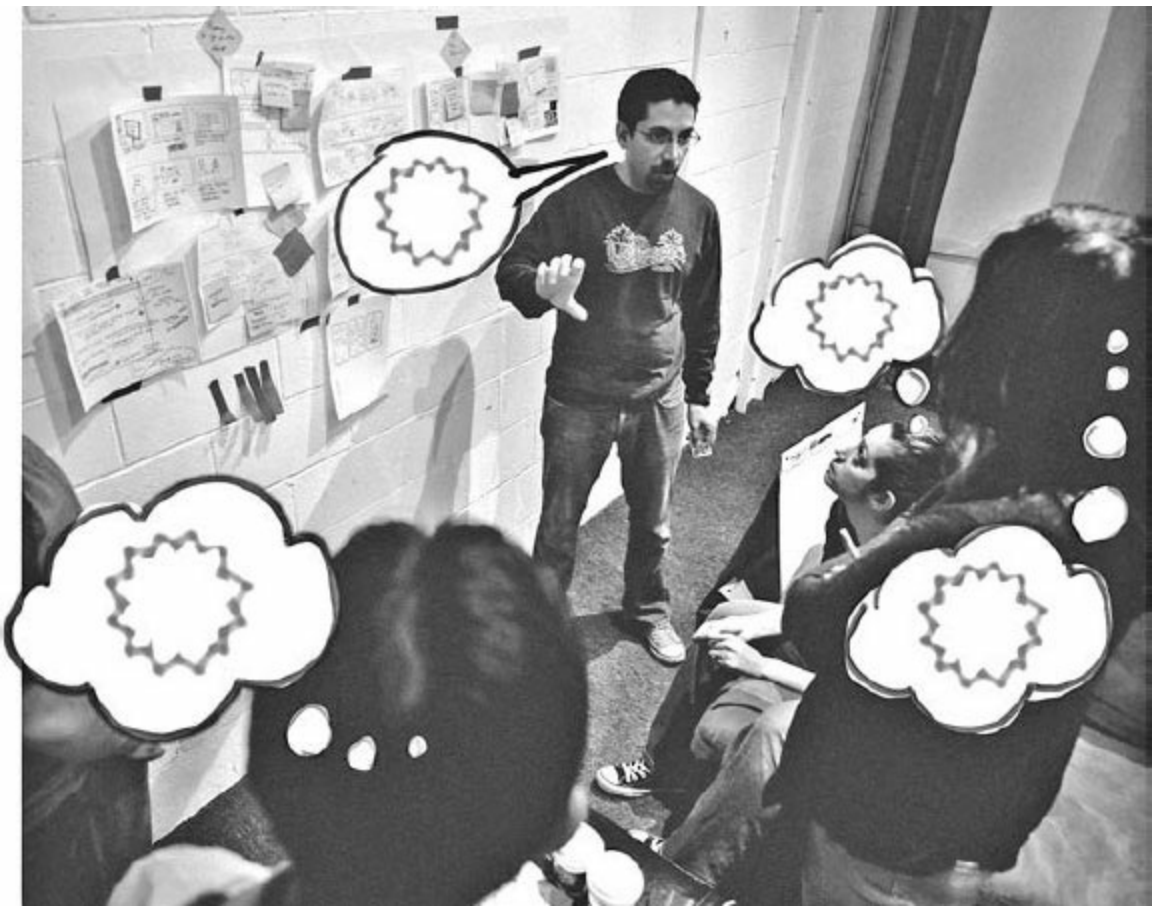
Если вы принимаете участие во множестве обсуждений создаваемого продукта, а затем документируете полученные результаты, то обязательно должны показать документ кому-то еще, кто тоже при этом присутствовал. Вы оба можете согласиться, что он хорош. Но помните: единое понимание заключается в деталях, которые в нем не отражены. Другой читатель, не присутствовавший при обсуждении, не извлечет из документа всего, что знаете вы. Даже если он говорит, что все понятно, не верьте этому. Найдите время, чтобы на основе документа рассказать историю точно так же, как я рассказал о своем отпуске, используя фотографию.

# Документируйте, чтобы активизировать воспоминания

Я пару раз слышал шутку: «Мы перестали писать документацию, и поэтому у нас теперь Agile». Это называется: кто знает – тот поймет, потому что на самом деле процесс, управляемый историями, требует для работы множества документов. Но зачастую эти документы совсем не похожи на традиционные задокументированные требования.

Применять Agile – значит много обсуждать, рисовать эскизы, записывать, возиться с карточками или стикерами. Приносить на обсуждение документы, а затем уносить их, испещренные пометками на полях и исчерканные маркерами. Воздух наполнен энергией и движением. Если вы сидите в конференц-зале и кто-то один сводит все сказанное в систему управления историями, определенно здесь что-то не так.

Когда вы описываете историю, почти все может быть использовано как инструмент коммуникации. А поскольку мы обсуждаем истории, пишем массу заметок и рисуем кипы рисунков, надо их где-то хранить. Мы складываем их, чтобы позднее просмотреть, фотографируем, включаем в разные документы.



Но помните: самое важное и нигде не записанное – это то, что мы вспомним при прочтении. Вот он, фактор фотографии из отпуска.

Обсуждайте, рисуйте эскизы, записывайте, наклеивайте стикеры, а затем фотографируйте полученные результаты. Лучше даже запишите краткое видео команды, обсуждающей записанное на доске. Вы вспомните массу очень важных деталей, чего никогда нельзя добиться документированием.

*Чтобы облегчить вспоминание, фотографируйте, а также записывайте*

*короткие видео по результатам обсуждений.*

# Обсуждайте то, что действительно нужно

Многие люди уверены, что их работа – формирование и сбор требований. Но это не так.

*На самом деле ваша работа – изменить мир.*

Я сказал это, чтобы привлечь ваше внимание, и понимаю, что это звучит слишком пафосно. Эта фраза и в самом деле обычно ассоциируется с миром во всем мире, ликвидацией нищеты и тому подобными вещами вроде попыток политиков договориться друг с другом. Но вообще-то я серьезно. Любая отличная идея, которую вы реализуете как решение для продукта, в небольшой, а может, в значительной степени меняет мир людей, использующих этот продукт. Если это не так, вы потерпели неудачу.

# До и после

Есть маленькая модель изменения мира, которую лично я использую и всегда держу в голове, вы тоже должны помнить о ней на протяжении всего обсуждения историй и выстраивания одинакового понимания.

Вот как я изображаю эту модель.


Модель начинается с исследования того, каков мир в настоящий момент, до начала работы. Когда вы смотрите на мир в состоянии «до», то предполагаете обнаружить людей недовольных, раздраженных, растерянных или злых. Однако мир очень велик, поэтому мы концентрируемся в основном на тех, кто работает с нашим ПО, а также на потенциальных его пользователях. Когда вы понаблюдаете, чем они занимаются, какие инструменты используют и как именно берутся за дело, скорее всего, вам в голову придет несколько идей, например таких.

- Какие новые продукты вы можете создать.
- Какие функции добавить в существующий продукт.
- Как улучшить создаваемые продукты.

В какой-то момент вам нужно будет обсудить свои идеи с другими людьми, после чего можете начать проектировать дизайн и писать *спецификации*. Если вы планируете передать эту работу кому-то другому, то и в самом деле можете употреблять слово *«требования»*. Но очень важно помнить, что требования – просто другое название возникших у нас идей о том, как помочь людям.

Имея эти требования, мы движемся по некоему процессу и в результате выпускаем продукт – какое-то программное обеспечение, которое выходит в свет – в мир в состоянии *«после»*. Мы с вами очень надеемся, что люди, которые изначально были недовольны, раздражены, растеряны или злы, после этого станут счастливыми. И они счастливы не потому, что им принесли красивую коробку с бантиком, ведь программное обеспечение в наши дни не доставляется в коробках. И не потому, что они прочитали протоколы изменений или скачали мобильное приложение в свой телефон. Они счастливы, потому что, используя программу, или сайт, или мобильное приложение, или что-то еще, созданное вами, замечают

изменения к лучшему.

Конечно, вам не удастся угодить всем сразу. Сказать вам об этом должна была мама. Одни люди будут радоваться любым изменениям больше остальных, а другие так и останутся недовольными, каким бы тяжелым ни был ваш труд и как бы вы ни старались найти самое лучшее решение.

# Суть не в программах

Все, что находится между идеей и выпуском продукта в свет, называется *объемом работы*. Это то, что мы создаем. Люди, занятые разработкой программного обеспечения по Agile, обычно измеряют *скорость* работы и стараются увеличить ее. Те, кто создает ПО, разумеется, беспокоятся о стоимости готового продукта, в том числе о временных затратах на его создание, планируемых и реальных.

Но в таком случае объем работы не является основным, ведь мы заботимся не о количестве работы. Нас интересует то, что получается в итоге, – *реальный результат*. Реальный результат – то, что получается в результате работы (отсюда и название), и его нелегко оценить, поскольку невозможно измерить результат, не доведя работу до конца. Кроме того, реальный результат нельзя измерить количеством добавленных функций или новыми возможностями, предъявленными пользователям. По сути, мы фиксируем, что именно люди делают иначе для достижения своей цели теперь, в результате сделанных вами изменений, и, самое главное, стала ли их жизнь несколько лучше[3].

Ну вот вы и изменили мир.

Вы добавили в него что-то изменяющее способ достижения людьми своих целей, и когда они это используют, мир для них меняется.

Если помните, наша цель – не создать новый продукт или функцию. Если вы обсуждаете эту функцию, то стараетесь понять, для кого она предназначена, как эти люди справляются со своими задачами сейчас и что можно поменять для них в будущем. Ожидание позитивных изменений мотивирует людей к использованию вашего продукта.

> *Плодотворное обсуждение историй включает в себя вопросы «Для кого?» и «Почему?», а не только «Что?».*

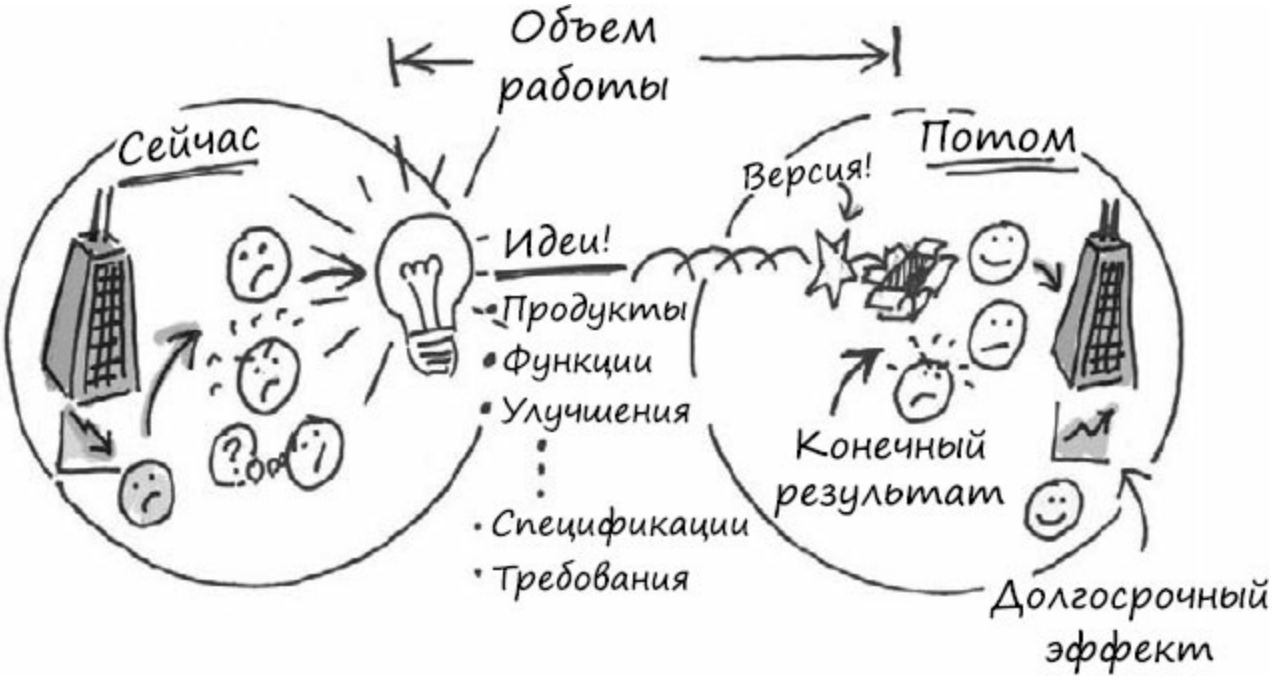
# Ладно, не только о людях

Я думаю о людях не меньше, чем все остальные, но, понятное дело, мы говорим не только о том, чтобы сделать их счастливыми. Если вы работаете в компании, которая платит вам и другим работникам зарплату, вам приходится концентрироваться на том, что конкретно больше всего помогает вашей организации зарабатывать больше денег, защищать свой рынок или захватывать новые, работать более эффективно. Ведь если в вашей компании дела идут не очень, то у вас нет средств (или работы), чтобы кому-нибудь помочь.

Поэтому модель придется немного переработать. На самом деле начинать нужно с обзора вашей организации изнутри. Там вы найдете еще больше людей, недовольных жизнью, чаще всего потому, что их бизнес работает не так хорошо, как они надеялись. Чтобы улучшить положение вещей, эти люди чаще всего предлагают сфокусироваться на конкретных пользователях или заказчиках с целью создать или усовершенствовать нужные им продукты. В итоге, как видите, все сводится к людям, потому что

> *ваша компания не получит того, чего хочет, пока заказчики или пользователи не получат того, чего хотят они.*

Развитием процесса становится выбор людей, на которых мы будем фокусироваться, проблем, которые нужно решить, а затем идей, которые можно внедрить в работающее ПО. После этого – если заказчики покупают, пользователи используют и все они довольны, – бизнес, который спонсировал данную разработку, начинает получать выгоду. Это может выразиться в возрастании прибыли, снижении издержек, повышении лояльности заказчиков, увеличении доли рынка. Таким образом, многие сотрудники вашей компании получают удовлетворение. Это и вас должно порадовать, ведь вы только что помогли своей компании получить выгоду, одновременно принеся пользу реальным людям. Это стратегия «вин-вин», или стратегия обоюдной выгоды.




Вот какими могут быть последствия получения хорошего реального результата – я

называю их *долгосрочным эффектом*. Реальные результаты вы чаще всего видите сразу после выпуска продукта, но долгосрочный эффект этой работы проявляется позже.

# Программируйте меньше

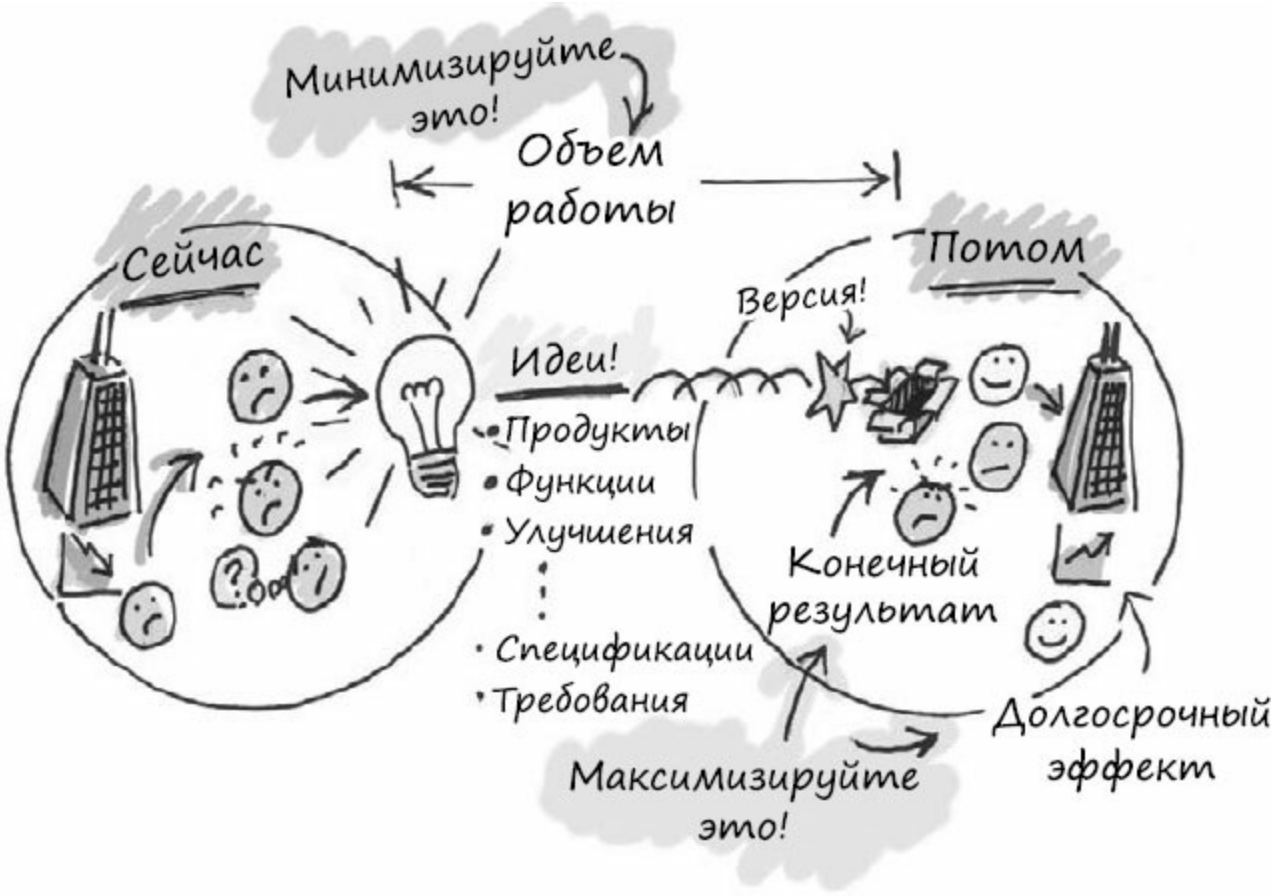
Да, это неприятная правда в мире программного обеспечения и, как я подозреваю, во множестве других сфер. Но я хорошо знаю эту кухню. И мне точно известно, что

> *у нас никогда не будет достаточно времени или ресурсов, чтобы разработать все, что нужно, – никогда!*

Одним из самых больших недоразумений в разработке ПО является то, что мы пытаемся выдать как можно больший результат за как можно меньший промежуток времени. Да, если бы у нас было слишком много работы, то увеличение скорости, конечно, помогло бы, верно? Но если вы трезво посмотрите на ситуацию, то поймете, что наша работа заключается не в том, чтобы сделать больше, – она в том, чтобы сделать меньше.

> *Минимизируйте объем работы, максимально увеличивайте результат и долгосрочный эффект.*

Хитрость здесь в том, чтобы внимательно рассмотреть людей, чьи проблемы вы пытаетесь решить. В эту группу входят лица, которые *принимают решение о покупке* вашего продукта для решения каких-то задач в своей организации, а также те, кто непосредственно с ним работает, то есть *пользователи*. Иногда это одни и те же люди, иногда – нет.




В вашем бизнесе есть множество возможных пользователей и заказчиков, на которых вы можете сфокусироваться. Ваша бизнес-стратегия должна подсказать, кого именно выбрать,

чтобы получить желаемый долгосрочный эффект. Но уверяю вас: ни у одного бизнеса нет достаточных ресурсов, чтобы сделать счастливыми *всех* – это просто невозможно!

Поймите меня правильно: выработка большего объема продукта за меньшее время – это всегда хорошо, но никогда не решает всех проблем.

# Страшные слова на букву «Т»

На протяжении почти всего первого десятилетия моей карьеры в IT, которое я провел за созданием ПО для розничной торговли, я вообще обходился без слова *«требования»* – ну или очень редко его слышал. Этот термин попросту не годился для обозначения того, что я делал. У меня было множество отдельных заказчиков, каждый из которых имел собственное представление о том, что ему подходит. Также я всегда помнил, что работаю на компанию, которая делает деньги, продавая мой продукт. Фактически я проводил долгие часы на выставках, помогая компании продавать наш продукт широкому кругу покупателей. В конце дня я знал, что мне придется работать с этими заказчиками после того, как они получат продукт, разработанный мной и моей командой, и я прилагал все усилия, чтобы действовать в их интересах. Это не означало, конечно, что я мог дать каждому абсолютно все, что он хотел, потому что все хотели разного. Кроме того, ни моя компания, ни команда не располагали неограниченным временем, поэтому я усердно работал над выявлением хотя бы того, что я могу изменить, чтобы люди были довольны. Казалось бы, такое положение вещей должно раздражать, но на самом деле это очень интересная часть работы.

По мере того как компания росла, на работу поступало все больше людей, привыкших к традиционному процессу разработки. Однажды ко мне пришла начальница другой команды и сказала:

«Джефф, нужно, чтобы вы внесли вот эти изменения в продукт, над которым сейчас работаете».

«Нет проблем, – ответил я, – только расскажите мне, для кого мы делаем эти изменения и какие задачи люди будут решать с их помощью».

Что я услышал в ответ?

«Это нужно для соответствия требованиям».

«Я вас понял, – кивнул я. – Мне только нужно знать подробнее, для кого мы внедряем эти штуки, как эти люди будут их использовать, а также какой этап их рабочего процесса изменится».

Она посмотрела на меня так, словно я был самым тупым человеком на свете, и повторила с нажимом:

«Это. Для. Соответствия. *Требованиям*».

И в этот момент я осознал, что слово *«требования»* на самом деле означает *«заткнись»*.

Вот что означают требования для большинства людей. Они перестают говорить о людях и проблемах, которые надо решить. Правда же состоит в том, что, даже если вы сделаете лишь часть того, что требуется, пользователи вполне могут остаться довольными[4].

Помните: ваша работа заключается не в том, чтобы достичь соответствия требованиям. Ваша работа в том, чтобы изменить мир.

# Вот и всё

Запомните эти три вещи, даже если больше ничего не вынесете из книги.

• Истории не требования в письменной форме; создание историй с использованием слов и картинок – механизм выработки одинакового понимания.

• Истории не требования, а дискуссии о решении задач вашей организации, заказчиков и пользователей. Результатом этих дискуссий становится соглашение о том, что именно нужно разработать.

• Ваша работа состоит не в том, чтобы больше разрабатывать за меньшее время, – вы должны стремиться увеличить результат и долгосрочный эффект, получаемые в ходе изменений, которые вы решили сделать.

Изначальный смысл историй – принципиально новый способ мышления при столкновении с проблемами, которые возникают при совместной работе над программным обеспечением (а также в любой другой деятельности). Если вы можете эффективно работать вместе с другими людьми и создавать нечто решающее проблемы, то скоро начнете править миром. Или как минимум той его частью, где работают с вашим продуктом.

Раз уж вы читаете эту книгу, я надеюсь, вы хотите изучить самую суть использования историй. Я надеюсь, вы работаете вместе с другими людьми и в ходе этого сочиняете истории о ваших пользователях и заказчиках, а также о том, как им помочь. Я надеюсь, вы рисуете картинки и составляете большие модели из стикеров. Я надеюсь, вы увлеченный и творческий человек. Я надеюсь, вы ощущаете, что создаете что-то значимое. Это чистая правда, если вы делаете все правильно. И это доставляет вам удовольствие.

А сейчас настало время обсудить самый интересный момент в составлении историй – построение карты историй.

# Глава 1. Общая картина

«Обожаю разработку Agile! Каждые пару недель у нас начинает работать что-то новенькое! Но, кажется, я уже запутался и не вижу общей картины».

Если бы я получал по 10 центов с каждого, от кого слышу что-то подобное о разработке Agile, то у меня была бы уже… хм… целая куча десятицентовиков. Словом, я слышал это множество раз. Вы, наверное, тоже слышали, а может, и сами высказывались в этом духе. Что ж, у меня для вас хорошие новости. Следование методологии Agile и управление разработкой посредством историй не всегда означает непременную утрату масштабного видения. Вы можете продолжать дискутировать о своем продукте в целом и одновременно видеть изменения, происходящие каждые несколько недель.

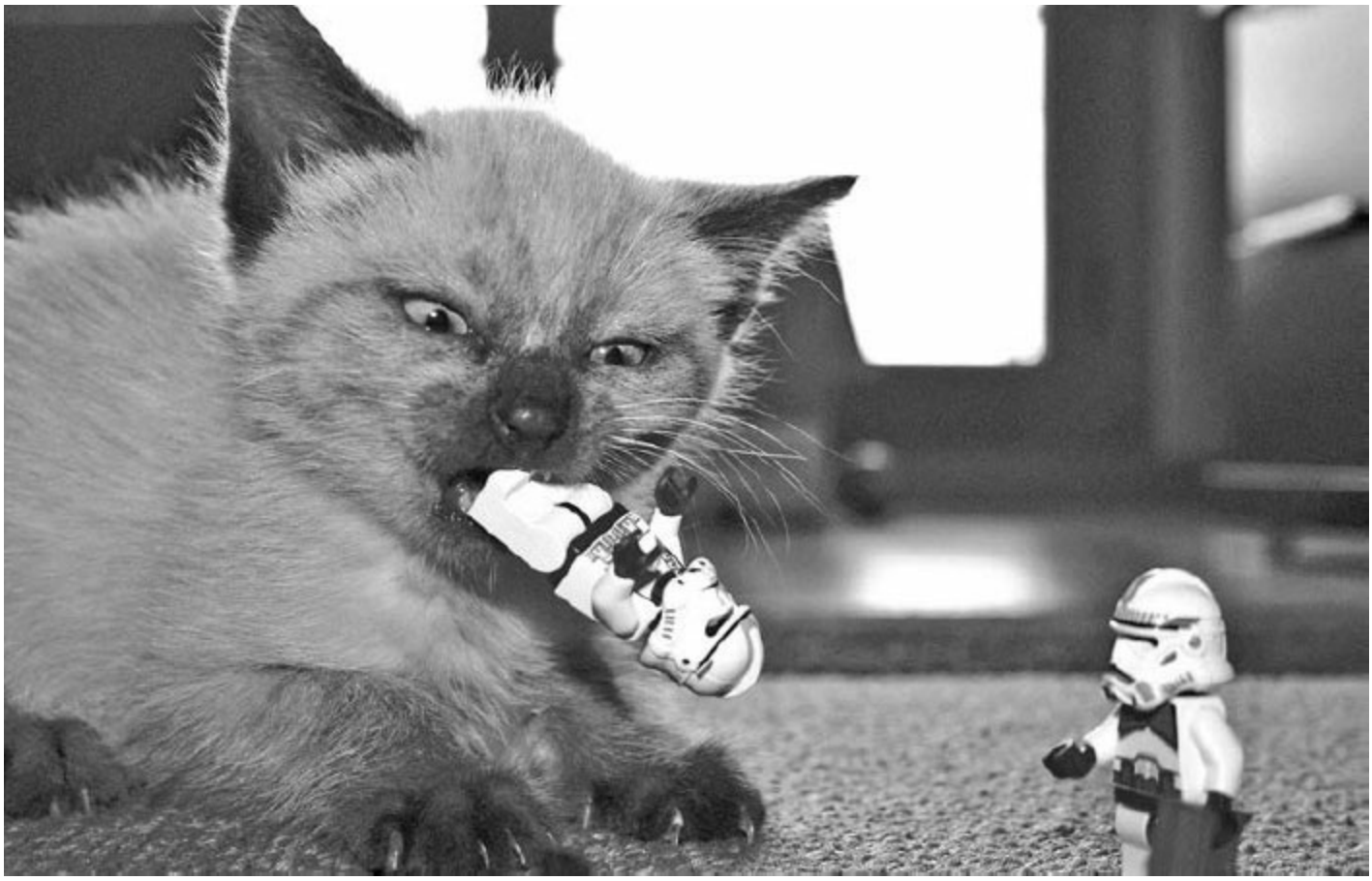
Поскольку вы терпеливо прочли главу «Сначала прочтите это», я не буду подробно останавливаться на самих историях и перейду к тому, как карты историй решают одну из самых больших проблем в разработке Agile. Если вам уже приходилось сталкиваться с составлением карт историй в проектах Agile, информации, содержащейся в этой главе, вполне хватит, чтобы приступить к делу.

# Слово на «А»

Если вы читаете эту книгу, то, наверное, знаете, что построение карт историй – способ работы с пользовательскими историями так, как это принято в процессах Agile. На текущий момент практически любая книга о разработке Agile так или иначе воспроизводит «Agile-манифест разработки программного обеспечения» (Agile Manifesto), написанный еще в 2001 году группой ребят, которым надоели громоздкие непродуктивные процессы разработки ПО, распространенные в то время. Я очень рад, что они его написали. Еще я рад тому, что долгосрочный эффект их работы затронул так много людей.

Мне жаль разочаровывать вас, но я не собираюсь приводить здесь этот манифест и рассуждать, почему он так важен. Думаю, вы и так это отлично знаете. Кстати, если вы не читали манифест, стоит сделать это.

На то место, которое занял бы в книге этот манифест, я лучше помещу смешное фото с котенком[5]. Почему? Да потому, что, как доказано, смешные фотографии котиков притягивают в Интернете больше внимания, чем какой бы то ни было манифест.



Но вы, наверное, недоумеваете: что общего у Agile-методов и этого котика? Да, в общем, ничего. Но гибкая разработка определенно связана с этой книгой, с историями, а также с развитием карт историй.

<Звучит ретромузыка>

Я работал в одном стартапе в Сан-Франциско в 2000 году, когда компания приняла на работу Кента Бека (это тот самый парень, который придумал экстремальное программирование и впервые изложил идею историй) в качестве консультанта по наладке процесса разработки ПО. Я совершаю этот исторический экскурс, чтобы показать, что мысль

о применении историй на самом деле очень стара. Если вы только начинаете работать с историями, то уже потеряли статус первооткрывателя, который могли бы иметь лет десять назад. Кент и другие пионеры экстремального программирования знали, что все известные в прошлом способы добиться точного соответствия требованиям не работали как следует. И у Кента возникла простая мысль, что все должны собраться вместе и рассказывать истории. Таким образом можно выработать единое понимание и вместе найти наилучшее решение.

# Истории надо рассказывать, а не писать

Когда я впервые услышал слово *«история»*, оно мне не понравилось. Я признаю это. То, что мы должны упрощать важные вещи, в которых нуждаются люди, называя их историями или сценариями, казалось мне неправильным. Но я тугодум, что мне уже было известно из обсуждения общего мнения. Мне потребовалось некоторое время, чтобы понять следующее.

> *Истории называются историями из-за способа их использования, а не потому, что их надо записывать.*

Еще до того, как я в полной мере осознал, почему истории получили такое название, я понял, что могу написать множество историй – в виде предложения или краткого заголовка, на карточках или стикерах. Я могу менять их местами и расставлять в соответствии с приоритетом, выбирая самые важные из них. Как только я решу, что одна из историй более важная, чем другая, можно начать ее обсуждение. Это было суперкруто! Почему я раньше не записывал ничего на карточках, чтобы удобно было организовать работу таким образом?

Проблема была в том, что одна карточка могла представлять собой нечто, внедрение чего в продукт заняло бы у разработчика несколько часов. Или дней. Или недель. Или целый месяц – кто знает? Точно не я – во всяком случае, пока мы не начнем говорить об этом.

Начав обсуждение истории во время работы над своим первым проектом Agile, я невольно вызвал неприятный спор, когда оказалось, что история слишком велика. Мне хотелось, чтобы он был реализован в течение следующей итерации, но разработчики, с которыми я разговаривал, доказали мне, что это невозможно. У меня было смутное ощущение, что я что-то делаю не так. Разработчики выделили небольшую часть, о внедрении которой на следующей итерации можно было говорить, но я был раздражен тем, что нам не удалось поговорить масштабно, рассмотрев все в целом. На самом деле мне хотелось знать, сколько времени займет разработка большой функциональности, которую хотелось получить в итоге. Я надеялся, что дискуссия поможет мне оценить это, но ничего не вышло.

# Изложение истории с начала до конца

В 2001 году я покинул команду, где тогда работал, и начал изменять привычный ход событий. Я и моя команда пытались выработать такой подход к работе с историями, который позволял бы сконцентрироваться на полной картине. Мы разрабатывали общее видение нашего продукта, вместе находили компромиссы и соглашения. У нас было множество карточек с заголовками историй, с помощью которых мы организовывали свои идеи, а также разбивали большую картину на мелкие части, которые отправляли в очередь на разработку. В 2004 году я написал первую статью о таком принципе работы, но не употреблял термин *«карты историй»* до 2007 года.

Оказывается, то, как вы что-либо называете, имеет большое значение. Только после того, как эта практика получила удачное имя, я смог оценить ее размах. На тот момент я думал, что сделал изобретение века, но затем стал встречать все больше людей, которые использовали очень похожие, если не точно такие же, методы. Оказалось, я обнаружил *паттерн*.

Впервые я услышал определение паттерна от своей подруги Линды Райзинг: когда вы кому-то рассказываете о своей грандиозной идее, а он отвечает, что тоже придумал и использует что-то подобное, это значит, что вы не изобрели нечто новое, а открыли паттерн.

Карты пользовательских историй – это паттерн. К нему прибегают разумные люди, чтобы составить представление о целом продукте или о целой функциональности. Они также используют этот метод, чтобы разбить большие истории на меньшие части. Но если вы не пришли к этому самостоятельно, не расстраивайтесь! Скорее всего, вам бы это удалось. Но чтение этой книги сэкономит недели и месяцы бесплодных поисков.

> *Карты историй нужны для того, чтобы разбивать большие истории на фрагменты по мере изложения.*

Сегодня компании одна за другой перенимают идею карт пользовательских историй. Моя подруга Мартина, работающая в SAP, в письме, написанном в сентябре 2013 года, сообщила: «…К настоящему моменту было проведено более 120 заседаний рабочих групп по картам пользовательских историй. Они так понравились многим представителям заказчиков! Это отлично зарекомендовавший себя рабочий подход в SAP».

Каждую неделю я слышу от многих рассказы о том, как карты историй помогли решить их рабочие проблемы. В настоящее время я узнаю от других людей куда больше нового об этом методе, чем когда-либо смог бы вынести из собственного опыта.

Изначальная идея историй была очень проста. Для начала она отвлекает наше внимание от обеспечения общего доступа к документам и переключает его на выработку общего мнения. Обычный метод работы с историями – составление их списка и его сортировка по важности. А затем необходимо приступить к обсуждению историй и их внедрению в программный продукт по одной за раз. На бумаге это выглядит вполне разумно, но в реальности может вызвать массу проблем.

# Гэри Левитт и плоский бэклог

Несколько лет назад я познакомился с Гэри Левиттом. Гэри был бизнесменом и как раз запускал новый веб-продукт. Этот продукт и сейчас работает, он называется Mad Mimi, что согласно задумке Гэри означало *«маркетинговый интерфейс музыкальной индустрии»*[6]. Гэри музыкант, и у него есть собственная группа. Он самостоятельно занимался административными делами, а кроме того, помогал управляться с ними и другим, работал в музыкальной студии и делал записи для клиентов.


В те дни, когда мы с Гэри познакомились, он получил заказ написать для шоу Опры Уинфри дюжину вступлений и концовок – небольших музыкальных фрагментов, которые используются для перехода к рекламе и т. п. Продюсеры телевизионных шоу покупают эти фрагменты так же, как редакторы газет покупают фотографии и рисунки. Словом, это нечто вроде музыкальных иллюстраций. У Гэри была идея создать довольно большое приложение, которое помогало бы людям вроде него и его знакомых сотрудничать между собой на таких проектах, а также делать другие вещи, которыми вынуждены заниматься музыканты и продюсеры, чтобы продвигать свою группу.

Гэри нужно было, чтобы кто-то разработал этот продукт, поэтому он сотрудничал с другим человеком, и этот другой человек работал в Agile. Он попросил Гэри написать список всего, что он хотел бы видеть в продукте, и расставить элементы списка по важности. Затем они бы обсудили верхние, самые важные элементы и начали создавать их один за другим. В Agile такой список функций обычно называется *бэклогом*, и, по мнению Гэри, было вполне разумно так и поступить: составить список и начать с самых важных вещей. Так он и сделал.

Гэри написал бэклог, и команда разработчиков начала создавать функции по одной. На момент нашей встречи деньги текли у Гэри как вода сквозь пальцы, потому что ему приходилось платить за каждый созданный фрагмент программного продукта. Продукт мало-помалу принимал требуемую форму, но Гэри видел, что до соответствия его видению еще далеко и деньги закончатся задолго до этого момента.

Я был знаком с человеком, работавшим с Гэри. Мой друг видел, что Гэри вне себя от

беспокойства, и очень хотел ему помочь. Кто-то из наших общих знакомых спросил меня, не хочу ли я поговорить с Гэри и помочь ему организовать его идеи. Мы познакомились и договорились встретиться в его офисе на Манхэттене.

# Говорите и пишите

Беседа началась. По мере того как Гэри рассказывал, я записывал ключевые слова его идей на карточках. У меня есть что-то вроде мантры, я люблю ее повторять, когда составляю карты пользовательских историй. Она звучит как «говорите и пишите», что означает: не дайте словам пропасть втуне. Запишите их на карточки, к которым сможете обратиться позднее. Вы удивитесь тому, как взгляд на пару слов, записанных на карточке, вызывает в памяти целое обсуждение. Мы можем двигать карточки по столу, выстраивая их в разном порядке. Мы начинаем использовать полезные слова вроде *«это»* и *«то»*, означающие записанное на карточках. Это экономит уйму времени. Я должен был заставить Гэри вытащить свои мысли наружу – это крайне важно для обеспечения общего мнения. Для него это было непривычно, но я без труда фиксировал идеи на карточках по мере того, как он их излагал.

> *Говорите и пишите: записывайте тезисы на карточках или стикерах, чтобы зафиксировать свои мысли по мере того, как рассказываете истории.*

Мы начали раскладывать карточки на письменном столе, но очень скоро место на нем закончилось. Во время моего визита Гэри занимался переездом в другой офис и большая часть мебели уже находилась в лофте в Нью-Йорке. Так что мы недолго думая перенесли разросшуюся карту пользовательских историй прямо на пол.

Вот как в конце дня выглядел пол.

***Мысль – запись – объяснение – место***

Работая с командой над созданием карты пользовательских историй или обсуждая какой угодно вопрос, не забывайте о простой вещи – визуализации,

которая поддержит дискуссию. Очень большой вред приносит то, что идеям позволяют просто исчезнуть сразу после того, как слушатели одобрительно покивали головой. Эти мысли никто не записывает, и их в дальнейшем невозможно восстановить. Потом, в ходе обсуждения, идеи всплывают снова и их приходится заново объяснять, потому что на самом деле их слушали невнимательно или просто забыли.

Выработайте привычку кратко записывать свою мысль перед тем, как ее озвучить.

1. Если вы используете карточки или стикеры, *запишите* на них несколько слов об идее, *как только она придет в голову*.

2. *Расскажите* о своей идее остальным, как только запишете ее на стикер или карточку. Объясняйте подробно и доходчиво. Рисуйте много картинок. Рассказывайте истории.

3. *Поместите* карточку или стикер в рабочее пространство, где каждый сможет увидеть ее, указать на нее, дополнить ее или переместить. Хочется надеяться, что там окажется много и других идей – ваших или чужих.

Я заметил за собой: когда максимально сосредоточиваюсь на том, что говорят другие, мне в голову приходит масса новых идей. Поначалу я пытался удерживать их все в памяти в ожидании момента, когда уместно будет вставить их в обсуждение, и порой приходилось ждать довольно долго. В какой-то момент я понял, что перестаю слушать человека, который сейчас говорит, поскольку мозг занят лишь тем, чтобы удержать свою прекрасную идею. Поэтому сейчас я просто быстро записываю мысль на стикере и откладываю его в сторонку, пока не наступит момент ее высказать. Каким-то образом запись идеи на бумаге высвобождает ресурсы мозга, и я могу сконцентрироваться на том, что слышу. А затем, когда нужно, взгляд на краткую запись на бумажке помогает быстро вспомнить идею и объяснить ее всем.

Я пришел к Гэри не за тем, чтобы сформулировать его требования. И первым, что мы обсудили, был вовсе не набор функций. Нам пришлось ненадолго вернуться и начать сначала.

# Оформите свою идею

Предметом первого обсуждения стала сама идея продукта. Мы говорили о бизнесе Гэри и его целях. *Зачем вы это создаете? Сформулируйте, какие преимущества даст этот продукт вам и другим людям, которые будут его использовать. Какие задачи он будет решать для этих людей и для вас?* Читая это, вы, наверное, догадались, что я думал о модели «до и после». Я пытался понять, какие результаты хочет получить Гэри, а не какую работу он планирует проделать.

Если я вешаю на доску два стикера, один под другим, то все предположат, что идея на верхнем стикере более важная. Не говоря ни слова и просто меняя положение записей, я обозначаю приоритеты. Попробуйте сделать это со списком целей. Расположите их в явно неверном порядке и покажите человеку, с которым работаете над достижением этих целей. Я проделал это упражнение с Гэри и его целями, и это помогло ему понять, что для него наиболее важно.

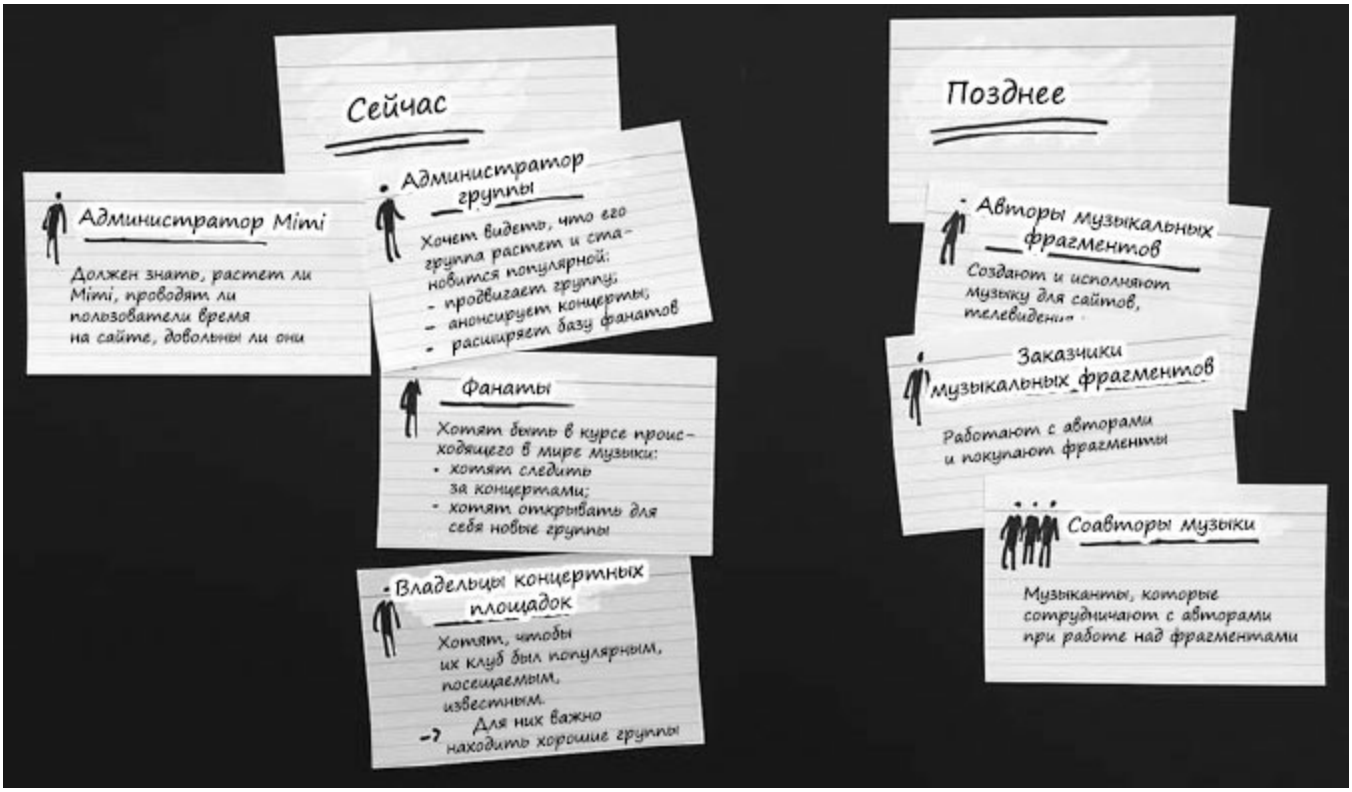
# Опишите своих пользователей и заказчиков

Мы с Гэри продолжили обсуждение. Следующей темой разговора стали заказчики, которые купят продукт, и пользователи, которые будут непосредственно с ним работать. Мы перечислили различные типы пользователей. Обсудили преимущества, которые они получат, задались вопросом, почему они будут использовать продукт, а также что, как мы предполагаем, будут с ним делать. Что продукт должен включать в себя с точки зрения пользователей? Ответы записали на целую груду бумажек. Карточки, на которых были указаны наиболее важные пользователи, сами оказались на верхних позициях. Забавно, что иногда это работает и таким образом – без явного намерения.


Еще до того, как мы погрузились в обсуждение деталей, я понял, что Гэри представлял себе очень большой продукт. Одна из неприятных особенностей разработки программного обеспечения заключается в том, что никогда не бывает достаточно денег для реализации всего желаемого. Отсюда вытекает, что цель, заключающаяся в разработке всего, *недостижима*. Поэтому нужно уменьшать количество того, что решено разработать. Поэтому первым делом я задал Гэри вопрос: «Если бы из всех этих пользователей и задач, которые они хотят решить, надо было выбрать только одну группу, кто бы в нее вошел?»

Гэри выбрал одну, и после этого мы наконец приступили к составлению историй.

### *Типы пользователей в Mad Mimi*




Вот разные пользователи Mad Mimi, которых описал Гэри. Уже простое перечисление их типов и краткие списки задач помогли понять, что тут многовато всего. Даже не начав обсуждать функциональность, мы решили отложить разработку для некоторых типов пользователей.

# Изложите свои истории

После этого я предложил: «Ну что ж, давай представим себе будущее. Предположим на минуту, что продукт выпущен и работает, и обсудим день жизни кого-то, кто использует его, а затем начнем составлять истории: сперва он делает одно, затем – другое, третье и т. д.»

И мы составили историю, расположив ее на полу справа налево. Иногда возвращались назад и вставляли какие-то элементы в середину, ведь карточки, на которых изложены идеи, очень легко перемещать.

Еще одна интересная вещь происходит при работе с карточками: если я кладу одну карточку слева, а другую справа – автоматически подразумевается, что вторая следует за первой. Для меня это просто чудо, впрочем, меня несложно удивить. Поразительно, как много мы можем поведать друг другу, не говоря ни слова.

*Перемещение карточек друг относительно друга позволяет вам общаться без слов.*

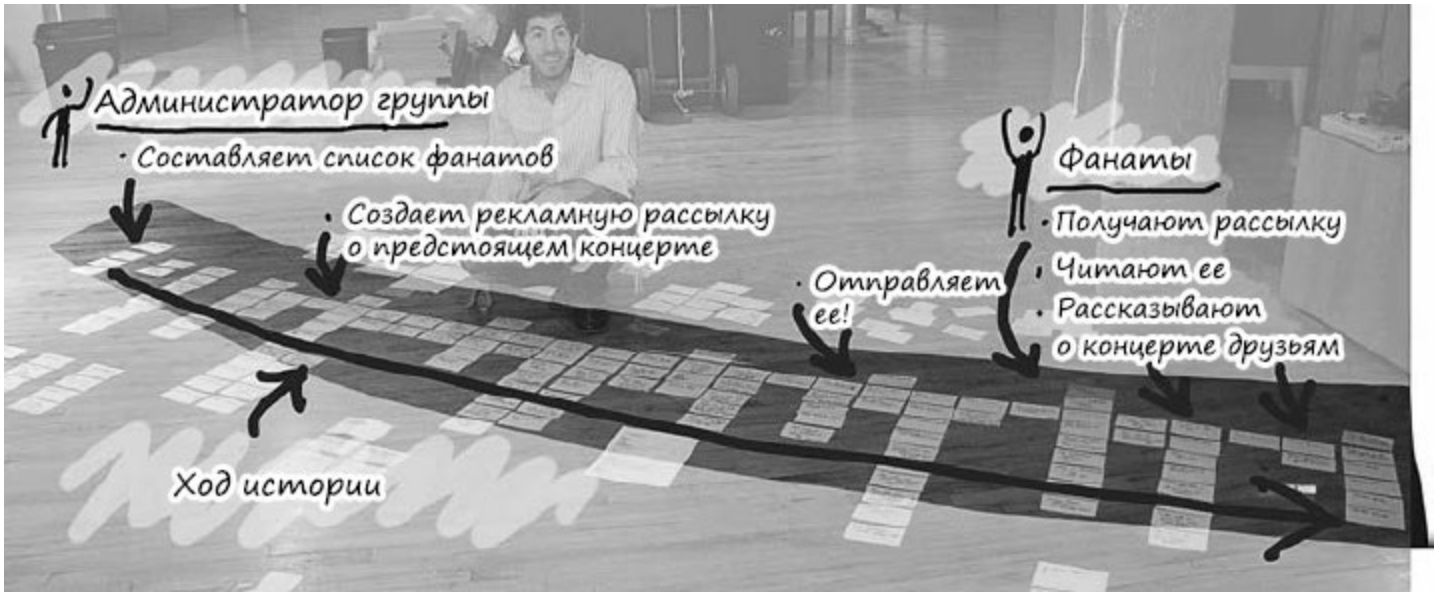
По мере того как мы говорим и пишем, а я фиксирую обсуждение, мы создаем нечто по-настоящему важное. Нет, не просто множество карточек на полу. А то, что по-настоящему важно, – *единое понимание*! Мы находимся, так сказать, на одной волне. У Гэри никогда не было такого взаимопонимания по поводу идеи своего продукта ни с кем другим, во всяком случае, не на таком уровне детализации. Даже он сам никогда не обдумывал продукт так углубленно. Он представлял лишь ключевые моменты, что-то вроде фрагментов основных сцен фильма, которые показывают в трейлерах.

Перед этим Гэри сделал то, о чем его просили: составил список историй на листе бумаги и рассматривал их по одной за раз. Обсуждение вращалось в основном вокруг деталей того, что необходимо разработать, а не вокруг общей картины. В результате в общей картине Гэри оказалось немало дыр. Вы и сами можете убедиться: независимо от того, насколько ясно вы представляете себе отдельную историю, проговаривание ее при составлении карты выявит пробелы в ней.

*Составление карты пользовательских историй поможет вам обнаружить пробелы в ней.*

Продвигаясь дальше, мы обнаружили, что истории редко предусматривают только одного пользователя. Гэри начал с истории администратора группы, который хотел продвинуть ее, для чего отправлял фанатам по электронной почте рекламную рассылку о предстоящем концерте. Затем мы перешли к обсуждению фанатки группы, которая читает рассылку и планирует посетить шоу.

После того как упомянули концерт группы, который состоится в определенном месте, на сцену вышел владелец площадки со своей историей – он хочет узнать подробности предложения. В этом месте наша история в буквальном смысле уперлась в стену, и пришлось продолжать ее следующим рядом, параллельным первому. Как видите, на фотографии карточки лежат в два ряда.




Иногда Гэри попадались в истории какие-то элементы, которые ему особенно нравились, и он начинал увлеченно описывать их в деталях. Одна карта под другой может означать приоритет, кроме того, часто это может быть просто *декомпозиция* – это такое модное слово, означающее разбиение чего-то большого на более мелкие составные части. По мере того как Гэри излагал детали, я записывал их на карточках и помещал под большим этапом истории. Например, Гэри очень увлеченно описывал создание рекламной рассылки, которую будут использовать администраторы групп, чтобы рекламировать свои концерты, и хотел учесть множество деталей.

Гэри жил в Нью-Йорке и при обсуждении листовок представлял себе все те крутые штуки, которые постоянно видел на стенах и фонарных столбах города. Одни выглядели так, словно кто-то склеил вместе текст и журнальные фотографии, а потом размножил полученное на ксероксе, но другие и в самом деле смотрелись эстетично и элегантно. Записав какое-то количество деталей, я предложил Гэри оставить их в покое, чтобы вернуться к ним позже, а сейчас продолжить обсуждение истории и продвинуть ее дальше. Утонуть в деталях очень легко, особенно если вам очень интересен предмет обсуждения. Но когда нужно получить общую картину, очень важно добраться до конца истории, прежде чем погружаться в мелочи. Еще одна мантра, которую я использую при работе с историями: «Мысли на милю вперед и на дюйм вглубь». Для тех, кому привычнее метрическая система: «На километр вперед и на сантиметр вглубь». Дойдите до конца истории, прежде чем тонуть в деталях.

> *Сосредоточьтесь на протяженности истории, прежде чем погружаться в ее глубину.*

В конце концов мы *добрались* до конца истории Гэри. Администратор группы успешно анонсировал концерт тысячам фанатов, которые распространили информацию среди своих друзей, и шоу имело огромный успех. К этому моменту мы оба представляли продукт кристально ясно.

«А сейчас, – сказал я, – давай вернемся назад, проработаем детали, а также рассмотрим некоторые альтернативы».

Просмотрев карту Гэри, вы можете заметить такие основные действия, как:

- регистрация;
- изменение набора сервисов;
- просмотр статистики моей группы;
- работа с календарем концертов;
- работа с аудиторией;
- публикация анонсов шоу;
- подписка на рассылку группы;
- просмотр предложений онлайн.

В основной части карты было много и других крупных функций, но и по этому набору вы можете представить, что нужно писать на карточках. Заметьте, легко понять, кто выполняет то или иное действие. Когда Гэри говорил о публикации анонса концерта, мы оба понимали, что это делает администратор группы. Когда я говорил о подписке на рассылку о концертах, Гэри знал, что речь идет о фанатах. Эти карточки находились в контексте обсуждения, и на них легко было сослаться.

«Публикация анонсов» оказалась очень большой функциональностью. Ее пришлось разбить на следующие шаги, расположившиеся слева направо под этой карточкой:

- запустить рекламу представления;
- просмотреть автоматически созданную Mimi рекламную рассылку;
- отредактировать рекламную рассылку;
- проверить созданную рассылку.

Обратите внимание на то, как ясно короткие глагольные фразы, написанные на карточках, говорят, что хочет сделать определенный тип пользователя. Такие формулировки облегчают воссоздание цельной последовательной истории: «Теперь администратор группы хочет опубликовать анонс о представлении. Чтобы сделать это, он должен запустить рекламу представления, затем просмотреть автоматически созданную Mimi рассылку, затем отредактировать ее, потом…» Достаточно просто вставить «а потом» в промежутке между двумя карточками, чтобы получить связную историю.

# Займемся вариантами и деталями

После того как мы разложили карту истории последовательно в длину от начала до конца, она начинает разрастаться в ширину. Карточка, представляющая собой один из больших шагов истории, становится началом колонки, где располагаются другие карточки – детали, на которые мы разбиваем большой шаг. Для этого мы останавливаемся на каждом шаге пользовательской истории и задаем вопросы.

- В чем особенности действий пользователя на данном этапе?
- Какие альтернативные вещи они могут сделать?
- Как можно сделать этот этап по-настоящему крутым?
- Как исправить ситуацию, если что-то пойдет не так?


До конца процесса мы добираемся, имея множество разных деталей. В результате получается целая история, повествующая об одном дне из жизни администратора группы, а также других людей, чье участие в процессе очень важно: фанатов и владельцев концертных площадок.

**Детали**

Если бы вы заглянули в историю шага, например, «Отредактировать рассылку», то увидели бы такие детали:

- загрузить изображение;
- прикрепить аудиофайл;
- разместить видео;
- добавить текст;
- изменить разметку;
- использовать ранее созданную рассылку как шаблон.

Как видите, для проработки всех подробностей даже этих меньших шагов требуется дополнительное долгое обсуждение. Но начать мы можем с простого их

перечисления.

Заметьте, что и на этих карточках использованы простые глагольные фразы, которые облегчают изложение историй. Мы можем составить из этих карточек историю, применяя связку «или он может», например, вот так: «Чтобы отредактировать рассылку, он может прикрепить аудиофайл, или добавить видео, или…» Очень удобно!

«А что сейчас? – спросил я у Гэри. – У нас есть и другие пользователи, и решить они хотят другие задачи. Хочешь обсудить их? Только, по-моему, если мы будем продолжать, то нам потребуется другая комната, побольше. И еще, Гэри: даже на разработку всего вот этого уже понадобится куча денег. Мы можем обсудить и все остальное, но у нас уже есть довольно много. По-моему, даже если запустить продукт только с этой частью, он будет классным».

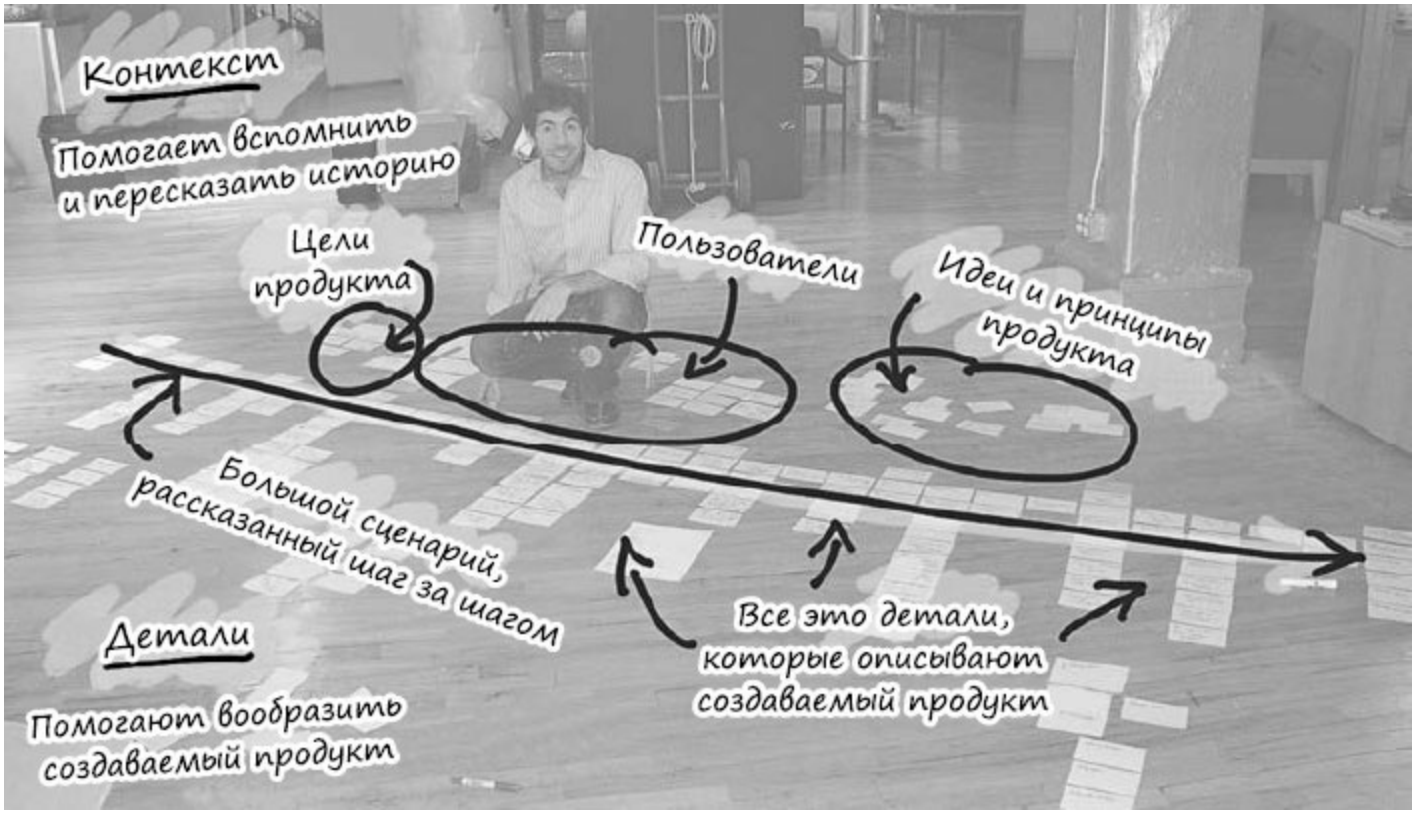
Гэри согласился: «Давай на этом остановимся».

В этой истории есть и грустный эпизод. Я знал, что разработчики, нанятые Гэри, уже сделали довольно много, поэтому спросил, соответствует ли хотя бы часть созданного ими программного обеспечения нашей карте.

«Почти ничего, – ответил Гэри, – потому что, когда я составил список функций и расставил их в порядке значимости, я был убежден, что мы должны начать с другого. Я думал о картине в целом, думал, что мне потребуются годы, чтобы реализовать ее. Но сейчас, после того, как мы все обсудили, я, конечно, начинал бы не с этого».

Самое важное в картах историй – это старое доброе обсуждение, а затем его изложение в форме карты. Обычно в центре внимания находится карта – последовательное, слева направо, изложение шагов, с помощью которых можно рассказать историю. Детали шагов располагаются сверху вниз. Но самое важное, что формирует продукт и дает больше всего контекста, обычно находится вне карты – это цели продукта, информация о заказчиках и пользователях. Если ваша карта располагается на стене, то будет очень хорошо разместить рядом с ней эскизы пользовательского интерфейса (UI) и другие заметки.

Всего за день совместной работы мы с Гэри пришли к одинаковому пониманию продукта, который он хотел создать. Правда, над нашими головами уже собирались тучи, и мы оба их прекрасно видели: каждая карточка заключала в себе множество деталей и требовала длительного обсуждения. А с точки зрения Гэри, все эти детали и обсуждение требовали денег на разработку программного обеспечения – денег, которых у него не было. Он получил один из самых фундаментальных уроков, которые дает разработка программ: времени на создание всего, что нужно, никогда не бывает достаточно.




Кроме того, Гэри предположил, что еще многие люди могли бы работать с его продуктом, если бы действительно хотели или были способны использовать его так, как он представлял. Но сейчас его беспокоило не это. В первую очередь нужно было хорошо поработать, чтобы сузить идею продукта до чего-то реалистичного и осязаемого.

В конце концов история Гэри завершилась благополучно. Но в следующей главе я расскажу о другой организации, которая тоже пыталась взять на себя больше, чем могла, а затем использовала карту историй, чтобы найти жизнеспособное решение.

## *Артджайл – творчество в искусстве встречается с творчеством в IT*

*Сиди (Клэр) Дойл, менеджер продуктов и тренер Agile (Assurity Ltd, Веллингтон, Новая Зеландия)*

**Предисловие**

The Learning Connexion (TLC) – колледж искусств в Веллингтоне, Новая Зеландия. Там изучают искусство и творчество. Программы TLC уникальны, так как они основаны на обучении действием, то есть изучении теории одновременно с практикой. Под руководством преподавателей студенты трудятся над работами на темы, которые сами выбрали для исследования.

TLC был типичной небольшой организацией, которая разрабатывала IT-системы по мере надобности, чтобы обеспечивать нужды, возникающие в определенный момент. Информация для студентов, например, располагалась в пяти разных местах и в каждом из них отличалась от хранящейся в других! TLC очень нужен был какой-то способ руководить студентами, которые будут здесь заниматься, поддерживающий кардинально иной принцип обучения, чем в большинстве образовательных учреждений.

У сотрудников TLC не было никакого опыта работы с IT-проектами. Все маленькие приложения, применяемые в колледже, когда-то были написаны чьими-то братьями, друзьями, соседями с использованием простых технологий вроде Microsoft Excel или Access. Единственное коммерческое приложение, используемое для составления официальных отчетов, повторно обрабатывало данные из других четырех источников.

Как бывшая студентка, я поддерживала контакты с сотрудниками колледжа, и поэтому, когда им понадобилась помощь, они обратились ко мне. К 2009 году я имела девятилетний опыт работы в IT, а в последние три года, с того самого момента, как впервые услышала о методологии Agile, мечтала сделать какой-нибудь проект, применив ее. И вот наконец звезды сошлись: правильное место, правильный проект и правильный момент, чтобы осуществить мечту!

**Проект «Феникс»**

Изначально были запланированы два заседания длительностью в полдня с участием наиболее значимых лиц из числа сотрудников колледжа. Я работала с большой разношерстной группой, поэтому первым делом хотела добиться одинакового понимания. Начала с обзора метода составления карт историй, а также выявления основных шагов в процессе руководства студентами колледжа.


До того как я показала членам команды картинку (основу карты историй), каждый из них имел собственное представление о том, что *они* делали, но, как сказала руководитель проекта Алиса, возможно, впервые все они получили ясную картину своего бизнес-процесса, а также связей всех шагов между собой.

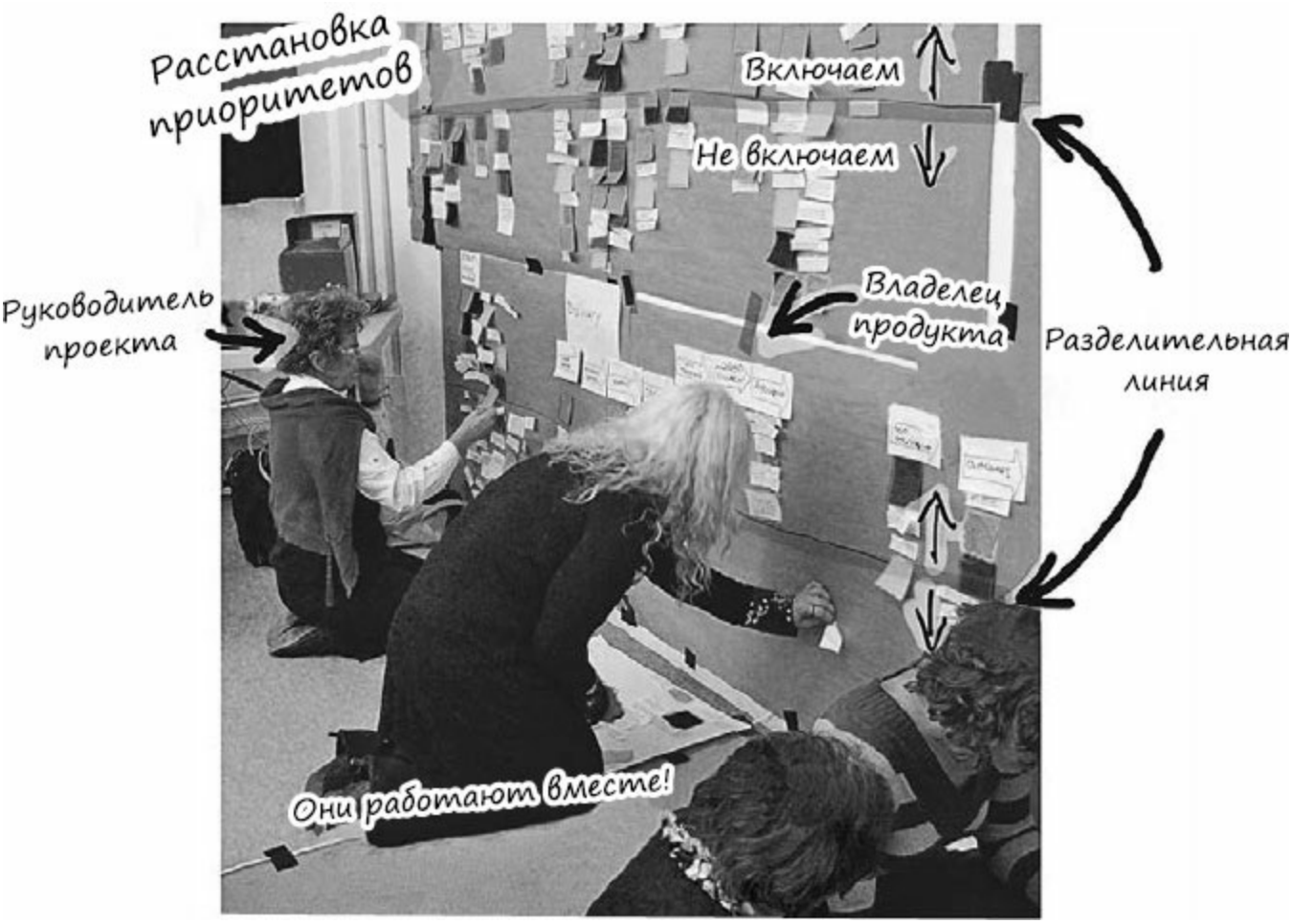
После этого, чтобы определить, чего люди хотят от системы, мы начали мозговой штурм. Содержание получилось довольно *масштабным*, а историй вышло очень *много*.


Нам повезло, что на заседании собрались творческие люди, привычные к методу позитивного исследования[7], поэтому обмен мыслями о том, что должна делать система, был для них обычным процессом.

Согласно диаграмме, основными этапами процесса были Запрос – > Допуск – > Вступительные экзамены – > Классы – > Итоговая работа – > Завершение – > Выпуск.


Затем, следуя правилам составления карт историй, мы прошлись по каждой секции, чтобы удостовериться в ее необходимости. После этого получили процесс следования студента по этому пути шаг за шагом. Одни люди с энтузиазмом включились в работу, когда поняли, где именно находится их место в общем процессе и почему они должны выполнять некоторые из своих обязанностей, а другие осознали, что остаются в стороне от некоторых шагов, которые были бы им

весьма полезны. Прохождение по карте историй, а также мое настоятельное требование располагать истории вертикально помогли выявить места, где они могли бы эффективно работать вместе, а также дублирующиеся шаги. До этого момента все члены команды имели смутное представление о том, что делает другой, но очень быстро выработали одинаковое понимание всего процесса и общий лексикон. Взять хотя бы пример шага «Классы»: он был переименован в «Обучение», так как не у всех студентов были классы.




Когда дело дошло до расстановки приоритетов, пришлось разделить все на категории «Обязательно», «Желательно» и «Неплохо бы». Здесь все очень просто: то, что «Обязательно нужно», располагалось выше разделительной линии, а все остальное – ниже. После того как мы обработали шаг «Вступительные экзамены», команда усвоила принцип и за остаток дня доделала все остальное. И моя помощь не понадобилась! Кроме того, они взяли на себя расстановку подзаголовков, которые были нужны, чтобы лучше описать процесс: «Сначала должно быть закончено все вот это, а затем – вот то». Таким образом, когда мы добрались до конца, они, действуя группой, создали обширную картину шагов, которые делает студент, начиная от вступительных экзаменов и заканчивая выпуском.

То, что планировалось как две четырехчасовые сессии, превратилось в три полных дня работы на заседаниях, куда люди приходили и уходили по мере необходимости (ведь им нужно было еще вести занятия и заниматься другой

работой). Такая гибкость означала, что почти все сотрудники колледжа прошли через комнату «Феникс» и внесли в общее дело свою лепту. Все утверждали, что такой процесс очень полезен для понимания общей картины, а также для того, чтобы пожелания каждого были учтены. Кроме того, было выявлено несколько пробелов и недоработок, а определить действительно важное оказалось очень легко. В конце концов мы получили ясную картину того, что должно было войти в первую версию программного обеспечения.

# Глава 2. Планируйте разработать меньше

*У вас никогда не будет достаточно людей, времени или денег, чтобы разработать все, что нужно. Никогда.*

Вообще говоря, мне не очень нравится изъясняться в таких категоричных терминах, как «никогда» или «всегда». Но приведенное утверждение – исключение: я не могу припомнить ни одной ситуации, где оно не было бы истинным, разве что у меня пробелы в памяти. Еще никто не приходил ко мне, чтобы поделиться радостью: «Нас попросили добавить вот эту новую функцию, а времени у нас оказалось намного больше, чем нужно!»

Но одна из самых классных вещей в методе карт историй заключается в том, что он дает вам и другим участникам процесса пространство для альтернативных решений и поисков способа получить наилучший результат за отведенное время.

Налейте себе чашечку кофе и усаживайтесь поудобнее. Настало время удивительных историй.

Я хочу вам рассказать о моих друзьях из Globo.com, крупнейшей медиакомпании Бразилии. Globo.com владеет радио– и телевизионными станциями, выпускает телефильмы и различные программы, а также газеты. Это медиамонстр в Бразилии и, кроме того, крупнейшая медиакомпания в мире, работающая на португальском языке.

Сотрудникам Globo.com лучше, чем кому-либо на планете, известно, что такое дедлайны, не подлежащие смещению. Например, компания хочет выпустить новую версию Fantasy Football Game – браузерной игры, которая дополняется и обновляется регулярно к чемпионату мира по футболу – самому громкому событию в футбольном мире. Если Globo.com не будет успевать к чемпионату, то перенос даты выхода игры их не спасет. Почему? Да потому что никто не изменит дату начала чемпионата. Globo.com планирует также выпустить разную функциональность и контент к Олимпийским играм, которые пройдут в Бразилии в 2016 году[8], и я могу гарантировать, что это будет сделано вовремя – иначе никак! Кроме того, в компании выпускается масса функциональности и контента к выходу огромного количества телевизионных программ и реалити-шоу. Ничто из этого не может быть перенесено, если Globo.com запаздывает с разработкой. Компания *всегда* должна закончить работу вовремя. И так как это очень важно для бизнеса, Globo.com отлично справляется. Не потому, что люди там работают быстрее, чем в любом другом месте, – быстро, конечно, но не *настолько* быстро. Секрет в том, что они умеют делать *меньше*.

# Составление карт помогает достичь единого мнения в больших группах

Посмотрите на рисунок.

Это всего лишь фрагмент огромной карты, созданной в результате совместной работы представителей восьми команд из трех разных групп Globo.com. Команды из «Спорта», «Новостей» и «Развлечений» составили вместе эту карту, чтобы продумать и распланировать работу, которая потребуется для преобразования, модернизации и обновления их основной системы управления контентом. С помощью этой системы компания управляет всеми сайтами новостей, спортивных событий, мыльных опер, публикует рекламу, ищет гостей реалити-шоу и делает еще многое. Такая огромная система должна быть способна обработать огромное количество видеофайлов и фотографий, счетчики и итоги голосования в реальном времени, экстренные новости и т. д. Система должна делать очень многое, и делать это без проблем.

В тот день, когда я пришел в офис Globo.com, они как раз составили эту карту и, судя по всему, вот-вот должны были угодить в *ловушку плоского бэклога*. Отдельные команды разработали собственные бэклоги с элементами, расставленными по степени важности. Было понятно, что предстоит огромный объем работы и что каждая группа зависит от остальных. Например, для хорошего новостного сайта требовалась работа не только группы новостей, но и других, которые создадут основные компоненты, нужные сайту для работы: фотографии, видео, данные, полученные в реальном времени, и еще многое.

Мы сели за стол, и я напомнил всем кое-что уже известное: «Я понимаю, что вы все принадлежите к разным командам, потому что концентрируетесь на разных областях, но тут у вас полноценная переработка *одной* системы управления контентом. Выпускать продукт вы будете вместе. И не сможете запланировать релиз, пока у вас не будет общего видения. Вам надо визуализировать все эти зависимости».

Они согласились со мной и быстро принялись за работу: нужно было преобразовать в

карту все индивидуальные бэклоги. За несколько часов карта была готова и размещена на стене с помощью стикеров, после чего история работы с системой управления контентом стала видна как на ладони.

Я не присутствовал при совместной работе членов команды над созданием карты, но, вернувшись позднее в этот же день, был поражен тем, как быстро им удалось это проделать. Они относились друг к другу доброжелательно и рассматривали мнение каждого. Оказалось, что удобнее всего организовать несколько комплексных бэклогов, которые вместе составляли одну связную историю. Таким образом, каждая команда могла видеть, как их работа встроена в общую картину.

*Карта историй продукта, с которым работают несколько команд, визуализирует зависимости.*

### *Анатомия большой карты*

Карта Globo.com – хороший пример того, на что похожа типичная карта после того, как вы обрисовали, связали и исследовали множество деталей.

**Карту организует каркас**

Вверху карты историй располагается каркас, который иногда имеет несколько уровней. Вы можете начать с базовой истории одного уровня, но по мере того, как она развивается и удлиняется, целесообразно ввести еще один или несколько уровней, чтобы в дальнейшем было проще подвести итог. Позднее я немного уточню терминологию, относящуюся к тому, что следует помещать на каждый уровень, но один старый друг напомнил мне, что не всегда нужно стремиться изъясняться точно. «Есть важные вещи, а есть мелочи», – сказал он. И это абсолютно верно.

Все это выглядит так, будто вы откопали скелет какого-то неведомого фантастического животного. Наверху вы размещаете этот скелет с множеством позвонков, находящихся на произвольном расстоянии друг от друга, и выпирающими ребрами деталей разной длины.


**Карта и ее значение для совместного выпуска продукта**

Эта карта была создана различными командами Globo.com. Над ней работали команды, ответственные за видеоматериалы, и команды, создающие ПО для внутреннего пользования, применяемое редакторами для создания контента и управления им. Были команды, ответственные за метаданные и связи между данными – всю эту семантическую разметку, в которую я, честно говоря, никогда не вникал. Были и те, кто занимался внешним представлением и обеспечивал хорошую картинку для конечных пользователей и/или потребителей. И конечно, много людей, выполняющих специфические функции, связанные со спортом, новостями или развлечениями.

Работа нескольких команд над картой была необходима потому, что ни одна команда не могла реализовать свою часть масштабной реорганизации без взаимодействия с остальными. Команды создали одну общую карту, потому что на протяжении этой работы они должны были думать как одно целое.

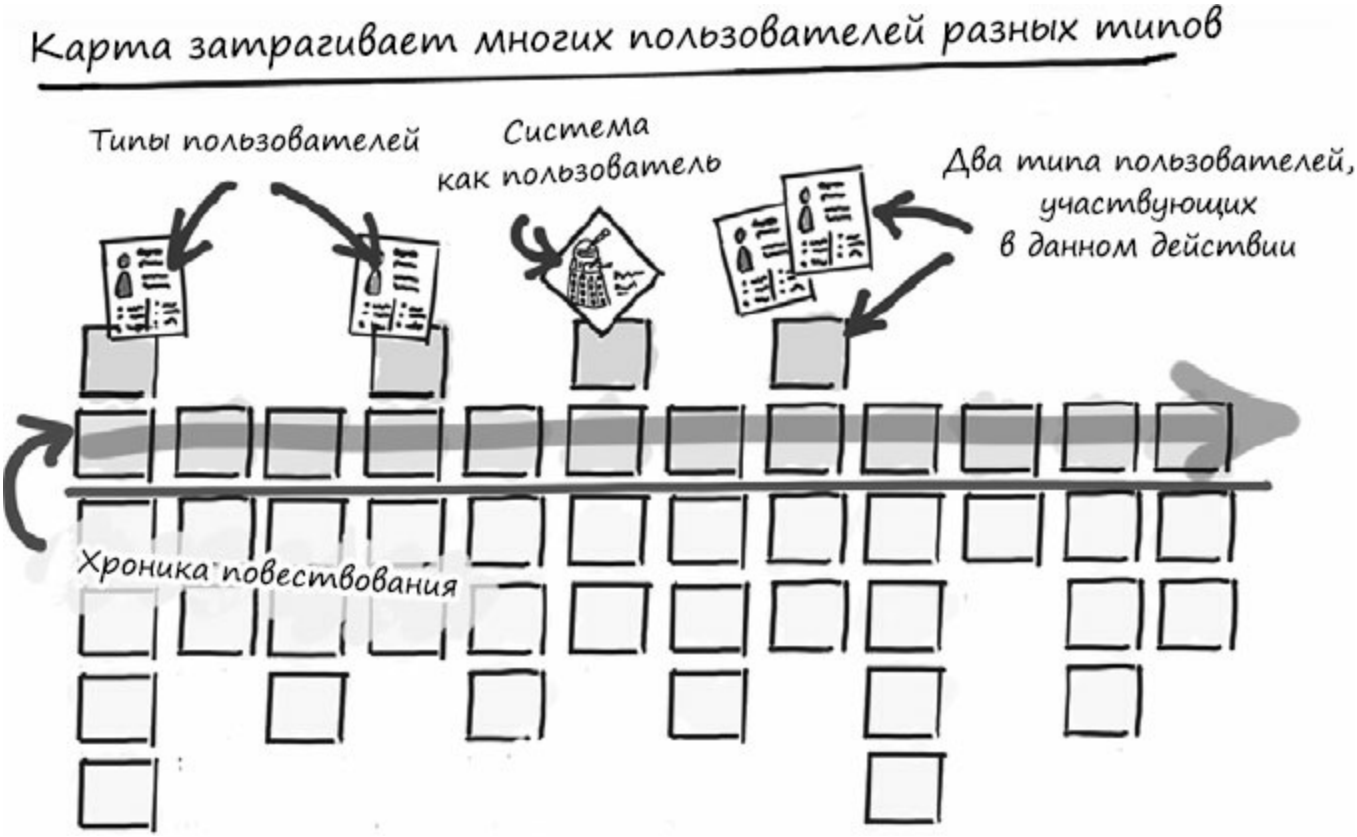
**Карта в изложении истории, включающей много пользователей и систем**

Карта началась с того, что слева обозначили названия специалистов, а также их действия по настройке расположения на экране текста, иллюстраций и видео. Затем в дело вступали другие люди, которые из этих материалов формировали страницы сериалов или новостных веб-сайтов. После них – редакторы, которые добавляли на страницы контент. Таким образом, весь каркас повествовал о том, как много людей в Globo.com занято конструированием контента на веб-сайте и управлением этим контентом.

Когда вы читаете каркас слева направо, он рассказывает историю обо всех людях, которые используют систему и производят какие-то действия для того, чтобы создать и настроить страницы и контент. Порядок слева направо я называю *хроникой повествования* – это такой академический способ обозначить

последовательность изложения истории. Разумеется, все упомянутые в ней люди делают свою работу одновременно, иногда процессы и вовсе протекают довольно беспорядочно – это понятно. Мы просто располагаем составляющие в порядке, удобном для формирования истории.

Хроника повествования такой большой системы затрагивает *множество различных пользователей и системных историй*. Мне удобно располагать наклейки или простые ярлычки с обозначением персонажей над каркасом, в результате чего легко понять, о ком конкретно мы говорим на определенном этапе истории. Можно очеловечить в том числе и процессы сервера или какие-либо компоненты, имеющиеся в системе. Например, мои друзья в SAP для элементов системы создают фантастических персонажей и используют изображения R2D2 или C3PO из «Звездных войн».



# Карты помогают вам распознать дыры в истории

Когда я разговариваю с людьми, которые уже составляли карты историй, они всегда признаются: «Каждый раз, составляя истории, мы находим дыры! Мы обнаруживаем, что *другая* команда должна была что-то сделать, но они, как оказалось, этого не знали. Мы обнаруживаем важные вещи и очень важные функции, которые забыли обсудить!»

После того как вы визуализировали весь продукт или какую-то функциональность, гораздо проще играть в игру «Что, если». Именно в этот момент мы начинаем задумываться: «Что будет, если что-то пойдет не так?» или «А что с этими, другими пользователями?». Играйте во «Что, если» с любыми сомнениями, которые у вас есть, и добавляйте наклейки в основную часть карты, где затем появятся идеи, которые нужно воплотить в программном обеспечении. В главе 1 Гэри играл во «Что, если», рассматривая варианты и альтернативы. Когда вы и другие команды проделаете это, то удивитесь тому, какое огромное количество проблем может обнаружиться при взаимодействии разных команд.

Один из часто приводимых аргументов, которые озвучивают, возражая против составления карт историй, заключается в том, что, когда команды садятся и составляют карты историй, на выходе получается слишком много всего. Но я убежден, что таким образом выявляются проблемы, с которыми так или иначе придется столкнуться позже, и в целом это скорее плюс, чем минус.

При традиционном подходе к разработке программного обеспечения ситуацию, когда что-либо новое обнаруживалось на одном из поздних этапов – уже после того, как было рассчитано время разработки и утверждена дата выхода, – мы называли *расширением границ проекта*. Я считаю, что в действительности расширяются границы не проекта, а понимания. То же самое происходит при построении карт историй – люди обнаруживают пробелы в своем понимании ситуации.

*Свои границы расширяет понимание, а не проект.*

# Всегда слишком много

Когда я покидал команду реорганизации системы управления контентом в Globo.com, все было прекрасно – все отлично знали, что им делать, и понимали задачу одинаково. Однако, вернувшись через несколько дней проведать ребят, я обнаружил их в растерянности – они поняли, насколько огромная работа им предстоит, а потребуется на нее, скорее всего, больше года. И, как уже догадался проницательный читатель, когда разработчик предполагает сделать что-то за год, на самом деле это означает два. Не потому, что он некомпетентен или не умеет планировать свое время, – просто-напросто оценить временные затраты на то, чего прежде не делал, очень нелегко. Ну и, конечно, все мы склонны смотреть на жизнь с оптимизмом.

«Задача слишком велика! Нужно сделать так много и это займет так много времени!» – говорили сотрудники Globo.com.

«И что, все-все это необходимо реализовать?» – спросил я, на что они, конечно, ответили:

«Разумеется! Это же все части одной большой системы управления контентом!»

«Но ведь обычно вы не делаете проекты так долго. Я знаю вашего генерального директора, он наверняка захочет увидеть результаты гораздо раньше, верно?»

«Так и есть, – подтвердили они. – Он хочет видеть что-то рабочее к выборам в Бразилии, то есть через несколько месяцев».

«А нужно ли, чтобы к этим выборам работало *все*?»

Ага! Стоило мне задать этот вопрос, как у них в головах словно зажглась лампочка. Конечно же, *все* им было не нужно. До этого момента они концентрировались на выявлении последовательностей и зависимостей, предполагая, что должны реализовать все задуманное. Это было возможно, вопрос только в сроках. Но теперь наконец фокус сместился на реалистичный результат.

> *Концентрируйтесь на результате, который вы надеетесь получить вне системы, чтобы принять решение о происходящем внутри системы.*

Сотрудники Globo.com сконцентрировались на выборах в Бразилии. Команды думали о великолепных *результатах* в виде восхищенных посетителей и рекламодателей, а также о компании-учредителе Globo, которая представит интерактивный контент о выборах в новом потрясающем стиле. Если бы им удалось этого достичь, это была бы победа.

Конечно, члены команды не в первый раз сталкивались с нереальными сроками разработки. Поэтому, подумав немного, они решили, что нужно запустить новостной сайт, а также несколько мелочей, которые нужны для его поддержки, так как именно туда посетители будут заходить, чтобы следить за ходом выборов. Концентрация на новостном сайте означала то, что работать начнут над инновационными методами отображения данных голосования в реальном времени, а также технологиями ускоренного выпуска новостей. И конечно, все это должно быть оформлено в новом, современном визуальном дизайне.

# Отделите релиз минимального работоспособного продукта

Мы взяли рулон синего скотча и наклеили на карте линии слева направо, чтобы сделать горизонтальные срезы. Затем закипела работа по перемещению карточек выше и ниже этих линий: вверх отправлялось то, что следовало включить в первый релиз, а вниз – то, что можно было отложить.

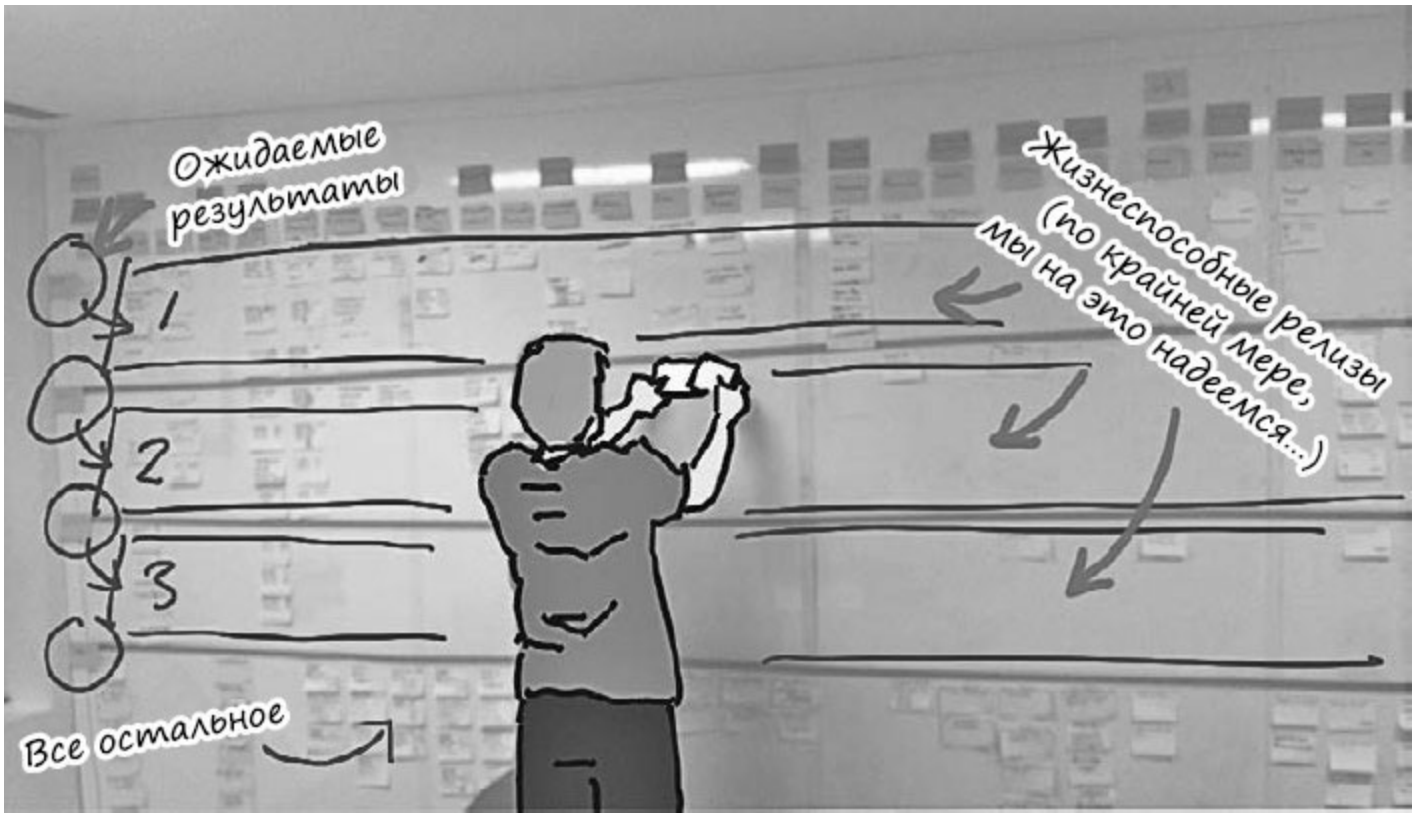

Принцип был такой: «Если мы запустимся к выборам в Бразилии, большинство бразильцев увидят этот элемент. Это масштабные изменения. Они затронут вот эти наши сайты, и мы будем смотреться неплохо. В этом релизе заключено все, что пользователи должны быть способны проделать после выпуска программного обеспечения, так что это произведет нужное впечатление».

> *Фокусируйтесь на результате – что пользователи увидят и будут способны сделать, когда система выйдет в свет, – и запланируйте выпуск только того, что обеспечит этот результат.*

# Составьте план релизов

Карта содержала инновации и улучшения, которые должны были оптимизировать все сетевые параметры Globo.com, но на реализацию их всех требовалось очень много времени. Возможность захватить лидерскую позицию среди СМИ, освещающих ход выборов, слишком дорого стоит, чтобы ее потерять. Сконцентрировавшись на этом, Globo.com смогла точно определить, что войдет в первый релиз.

После этого команды приступили к работе, помня о том, какие типы веб-параметров и других улучшений должны быть включены в следующие релизы. Они наклеили слева от каждого разделителя стикеры, описав на них в нескольких словах ожидания от релиза – желаемые результаты. Затем продолжили перемещать карточки выше и ниже соответствующих разделителей.




В итоге у них получилась целая релизная стратегия, руководствуясь которой можно было одолеть весь объем работы, необходимый для окончательной замены старой системы управления контентом на новую. Кроме того, в результате этого стали хорошо видны реальные выгоды от каждого релиза. Прочитав сверху вниз записи на левом краю карты, можно было получить список релизов, каждый с особым ожидаемым результатом. Его можно назвать *планом релизов*.

Обратите внимание: этот список не является набором функциональностей. Это перечень достижений в реальном мире – вы же помните, что ваша работа заключается не в создании программного обеспечения, а в изменении мира? Трудность заключается в выборе людей, чей мир вы должны изменить в этот раз, и определении того, как именно это сделать.

> *Секрет выбора приоритетов при разработке программ – концентрация на конкретных ожидаемых результатах.*

Обратное тоже верно: если вы не можете определиться, каких результатов ожидаете, каких конкретных выгод хотите достичь, расставить приоритеты практически невозможно.

## Не пытайтесь расставить приоритеты в функциональностях вместо результатов

Заметим: команда Globo.com начинала с огромной задачи обновления всей системы управления контентом. Обновление системы – результат, которого хотела достичь команда. Это сулило множество преимуществ. Очень важно было разбить этот поистине монстрообразный результат на более мелкие, больше подходящие для фокусирования.

Запомните: результаты, которые мы кладем в основу расстановки приоритетов, – это конкретные изменения в поведении конкретных людей, вовлеченных в конкретные действия и процессы. Сконцентрировавшись на выборах в Бразилии, Globo.com стала ориентироваться лишь на людей, которые следят за новостями – особенно за обновляющимися данными голосования. Но в результате этого за бортом остались зрители мыльных опер, спортивные болельщики и множество других типов пользователей. Этим людям придется довольствоваться старой версией сайта несколько дольше. Помните, что невозможно удовлетворить всех и каждого одновременно.

# Волшебство? Так и есть

Конечно, меня немного переполняют эмоции, но разделение действительно одна из самых крутых вещей в организации идей для ПО с помощью карты историй.

Много раз и я сам, и команды, с которыми работал, фиксировали в карте мысли, возникшие по поводу замечательного продукта, а потом ужасались тому, какой огромный объем работы нужен, чтобы воплотить все их в жизнь. Критически важным кажется *все*! Но затем мы немного остываем и задумываемся о конкретных людях, которые пользуются нашим продуктом, а также о том, что им нужно для достижения успеха. Эти мысли мы излагаем в паре предложений. А потом стоит лишь отсечь все лишнее – и все поражены тем, как на самом деле мало нужно, чтобы выпустить жизнеспособное решение! Это *волшебство*.

Гэри из главы 1 как раз и проделал нечто подобное. Он изначально сузил область своего внимания до менеджера группы, фанатов, а также администратора Mimi – нужно ведь, чтобы кто-то поддерживал сайт. Владельцев концертных площадок и авторов музыкальных фрагментов ему пришлось оставить на потом. В итоге, остановившись всего на нескольких типах пользователей и основном процессе рекламного продвижения, Гэри создал потрясающую платформу для рассылок по электронной почте. Может быть, вы сами являетесь пользователем Mimi и знаете, как она работает.

Когда мы излагаем свои мысли на большой красивой карте, все эти шаги сделать проще. С ее помощью множество людей могут работать вместе, чтобы прийти к общему решению.

### *Поиск маленького жизнеспособного релиза*

*Крис Шинкл, SEP*

FORUM Credit Union – один из самых больших и технически развитых кредитных союзов в стране. Хотя культура разработки ПО в компании отличается и креативностью, и компетентностью, ее сотрудники обратились в SEP за помощью в создании новой банковской онлайн-системы, которая должна была конкурировать с типовыми коммерческими решениями. Целью было добавление возможности пользоваться мобильным банкингом, СМС-банкингом, а также лично управлять своими финансами.

SEP начала работу с двухдневной исследовательской сессии составления карты историй, включающей результаты, персонажей и сами истории. В результате хорошо организованного обсуждения удалось получить большой набор предположений по поводу необходимой функциональности, отсортированный по приоритетности, однако для расстановки по приоритетности историй результатов и персонажей оказалось недостаточно. Но к концу второго дня карта историй занимала две стены в 1000-футовом[9] кабинете отдела разработки!


После того как карта была составлена, SEP обучила сотрудников FORUM составлять простую модель расстановки приоритетов.

• *Конкурентное преимущество* – преимущество, которое отличает их от конкурентов.

• *Спойлер* – функциональность, которая стремится перекрыть чужое конкурентное преимущество.

• *Фактор снижения затрат* – функциональность, введенная для снижения затрат организации.

• *Минимальный пакет* – минимальный набор функций, необходимый для успешного конкурирования на рынке.

SEP обозначила категорию каждой истории стикерами разных цветов, что вызвало любопытные обсуждения. Как оказалось, то, что одни участники считали конкурентным преимуществом, с точки зрения других являлось минимальным пакетом. Еще интереснее было то, что об этом, очевидно, говорилось впервые! Карта историй с моделью расстановки приоритетов заставила людей обсудить вопросы, которые раньше никогда не возникали! В результате этой работы члены команды наконец пришли к общему мнению относительно приоритетов.


После сортировки историй SEP использовала систему голосования, чтобы помочь участникам свести воедино их замыслы и результаты обсуждений, а затем прийти к четкому набору функций, нацеленному на результат. К удивлению всех присутствующих, оказалось, что некоторые истории можно отложить на неопределенный срок или исключить вовсе. Только по самым общим подсчетам это сэкономило компании несколько сотен тысяч долларов еще до написания первых строк кода.

Дуг Тру, генеральный директор FORUM, о значении карт историй для запуска проекта сказал так: «Когда мы начали работу по проекту с использованием карт историй и персонажей, я чувствовал некоторый скепсис. Честно говоря, мне казалось нецелесообразным тратить на это время. Но на следующий день польза от потраченного времени стала очевидной. Сейчас я уже не могу представить себе запуск такого большого и сильно затрагивающего интересы пользователей проекта без этого предварительного процесса».

# Почему никто не любит МЖП

Термин *«минимально жизнеспособный продукт»* (МЖП) давно используется в индустрии программного обеспечения. Хотя первое его употребление приписывают Фрэнку Робинсону, сейчас более популярны определения Эрика Райса или Стива Бланка. Хотя множество умных людей пытались объяснить, что такое МЖП, многие, включая меня, так и не могут толком разобраться в этом понятии. В каждой организации, где мне доводилось слышать этот термин, под ним подразумевалось что-то свое. Бывало даже, что люди, работавшие в одной организации, в разговорах друг с другом подразумевали под МЖП разные вещи.

Как и большинство слов в словаре, этот термин имеет несколько значений. Я приведу три примера определений: одно плохое и два хороших.

Начнем с плохого.

> *Минимально жизнеспособный продукт – это не самый ужасный продукт, который вы потенциально способны выпустить.*

Но МЖП – это *не* продукт, с которым могут работать ваши пользователи, – разве что в самых простых обстоятельствах, да и то при условии высокого болевого порога. Тем не менее я регулярно вижу организации, пытающиеся довести до ума ужасные продуктовые решения, чтобы «кто-нибудь мог использовать продукт», хотя каждому участнику этой работы очевидно, что он сам не стал бы с ним мучиться.

Если бы Globo.com воспользовалась этим определением для своего первого релиза, то, скорее всего, получила бы плохой результат. Проделанная работа не впечатлила бы никого. Компания потерпела бы убытки, и для нее гораздо лучше было бы вообще ничего не выпускать.

Когда говорят о живом организме, под словом «жизнеспособный» обычно понимают, что организм может выжить в нашем мире. К программному обеспечению это тоже относится.

> *Минимально жизнеспособный продукт – это продукт минимального объема, при выпуске которого успешно достигаются желаемые результаты.*

Это определение мне нравится больше всего. *«Минимальный»* – субъективная характеристика, поэтому вам нужен конкретный субъект для определения минимума и этот субъект не вы. Точно определите, кто ваши заказчики и пользователи и что им нужно для решения их задач. Что им *минимально необходимо*? Я клянусь вам, это сильно облегчит обсуждение. Есть, правда, и альтернатива – когда решение принимают вышестоящие лица, но это намного хуже.

В настоящее время я предпочитаю термин *«минимально жизнеспособное решение»*. В большинстве случаев, когда я сотрудничаю с организациями, мы не работаем над полностью новыми продуктами. Чаще всего это какие-то новые функции или возможности, улучшения уже имеющихся функций. Поэтому термин *«решение»* здесь более уместен. Поэтому давайте пересмотрим мое определение.

*Минимально жизнеспособное решение – это самое простое решение, при воплощении которого в жизнь успешно достигаются желаемые результаты.*

А сейчас мы приступим к трудной части…

Все это только гипотезы.

Когда мы отделяем какой-то набор функциональностей программы и называем это минимально жизнеспособным решением, мы не можем точно знать, что это так.

Проблема с оценкой результатов заключается в том, что увидеть их вы можете лишь тогда, когда все уже сделано. Определяя, что войдет в релиз, вам приходится предполагать, что будет. Придется представить себе заказчиков, которые купят ваш продукт, и пользователей, которые выберут именно его, если смогут, а кроме того, учесть, что вы реально успеете сделать за имеющееся время. Вам придется предполагать, сколько нужно сделать, чтобы пользователи были довольны. Очень много предположений.

Это и в самом деле сложно, ведь если вы запланируете меньше необходимого, то есть меньше минимума, то у вас ничего хорошего не выйдет. А если больше (к чему, как правило, склонны люди в попытке перестраховаться), то потратите слишком много денег и велик будет риск вообще не справиться с задачей. И самый плохой вариант – может оказаться, что вы изначально выбрали неверное направление и поэтому не сделали ровным счетом ничего полезного.

Поэтому неудивительно, что определение, где фигурирует «худший продукт, который вы можете выпустить», до сих пор так распространено.

# Новый МЖП вообще не продукт!

Некоторые из вас, вероятно, чувствовали небольшое раздражение, читая эти две главы. Возможно, вы сейчас думаете: «Джефф упускает из виду кучу важных вещей!» И, возможно, вы правы. Самые важные моменты, которые вы обсуждаете во время составления историй и карт историй, чаще всего такие.

• Где у нас самые большие и рискованные предположения? Где неопределенность?

• Что и где мы можем узнать, чтобы заменить предположения реальной информацией?

Таким образом, мы приходим к следующему определению МЖП, ставшему популярным благодаря книге Эрика Райса «Стартап по методологии Lean» (The Lean Startup, издательство Crown Business). Как часто случается, Эрику пришлось усвоить то, о чем мы сейчас говорили, на собственном горьком опыте. Эрик работал на компанию, выпустившую жизнеспособный, как там думали, продукт, но оказалось, что это не так. Эрик поступил разумно, устремив свое внимание на изучение фактов – проверку всех тех предположений, которые сделали в компании перед выпуском первого релиза МЖП. Он заключил, что мы должны ставить небольшие эксперименты, делать прототипы и проверять наши гипотезы о том, что минимально и жизнеспособно. Если вы последуете примеру Эрика, что я рекомендую сделать, ваш первый продукт будет, по сути, экспериментом, а также следующий и так до тех пор, пока не убедитесь, что сделали нечто действительно стоящее.

> *Минимально жизнеспособный продукт – это также нечто наименьшее из того, что вы можете сделать или создать, чтобы доказать или опровергнуть предположение.*

Поскольку ребята из Globo.com смогли запланировать разработать меньше, они не обманывали сами себя. Они осознавали, что предстоит еще много работы, чтобы доказать правоту своих предположений, так что теперь и они, и все остальные должны закладывать в план много исследовательской работы. С этого мы и начнем разговор в следующей главе.

# Глава 3. План быстрых исследований

Это мой друг Эрик, стоящий перед своим бэклогом и доской задач в кабинете своей команды. Эрик – владелец продукта; он и его команда тяжело работают, чтобы выпустить успешный продукт, но пока не добились этого. Впрочем, это не слишком беспокоит Эрика, ведь у него есть стратегия достижения успеха. И она обязательно сработает.


Эрик работает на компанию под названием Liquidnet – огромную торговую сеть для крупных инвестиционных организаций. Задолго до того, как Эрик сфотографировался перед своей доской, кто-то в этой компании обнаружил группу заказчиков, которых Liquidnet могла бы обслуживать лучше, а также выдвинул пару предложений, как это сделать. Эрик был в команде, которая начала работу с этими идеями. Это работа владельцев продукта. Не думайте, что эти люди всегда работают только со своими собственными идеями, это не так. Одна из самых больших сложностей в работе владельца продукта – принять чье-то предложение и продвигать его до воплощения в жизнь или, напротив, доказать его несостоятельность. Лучшие владельцы продуктов вроде Эрика включают в эту работу всю команду.

# Начните с обсуждения перспектив

Эрик начал свою работу вовсе не с составления бэклога пользовательских историй. Оттолкнувшись от отличной идеи, пришедшей кому-то в голову, он рассмотрел ее как благоприятную возможность для всей компании, чем она на самом деле и являлась. Он говорил об идее с руководством компании, обсуждалось следующее.

*В чем* состоит идея?

*Кто эти заказчики?* Что за компании, как мы предполагаем, будут покупать продукт?

*Кто наши пользователи?* Какие типы людей в этих компаниях, мы считаем, будут использовать продукт и для чего?

*Почему он им нужен?* Какие задачи заказчиков и пользователей решит продукт и почему они не могут решить их сегодня? Какие преимущества получат эти люди, купив продукт и начав его использовать?

*Почему мы его создаем?* Если мы создадим этот продукт и он будет успешным, что это нам даст?

Эрику необходимо было, чтобы он и другие сотрудники его организации одинаково понимали проблему, прежде чем взять на себя ответственность за реализацию этой возможности. Было очевидно, что в ближайшие месяцы ему придется изложить историю этого продукта множество раз, поэтому стоило заранее изучить все как следует.

> *Первое обсуждение истории нужно для формирования структуры потенциала.*

# Подтвердите наличие проблемы

Эрик, конечно, доверял интуиции своего руководства, но твердо знал: любая грандиозная идея – это только гипотеза. А гипотезу об успешности идеи можно проверить только одним способом – своими глазами увидеть ее успешную реализацию.

Поэтому первым делом Эрик поговорил с заказчиками и пользователями, чтобы узнать о них побольше. Так он убедился, что у некоторых заказчиков и в самом деле были проблемы и они очень заинтересованы в их решении. Эрик поговорил и с людьми, которые должны были работать с продуктом. Они ничего о нем не знали, поэтому могли лишь поделиться с существующим рабочим окружением проблемами, которые должен был решить новый продукт.

> *Убедитесь, что проблема, которую вы собираетесь решить, на самом деле существует.*

За время общения с пользователями и заказчиками Эрик наметил группу людей, которых считал хорошими кандидатами для пользовательского тестирования новой программы. Некоторые компании называют таких людей *партнерами по разработке на стороне заказчика*. Обратите внимание на эту деталь, так как позже она встретится в этой истории.

Фактически на этой стадии действовал не только Эрик. Он работал с небольшой командой других людей, которые много обсуждали эту проблему со своими заказчиками и в конце концов пришли к тому, что проблему решить непросто – имелись и другие проблемы, которые нужно было решить в первую очередь. Очень важно понять следующее: чем больше они узнавали, тем сильнее менялась оригинальная идея – вплоть до неузнаваемости. Счастье, что они не начали с разбегу воплощать в жизнь изначальный замысел! Эта работа оказалась бы не нужной ни заказчикам, ни их организации.

К настоящему моменту команда Эрика, поговорив с заказчиками, сформулировала и была готова реализовать решение, которое было бы востребовано у пользователей, а значит, принесло бы выгоду компании-нанимателю. Вот теперь команда могла бы пойти ва-банк и поставить на кон все. Наконец можно было приступить к созданию бэклога историй, описывающих их решение, и набрать команду специалистов для воплощения его в жизнь. Кроме того, будучи разумными людьми, они разработали карту историй, чтобы проработать детали большой идеи в виде набора конкретных функций для реализации. Но поскольку они *и в самом деле* разумные люди, последнее, что они собираются делать на этом этапе, – это начать программировать.

# Прототипируйте, чтобы исследовать

Эрик начал работать как владелец продукта. Сначала он рассмотрел свое решение как набор простых несложных историй – *пользовательских историй*. Затем перешел к визуализации идеи, превратив ее в простые наброски интерфейса. А после этого создал высокоточный прототип. Это еще не было работающее программное обеспечение – всего лишь простой электронный прототип, созданный с помощью несложной программы наподобие Axure или даже PowerPoint.

Для Эрика все это было исследовательскими этапами, которые помогли ему детально рассмотреть свое решение. Разумеется, ему хотелось скорее выдать продукт пользователям, чтобы узнать их мнение, но он знал: нужно быть твердо уверенным в том, что продукт решит проблемы пользователей, прежде чем выпускать его.

> *Создавайте прототипы и эскизы, чтобы детально представлять себе свое решение.*

Между прочим, я скрыл от вас важную деталь: Эрик был дизайнером взаимодействия. Для представителей этой профессии вполне естественно проводить много времени в общении с заказчиками и пользователями, а также создавать простые прототипы. Но для нового продукта он выступил также в роли владельца – человека, который несет полную ответственность за успех всей работы. Другие владельцы продукта в компании, где работал Эрик, не имели навыков дизайна, но они работали в тесной связке с дизайнерами, которые помогали им проводить интервью с пользователями и визуализировать решения.

Я не присутствовал, когда Эрик показывал свои прототипы пользователям, и не знаю точно, как прошло тестирование. Но я бывал в таких ситуациях множество раз и всегда поражался тому, как много я могу почерпнуть у людей, которые будут использовать мое решение на практике. Все, что я могу вам сказать: будьте готовы к неприятным сюрпризам и плохим новостям, хотя на самом деле плохим результатам надо радоваться, ведь вы могли получить их через несколько месяцев, потратив время и деньги на разработку программного продукта. Вот тогда это и в самом деле беда, а пока изменения сделать легко и стоит это недорого. Именно так Эрик и поступил.

> *Разрабатывайте прототипы и проверяйте с реальными пользователями, насколько полезно и пригодно к использованию ваше решение*.

Эрик повторял процедуру внесения изменений и тестирования несколько раз, пока не убедился, что наконец получил по-настоящему хорошее решение. Теперь можно было приступить к заполнению бэклога и усадить за работу команду программистов, чтобы они реализовали обновленный прототип в настоящее, работающее программное обеспечение. Но Эрик пока не собирался этого делать. По крайней мере, конкретно этого. Он пока не был готов принять этот вызов.

## Критически относитесь к тому, что люди говорят о своих желаниях

Эрик прототипировал то, что считал хорошим, жизнеспособным решением. Но он не был полностью уверен в том, что оно минимально, ведь он продемонстрировал людям множество крутых идей. А когда вы показываете людям такие идеи, неудивительно, что они в восторге. Но Эрик знал, что его работа состоит в уменьшении объема разработки, притом что люди должны остаться довольными. Что же можно отбросить, сохранив при этом жизнеспособность решения?

Была и другая причина для беспокойства. Эрик хорошо знал: когда люди утверждают, что им нравится идея и они охотно работали бы с продуктом, они тоже только предполагают.

Вспомните, как вы сами что-нибудь покупали. Вы, наверное, рассматривали этот продукт. Консультант магазина, возможно, демонстрировал вам какие-то его впечатляющие функции. Вы, может быть, даже опробовали его самостоятельно и представляли, как будете пользоваться этим продуктом и как это будет здорово. Но когда вы купили продукт и начали им пользоваться, оказалось, что все крутые функции не так важны, как вы думали, а важно то, о чем вы не задумывались вовсе. А может быть, оказалось, что вам вообще не так уж нужна была эта покупка. Впрочем, может быть, так происходит только со мной – в моем гараже целая куча хлама, который мне лучше было бы никогда не покупать.

Вернемся к Эрику. Он знал, что его заказчики и пользователи представляли себе продукт, который очень хотели бы использовать, и это давало ему некоторую уверенность. Но по-настоящему убедиться в своей правоте он смог бы лишь после того, как эти люди реально *пользовались* бы этим продуктом каждый день. Именно к такому результату он стремился, и только он дал бы компании то, чего она хотела достичь. Чтобы проверить это, нужно было нечто большее, чем прототип.

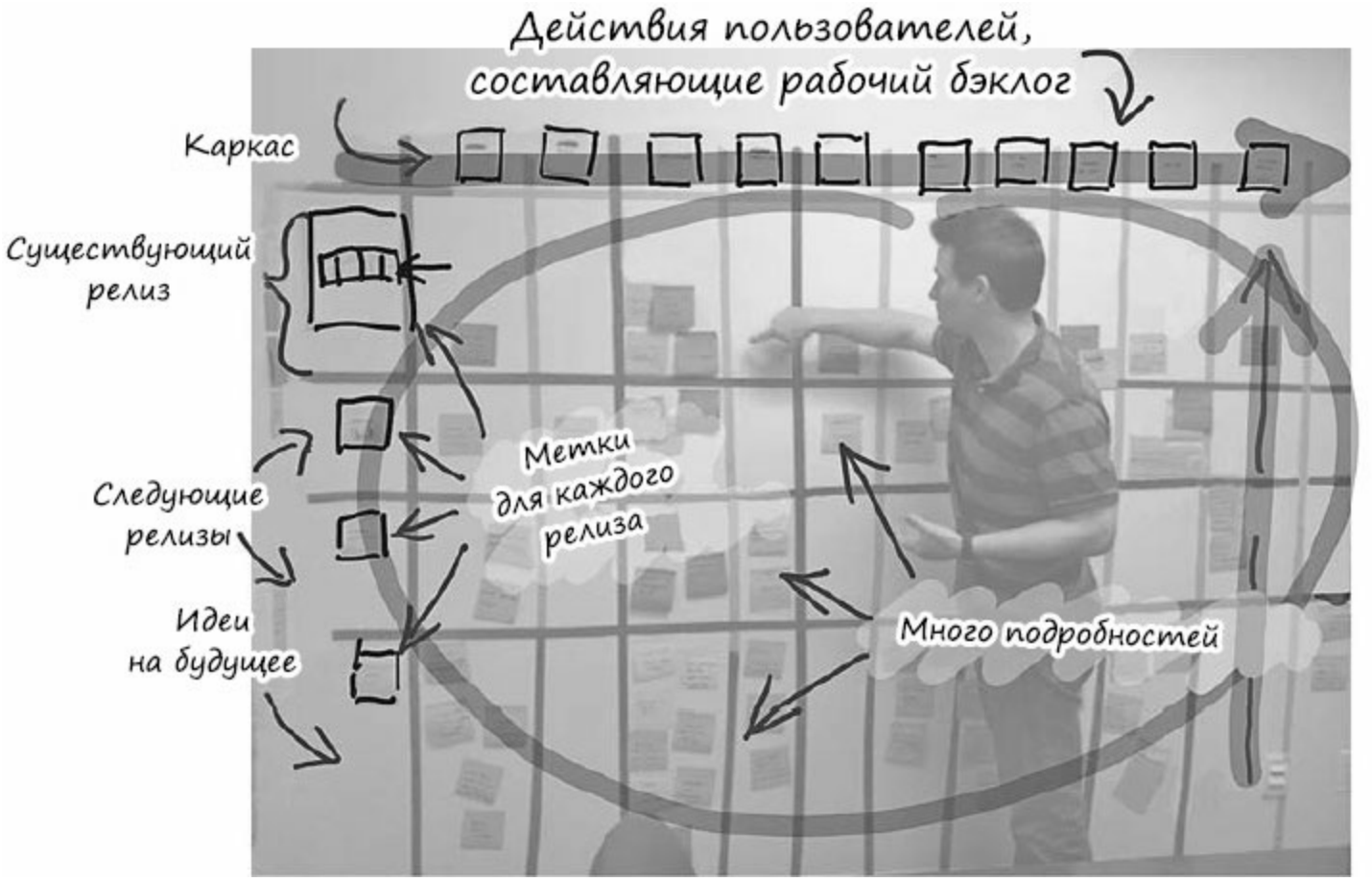
# Разрабатывайте, чтобы исследовать

Для Эрика настало время продемонстрировать свою компетентность.

Эрик и его команда приступили к разработке программного обеспечения. Но их первая цель не разработка минимально жизнеспособного продукта. Пока они собираются сделать совсем немного – продукт должен всего лишь позволить потенциальным пользователям выполнять что-то для них полезное. Вряд ли он сильно впечатлит пользователей, а может быть, даже не понравится некоторым. Этот продукт определенно не будет пользоваться популярностью в отделах маркетинга и продаж. Достаточно лишь, чтобы его увидели только те люди, которые могут воспользоваться им хотя бы один раз и на практике понять, решает ли он их проблему.

К счастью, у Эрика была небольшая группа как раз таких людей. Это были заказчики и пользователи, с которыми он работал раньше, когда изучал проблему и подтверждал ее существование, – партнеры по разработке со стороны заказчиков. Именно они давали обратную связь по первым прототипам. Из этой группы Эрик выделил несколько человек, которые могли бы помочь в дальнейшем исследовании. Они должны первыми увидеть продукт – первую версию, меньше минимальной, определенно нежизнеспособную. Эрик надеялся, что они станут первыми испытателями.

И вот что сделал Эрик.




Здесь Эрик очерчивает часть существующего бэклога. Когда делалась эта фотография, программный продукт уже был предъявлен партнерам по разработке и Эрик назначил встречу для обсуждения обратной связи. Кроме того, команда добавила в продукт несколько простых метрик, чтобы можно было видеть, действительно ли люди используют программу

и что именно они в ней делают.

Эрик отлично знал, что люди, как правило, вежливы и не хотят вас обидеть. Они могут сказать, что им очень нравится продукт, хотя на самом деле они даже не пробовали его использовать. Но от вежливости в данном случае нет никакого толку, ведь настоящий результат, который Эрик хотел бы получить, – это реальная работа пользователей с продуктом. Кроме того, люди часто бывают придирчивы: они могут перечислить множество проблем, которые есть у продукта, или жаловаться на баги, но метрики показывают, что продукт все-таки используется каждый день. Значит, есть чему порадоваться, несмотря на все жалобы, хотя жалобы – это тоже хорошо, ведь благодаря им Эрик получал идеи дальнейших улучшений.

Бэклог Эрика был организован как карта историй с каркасом из желтых стикеров, расположенным наверху. На этих желтых стикерах в виде глагольных фраз были кратко обозначены действия пользователей с продуктом, которые формировали большую и связную высокоуровневую историю. Ниже располагались подробности – конкретные действия пользователей и то, что им нужно для реальной работы с продуктом. И если детали, над которыми работали Эрик и его команда, менялись от релиза к релизу, то каркас оставался неизменным.

Над тем, что находится в верхнем срезе, выше клейкой ленты, Эрик и команда работают прямо сейчас. Этот релиз займет у Эрика два спринта: они используют методологию Scrum, где спринт, как правило, составляет двухнедельный отрезок времени, так что два спринта равны примерно месяцу. Ниже на доске тоже есть несколько срезов. Следующий слайд, по мысли Эрика, содержит то, что должно выйти в следующем релизе, и т. д. Слева от каждого среза точно так же, как было в Globo.com, наклеен стикер с названием релиза и несколькими словами о том, что ребята хотят изучить в нем, – кроме самого верхнего: там был наклеен постер из мультфильма Dilbert – какая-то шутка, понятная членам команды, но неизвестная мне.


Бэклог — карта историй

Доска с задачами для разработки

Если вы присмотритесь, то заметите, что верхняя часть текущего среза пуста – это я дорисовал контуры стикеров там, где они должны были быть. По мере того как члены

команды совместно работали над планированием, они перемещали эти стикеры на доску с задачами по разработке, находящуюся справа от карты историй. Эта доска показывает истории, которые находятся в работе в текущем спринте, вместе с задачами по предъявлению, в рамках которых программисты и тестировщики обрабатывают различные технические детали, необходимые для успешного воплощения идей истории в работающее ПО.

В работе Эрика с бэклогом в виде карты историй есть еще одна хитрость, доказывающая прозорливость команды: высота верхнего среза. Как только Эрик и его команда заканчивают работу над очередным срезом и отправляют его партнерам по разработке – они называют их *бета-заказчиками*, – они перемещают вверх стикеры из нижнего среза. После этого следующий релизный срез обсуждается более детально: команда проговаривает все «что, если…», чтобы обнаружить проблемы и внести недостающие детали. Обсуждаются также некоторые идеи для следующего релиза, в результате этой дискуссии большая идея может быть разбита на две-три более мелкие. И в конце концов устанавливается высота этого среза, чтобы расставить приоритеты и определить, что следует брать в работу первым.

Теперь вы поняли, какие это умные ребята?

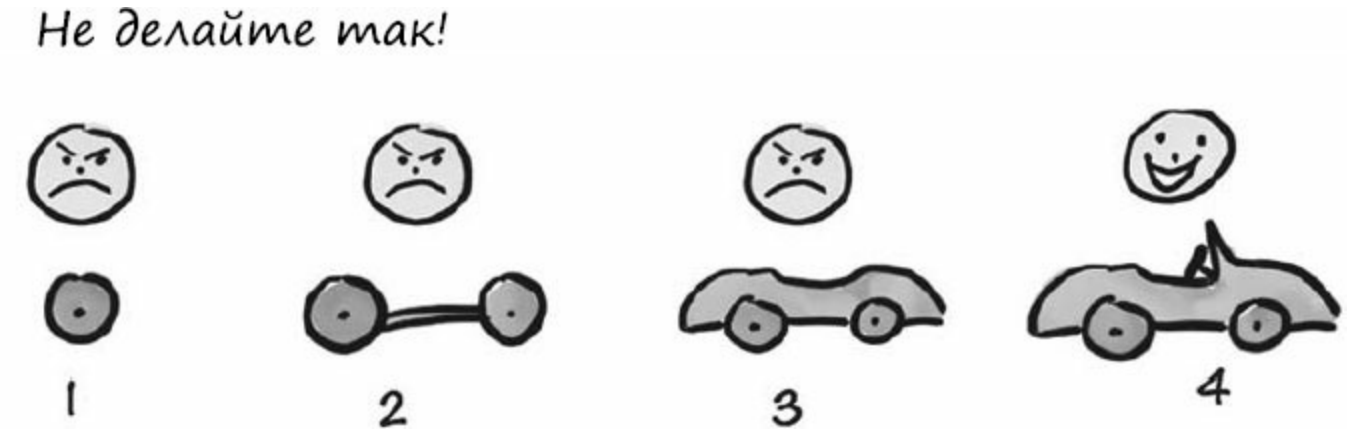
# Повторяйте до жизнеспособности

Эрик мог бы запустить весь этот процесс, сразу оттолкнувшись от идеи минимально жизнеспособного процесса, но он намеренно для начала построил меньше минимума, чтобы затем добавлять понемножку каждый месяц. От партнеров по разработке постоянно поступает обратная связь – как из прямого обсуждения с ними (субъективная), так и из анализа данных (объективная).

Эрик намеревается продолжать в том же духе и дальше, обеспечивая медленный, но постоянный рост проекта и внедрение улучшений, пока его партнеры по разработке не начнут реально использовать продукт в ежедневной работе. Кроме того, Эрик надеется, что эти люди в конце концов порекомендуют продукт другим – настоящим заказчикам. Лишь когда это произойдет, Эрик будет уверен, что создал минимальный *и* жизнеспособный продукт. И только тогда его можно будет без опаски выпустить на рынок, где он, вне всяких сомнений, станет успешно продаваться. Если Эрик с ребятами попробуют продать продукт до этого момента, то, очевидно, будут иметь дело с толпой разочарованных клиентов, ведь с этими людьми не были установлены личные отношения на протяжении процесса разработки, и поэтому они настроены далеко не так снисходительно.

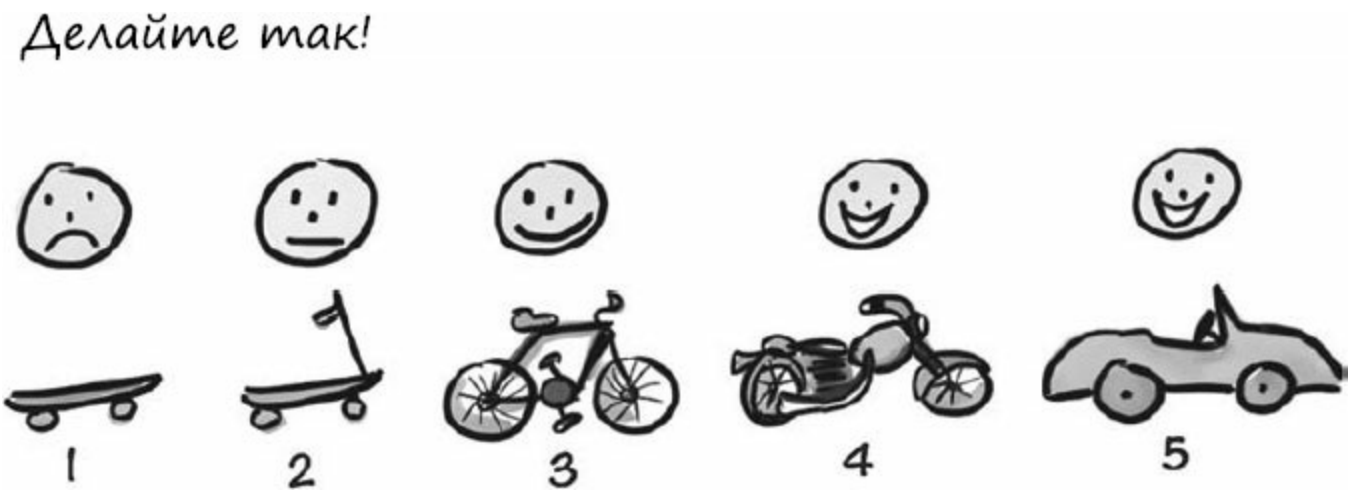
# Как делать не надо

Эрик мог взять свой самый последний, лучший прототип, разбить его на составляющие и начать разрабатывать один фрагмент за другим. Несколько месяцев спустя он был бы готов что-то выпустить и только после этого узнал бы, верны ли его смелые предположения и догадки. И скорее всего, они бы оказались неверны. Можете мне не верить, но чаще всего бывает именно так.




Вот простая иллюстрация, нарисованная моим другом Хенриком Нибергом, – здесь отлично показана ситуация, когда из-за неверной релизной стратегии с каждым выпуском пользователь получает нечто, что не может использовать, пока в последнем релизе не выпустят наконец что-то полезное.

Хенрик предлагает вот такую альтернативную стратегию.




Если я планирую свои релизы по этому принципу, то с каждым выпуском люди получают что-то, что могут реально использовать. Если я, как в этом простом примере, хочу передвигаться на большие расстояния, да еще и таскать с собой какие-то вещи, а вы даете мне скейтборд, я, конечно, буду не слишком доволен. Я скажу вам, как сложно будет уехать далеко на этой штуке, хотя, уверен, есть и любители кататься на скейтборде по автомобильным дорогам. Если вы хотели мне угодить, то, возможно, несколько огорчитесь, но ваша настоящая цель – исследование. Поэтому все нормально: вы узнали, что я хотел бы иметь возможность путешествовать на более дальние расстояния и, кроме того, делать это с комфортом.

По мысли Хенрика, таким образом, дело доходит до выпуска велосипеда, с помощью которого я получаю возможность передвигаться, более соответствующую моим потребностям. А получив мотоцикл, я действительно могу видеть, что ваши идеи работают, – я еду и получаю от этого удовольствие. Таким образом, это и есть минимально жизнеспособный для меня продукт. Если мне очень понравился мотоцикл, возможно, следующим шагом будет его улучшение до более крупной и быстрой модели, скажем, Harley-Davidson, а до спортивного автомобиля дело не дойдет (у меня как раз наступил кризис среднего возраста, поэтому мысль о Harley-Davidson мне очень нравится). Но принять это решение можно только после того, как я опробую мотоцикл и мы оба получим определенный опыт.

> *Рассматривайте каждый релиз как эксперимент и помните о том, что хотите узнать.*

Но что с другими пользователями, которые хотят путешествовать подальше или имеют детей? *Этот* сегмент рынка не удовлетворит ни один этап из перечисленных выше.

Всегда помните о вашей основной целевой аудитории, заказчиках, пользователях, а также результатах, которые надеетесь получить. Обеспечить потрясающий результат для всех типов пользователей очень нелегко, поэтому фокусируйтесь.

# Эмпирическое обучение

По сути, мой друг Эрик применил то, что в методологии Lean Startup называют стратегией *эмпирического обучения*. Эрик знал, что проблемы, которые он решает, пользователи и заказчики, которым он стремится помочь, и даже его собственные решения – только предположения и догадки. В основном это были обоснованные предположения, но все-таки предположения, а не факты. Поэтому Эрик и начал работу по выделению этих предположений, а затем их проверке, продвигаясь от проблем заказчиков и пользователей к уже существующим решениям. На каждом этапе он что-то делал или создавал с конкретной целью – проверить определенные предположения.

Эрик выполнил основную часть из цикла «создание – обратная связь – извлечение опыта», описанного Эриком Райсом. По определению Райса, каждый выпущенный релиз является минимально жизнеспособным продуктом, но, как вы могли заметить, он не был таковым в глазах конечных пользователей и заказчиков, по крайней мере пока не был. Поэтому мне нравится уточнение для определения МЖП Райса – *минимально жизнеспособный продукт-эксперимент* (МЖПэ). МЖПэ – самый маленький продукт, который можно создать, чтобы что-то изучить. В результате я пойму, что же в действительности жизнеспособно с точки зрения основных заказчиков и пользователей.


В ходе этой работы Эрик использовал множество инструментов и техник, хотя изложение историй с использованием слов и картинок помогало ему уже давно. Задействовав карту, с помощью которой были организованы истории, он смог удерживать в голове своих заказчиков, пользователей и этапы их работы с приложением по мере того, как

продукт итеративно улучшался и повышалась его жизнеспособность.

Мне нравится термин *«исследование продукта»* – он отлично описывает то, что мы делаем на этом этапе. Наша цель пока не в том, чтобы что-то создать, – скорее убедиться, что мы создаем правильную вещь. Как оказалось, разработать что-то и дать заказчикам этим какое-то время попользоваться – лучший способ убедиться, что проект развивается в верном направлении. Я позаимствовал определение *исследования* у Марти Когана[10], а мое определение включает практики Lean Startup и Lean User Experience, практики «мышления дизайнера» и множество других идей. Мое понимание того, что нужно делать на этапе исследования, постоянно развивается, но цель остается неизменной: как можно быстрее убедиться, что работа над продуктом идет в правильном направлении.

# Ставьте минимальные эксперименты

Если мы договорились, что наша цель – изучение, то можем создать лишь то, что минимально необходимо для изучения, сконцентрировавшись на том, что хотим узнать. Если вы учли этот принцип, то, вероятно, создаваемое на ранних этапах с целью исследования не может быть продуктом вообще. Скорее даже, если у вас выйдет что-то похожее на продукт, вы, очевидно, сделали слишком много.

Рассмотрим пример. Допустим, я владелец продукта в компании, которая создает ПО для крупных сетевых магазинов. Мне очевидно, что мои продукты будут работать на основе большой базы данных в Oracle. Но также я знаю, что иметь дело с разработчиками базы данных – сущее наказание: они рассматривают, как под микроскопом, каждый мой шаг. Порой, чтобы внести самое простое изменение, требуется неделя работы, а то и больше. Это, разумеется, значительно замедляет работу моей команды. Конечно, ребят, работающих с базой данных, можно понять, ведь от этой базы зависит работа всех других приложений и пропустить какие-то проблемы с ней слишком рискованно. У них есть хорошо налаженный процесс оценки и внесения изменений в базу, он просто занимает слишком много времени.

Самая рискованная часть для меня – убедиться, что мы делаем правильный продукт. Поэтому мы и создаем ранние версии ПО, используя простые базы данных, хранящие данные в памяти. Конечно, они не способны масштабироваться, и поэтому мы никогда не смогли бы выпустить ранние версии для широкой аудитории. Но проводимые на начальных стадиях эксперименты с минимально жизнеспособными продуктами (сейчас мы их даже не называем так) позволяют проверить наши идеи на небольшом количестве пользователей, задействуя в то же время реальные данные. После серии итераций с пользователями, найдя решение, которое, по нашему мнению, будет работать, мы вносим изменения в реальную базу данных и отключаем приложение от базы с данными в памяти. Специалистов по базам данных такой способ тоже устраивает, ведь они знают, что мы вносим изменения лишь после того, как убедимся в верности своего решения.

# Резюме

Гэри использовал карту, чтобы избавиться от ловушки плоского бэклога и увидеть свой продукт целиком, а затем сконцентрироваться на том, для кого он предназначен и чем должен быть.

В Globo.com использовали карту для координации разработки большого плана несколькими командами, а также выделения части продукта, которая, по их мнению, могла быть жизнеспособным решением.

Осталось справиться еще с одной трудностью – настоящей бедой разработки программного обеспечения и ее своевременного завершения. Предположим, вы уверены, что у вас есть нечто достойное быть разработанным. Для многих людей очень важно, чтобы вы закончили разработку к определенной дате. Существует секрет, как сделать эту работу вовремя, на протяжении веков используемый в искусстве. В следующей главе мы рассмотрим, как применить его при разработке ПО.

# Глава 4. План своевременного завершения

Это Аарон и Майк. Они работают в компании под названием Workiva, где выпускают ряд продуктов на основе платформы Wdesk. Эта организация помогает другим большим компаниям решать серьезные задачи и является одной из крупнейших компаний типа «программное обеспечение как услуга», о которых вы, скорее всего, никогда не слышали.


Аарон и Майк выглядят очень довольными. Но это типично для людей, которые работают вместе над решением сложных задач. Или это потому, что один из парней держит в руке бутылку пива? Нет, дело вовсе не в этом. Они довольны, так как только что нашли решение сложной задачи, а пиво является лишь наградой за труд. Кстати, если *вы* не получаете пиво или еще какое-нибудь поощрение за решение сложных задач там, где работаете, стоит с кем-нибудь поговорить об этом.

Аарон и Майк только что завершили очередной раунд исследования продукта, и теперь они уверены, что нашли решение, которое стоит разработать и запустить для использования клиентами.

Исследование началось со смутной идеи: выяснить, кому и каким образом могла бы быть полезна функциональности, с которой они начали работать. Они обсудили идею с заказчиками и проверили свои предположения о том, как те работают в данный момент и с какими проблемами сталкиваются. После этого ребята создали простые прототипы. Аарону и Майку оказалось вполне достаточно создать простые электронные прототипы в Axure и удаленно протестировать их вместе с заказчиками, чтобы убедиться, во-первых, что решение применимо и полезно, а во-вторых, что заказчики его оценят. Необходимости встраивать

прототип данной функциональности в работающее ПО, чтобы получить ответы на свои вопросы, не было.

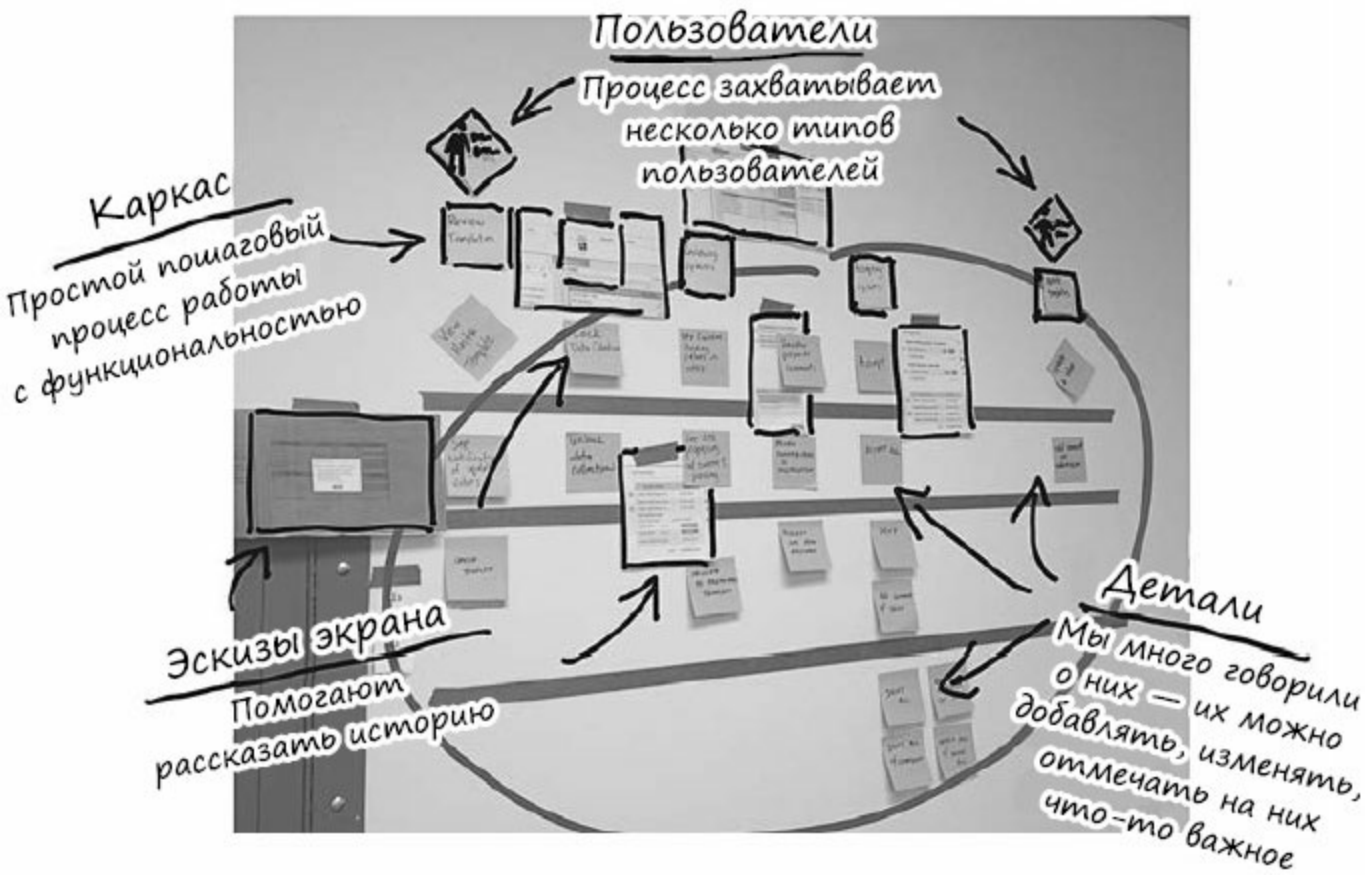
После нескольких итераций с простыми прототипами ребята наконец убедились, что у них есть нечто достойное реализации. Вам может показаться, что это заняло очень много времени, но на самом деле потребовалось около трех дней. Последним шагом стало создание бэклога и плана предъявления функциональности пользователям. План вы можете видеть на фотографии. Это очень хороший план, именно поэтому они так радуются.

Очень важно отметить, что эта карта не охватывает весь продукт – она касается лишь функциональности, которую они добавляют в существующее приложение. Вот почему она намного меньше карты Гэри из главы 1 или карты команд Globo.com. Я подчеркиваю это потому, что некоторые ошибочно полагают: они должны составить карту историй всего продукта, чтобы внести крохотное изменение, и по этой причине отказываются от использования карт вообще.

*Включайте в карту лишь то, что необходимо для поддержки обсуждения.*

# Поделитесь с командой

Чтобы внедрить новую функциональность, Майку и Аарону нужно было, чтобы у них и команды выработалось общее мнение о ней. Ребятам из их команды нужно было осознать проблему и возможности для улучшения, а также оценить, сколько времени займет реализация решения. Для этого Майк с Аароном и создали окончательный вариант карты. С ее помощью они изложили историю функциональности шаг за шагом с точки зрения пользователя. Вы заметили, что на карте присутствуют распечатки экранов? При построении карты ее авторы использовали экраны, отмечая на них детали по мере продвижения, чтобы при ознакомлении с историей легче было понять решение. Именно так поступают сотрудники компании Disney, когда излагают сценарий мультфильма с помощью раскадровок.




Если члены команды задавали вопросы, почему на экране происходит то или иное, Майк и Аарон рассказывали о вариантах, которые они рассматривали, и о том, как вели себя пользователи. Если кто-то спрашивал, что именно происходит после введения данных, ребята были готовы дать ответ, так как уже обдумывали это. А если они не могли ответить, то возможные решения обсуждались всей командой, записывались на стикерах или помечались на эскизах и заносились в модель. Было добавлено довольно много стикеров, касающихся деталей, не предусмотренных Аароном и Майком, но пришедших в голову членам команды. Позднее Аарон признался мне, что команда обнаружила несколько технических зависимостей, о которых они с Майком никогда бы не догадались.

# Секрет верной оценки затрат времени

Каждому, кто давно плавает в море разработки программного обеспечения, хорошо известно: одна из самых трудных задач, которую нам приходится решать, – необходимость оценить, сколько времени займет разработка. Я хочу открыть вам один из самых важных секретов правильной оценки времени.

> *Лучше всего затраты времени оценивают те разработчики, которые верно понимают, что именно они оценивают.*

Существует множество методов, с помощью которых, как предполагается, можно дать очень точную оценку времени, но я не буду приводить их здесь. Я лишь *хочу* сказать: ни один из этих методов не будет работать, пока все люди, разрабатывающие концепцию ПО и реализующие ее, не начнут понимать ситуацию одинаково.

При выработке одинакового понимания временные оценки не должны замалчиваться. Поэтому лучше всего прямо сейчас обсудите их с кем-нибудь еще.

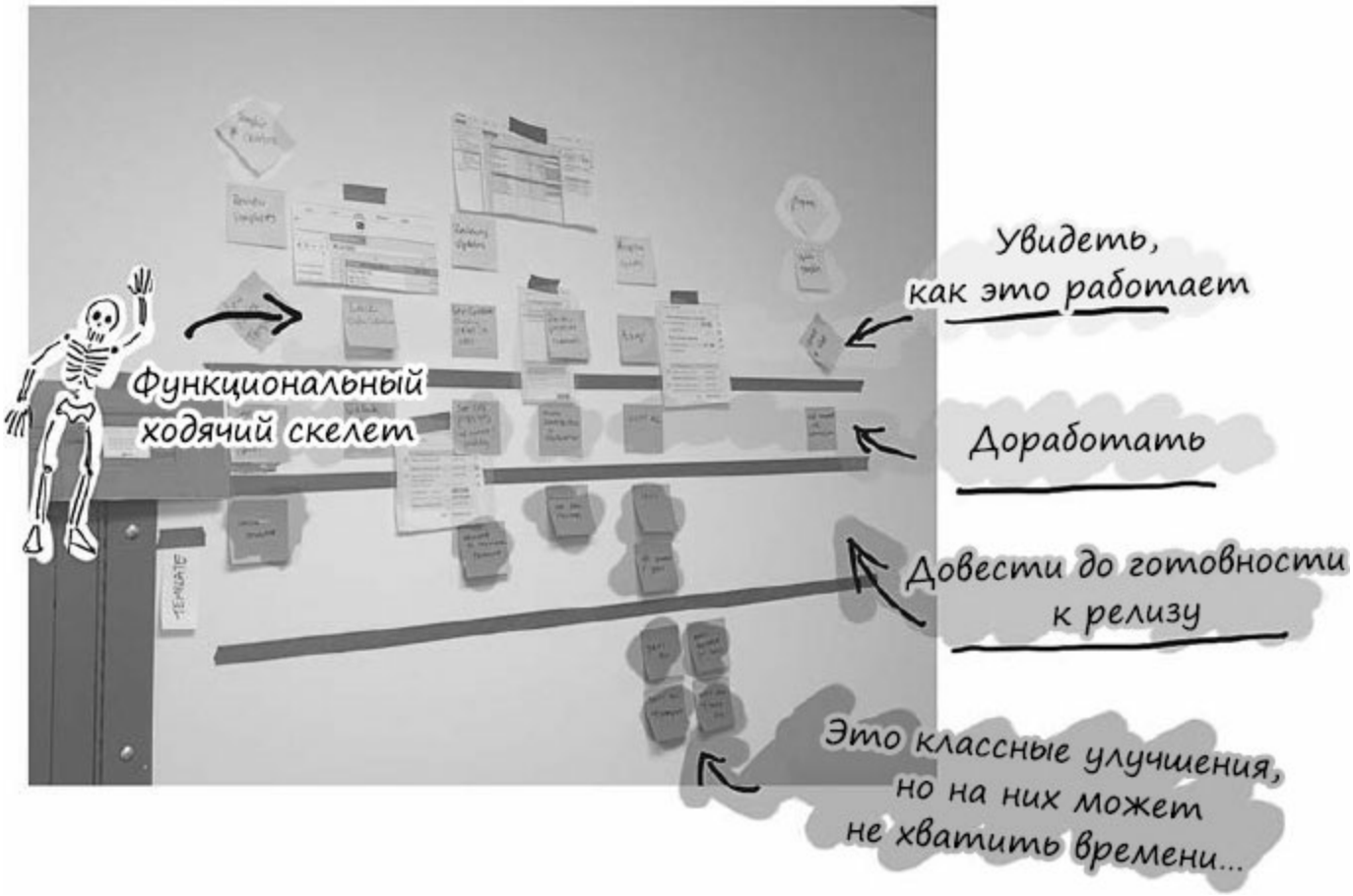
# Планируйте разработку шаг за шагом

У команды Workiva не было возможности уменьшить количество необходимых элементов в разработке. Они не могли проделать то, что сделала команда Globo.com, как описано в главе 2, то есть оставить часть запланированного на потом. В Workiva было решено, что нужно разрабатывать сразу все. На этапе прототипирования у них уже была возможность исключить многое и подтвердить, что решение до сих пор остается полезным для заказчиков. Тем не менее на карте можно выделить три среза.

«Зачем они вообще тратили на это время?» – можете удивиться вы. Действительно, предъявить треть от того, что нужно заказчикам, – это примерно как продавать одну треть спортивного автомобиля: никто не сможет на нем ездить. Но Майк владелец продукта. Он не может отойти в сторону после того, как найдено хорошее решение. Его роль немного меняется, и сейчас он больше похож на режиссера в кино: должен присутствовать при съемках каждой сцены. И ему нужно решить, какие сцены должны быть отсняты первыми, а какие – последними. Он знает, что к концу съемок фильма все сцены сложатся вместе и составят одно последовательное повествование.

Поэтому Майк и работал со своей командой над составлением плана разработки. Вот что они сделали – разделили карту тремя поперечными срезами.

Первый срез включает в себя основные функциональные элементы. Как только все фрагменты, входящие в этот срез, будут реализованы, работа с функциональностью может быть проделана в приложении шаг за шагом. Функциональность будет работать не во всех ситуациях, где запланировано, поэтому предъявлять ее пользователям на этом этапе нельзя: последует лавина жалоб и возмущений. Но зато Майк и его команда смогут понаблюдать за работой функциональности с начала до конца. Они смогут ввести реальные данные и оценить производительность приложения, а также применить некоторые средства автоматизированного тестирования, чтобы оценить способность к масштабированию нагрузки. Можно узнать многое о технических рисках, которые способны вызвать значительные проблемы позднее. Таким образом, двигаясь вперед, команда будет более уверена в своевременном окончании разработки. Или по крайней мере будут выявлены непредвиденные трудности, которые могут замедлить работу. Первый срез я называю *функциональным ходячим скелетом* – этот термин я позаимствовал у Алистера Коберна. Я слышал также, что в этом значении употребляются слова «стальной каркас» или «трассирующая пуля».




Второй срез нужен для дополнения функциональности и приведения ее в состояние, близкое к готовности для релиза. Кроме того, без сомнения, в процессе этого возникнет что-то непредусмотренное, например, упущенные характеристики, необходимые для работы функциональности, важные детали, которые невозможно было выявить, работая с прототипом. Возможно, обнаружится, что производительность системы ниже ожидаемой и необходима дополнительная работа, чтобы довести скорость до нужной величины. Все эти так называемые предсказуемые неожиданности – концепция, близкая к «неизвестным неизвестным» Дональда Рамсфилда. Не думайте, что ничего такого не обнаружится. Вы же знаете: они существуют.

И наконец, третий срез нужен для окончательной доработки функциональности и доведения ее до наилучшего состояния. Кроме того, на этом этапе придется разбираться с теми самыми неожиданностями.

# Не каждый срез достоин релиза

Все перечисленные срезы не являются запусками продуктов для заказчиков и пользователей – это всего лишь своего рода вехи, на которых члены команды должны приостановиться и оценить, как далеко они зашли. Но пусть это вас не смущает: с точки зрения заказчика или пользователя, работа еще не закончена.

Команды Майка и Аарона оценили, что работа над функциональностью потребует примерно двух месяцев. Как и Эрик, они использовали двухнедельные спринты, значит, для реализации потребовалось бы четыре спринта. Я думаю, они могли бы разделить карту на четыре среза, чтобы на каждый пришелся один спринт, но ребята не рассматривали срезы с этой точки зрения. И вам тоже лучше так не делать. Воспринимайте срезы просто как наборы определенных задач по разработке, каждый со своей исследовательской целью. А решать, в какие спринты или итерации войдет очередной набор, нужно по мере необходимости.

# Еще один секрет верной оценки временных затрат

Еще один страшный секрет оценки (который на самом деле вовсе не должен быть секретом) заключается в том, что оценки… оцениваются заранее. Я уверен: порывшись как следует в сборниках оксюморонов, выложенных в Интернете, вы непременно найдете там термин *«точная оценка»*. Если нам точно известно, сколько времени займет какая-то работа, разве мы вообще называли бы это *оценкой*?

Тем не менее, если вы создаете маленькие фрагменты программного обеспечения, одно из явлений, в которых вы уверены, – это время, которое понадобилось на разработку. Эта величина является *измерением*, и поэтому она куда точнее оценки.

Таким образом, второй секрет очевиден: чем чаще вы делаете измерения какой-то величины, тем легче вам становится предсказывать ее. Если вы каждый день ездите в офис, то, думаю, легко предскажете, сколько времени займет дорога сегодня. Если же я спрошу вас, за сколько времени вы доберетесь в другое место примерно в том же районе, то, уверяю, вы ошибетесь в оценке не более чем на 10 минут. Именно так работают оценки временных затрат.

Разделяя большие куски работы на меньшие, мы получаем больше возможностей для измерений. Какие-то тонкости, конечно, есть всегда, но в целом предсказания всегда будут тем точнее, чем больше есть примеров выполнения аналогичной работы, где известны временные затраты.

Как владельцу продукта, Майку чрезвычайно важно уложиться в сроки при выпуске этой функциональности. И, как хороший владелец продукта, он старается, чтобы каждый в его маленькой команде чувствовал, насколько он значим для достижения этой цели. Майк относится к предварительным оценкам времени так же внимательно, как к бюджету на выпуск.

# Регулируйте свой бюджет

Еще на самой ранней стадии Майк и Аарон работали вместе с программистами, которым доверяли, над начальной оценкой временных затрат. Они рассматривали эту величину как *бюджет* и старались хорошо организовать его.

С каждым небольшим фрагментом, который создавала команда, они могли получить измерения реальных затрат времени. Каждый кусочек работы расценивался как трата средств из бюджета. К примеру, могло оказаться, что бюджет израсходован наполовину, в то время как выполнена лишь треть работы по реализации запланированной функциональности. Этого, конечно, никто не ожидал, но по крайней мере они могли стать еще внимательнее и каким-то образом на это реагировать. Например, можно было бы позаимствовать бюджет у других функциональностей, над которыми команда в то время работала. Или немного упростить функциональность, но так, чтобы не особенно затронуть интересы пользователей. В конце концов они могли просто смириться с неизбежным и поискать способы как-то изменить ожидания людей, которым пообещали сделать работу к определенной дате.

В зависимости от того, насколько плохо будут обстоять дела, понадобится увеличение запасов пива.

Разделяя карту на срезы для перехода к стратегии разработки, ребята стараются как можно раньше выявить элементы, которые, скорее всего, подорвут бюджет. Это рискованные элементы, и найти их можно, только обсуждая продукт со всей командой.

### *Выявление рисков при составлении карт историй*

*Крис Шинкл, SEP*

Крупная компания по обеспечению безопасности хотела создать беспроводную систему контроля доступа для зданий средних размеров (школы, кабинеты врачей, магазины и т. п.) в средней ценовой категории. Компания обратилась в SEP для разработки встроенного программного обеспечения для замков, а также беспроводного терминала ZigBee, с помощью которого обеспечивалось взаимодействие с ними.

Проект был невероятно интересным с технической точки зрения, но одновременно богатым вероятностью сесть в лужу: ограниченный бюджет, узкие временные рамки, изменения, вносимые руководством по ходу разработки, непроверенные технологии, а также постоянное увеличение объема работы.

Естественно, все немедленно пошло наперекосяк. Команда проекта нарушила сроки выполнения нескольких промежуточных этапов. Клиент был недоволен, и, конечно, команда была деморализована. В ходе поиска причин сложившейся ситуации разработчики выяснили, что главной причиной нарушения сроков стала незапланированная работа, в основном вызванная неопределенностями и оправдавшимися рисками. Нужно было срочно что-то менять.

Команда немедленно взялась за решение проблемы, как поступили бы на ее месте все умные люди. Что же они придумали? Они решили модифицировать карту историй.


С высокоуровневой точки зрения они увеличили частотность и точность своей карты историй. Увеличивая частотность отображения истории для каждого временного промежутка, соответствующего релизу, они ожидали повышения вероятности обнаружения рисков, а увеличив точности карты, которая теперь включала рискованные истории наряду с обычными действиями, задачами и деталями, надеялись визуализировать, обсуждать и обрабатывать риски.

Результаты оказались поразительными.

Команде было известно, что ширина и глубина обычной карты историй позволяют ощутить размер проекта. Кроме того, разработчики знали, что количество способов прохождения карты – хороший индикатор уровня сложности. Но так как риски и неопределенности не были отображены на карте историй, она не отражала действительный объем работы (включая исследования), которую требовалось проделать.

Новая карта, включающая в себя рискованные истории, позволила оценить объем и сложность работы более точно. Эти величины оказались лучше на ней представлены потому, что их составляли не только известные запланированные истории, но и новые, неизвестные рискованные истории. Их можно было расценивать как знания, которые команда должна была получить, прежде чем уверенно двинуться вперед, работая с известными историями.

Как вы уже догадались, новая карта оказалась куда более полезной для планирования, чем прежняя. На ней были хорошо видны риски и неопределенности, проработка которых требовала времени. Возможность

запланировать это время сделало работу команды куда более предсказуемой и гибкой.

Обнаружилось и еще одно выгодное следствие такого подхода: возможность измерять и обновлять то, как сотрудники продвигаются в исследованиях. Вдобавок к традиционной диаграмме сгорания задач команда теперь обновляла и диаграмму снижения рисков. Особенно полезно это было в тех случаях, когда заказчик был недоволен состоянием диаграммы сгорания задач, но мог оценить усилия команды по диаграмме снижения рисков.

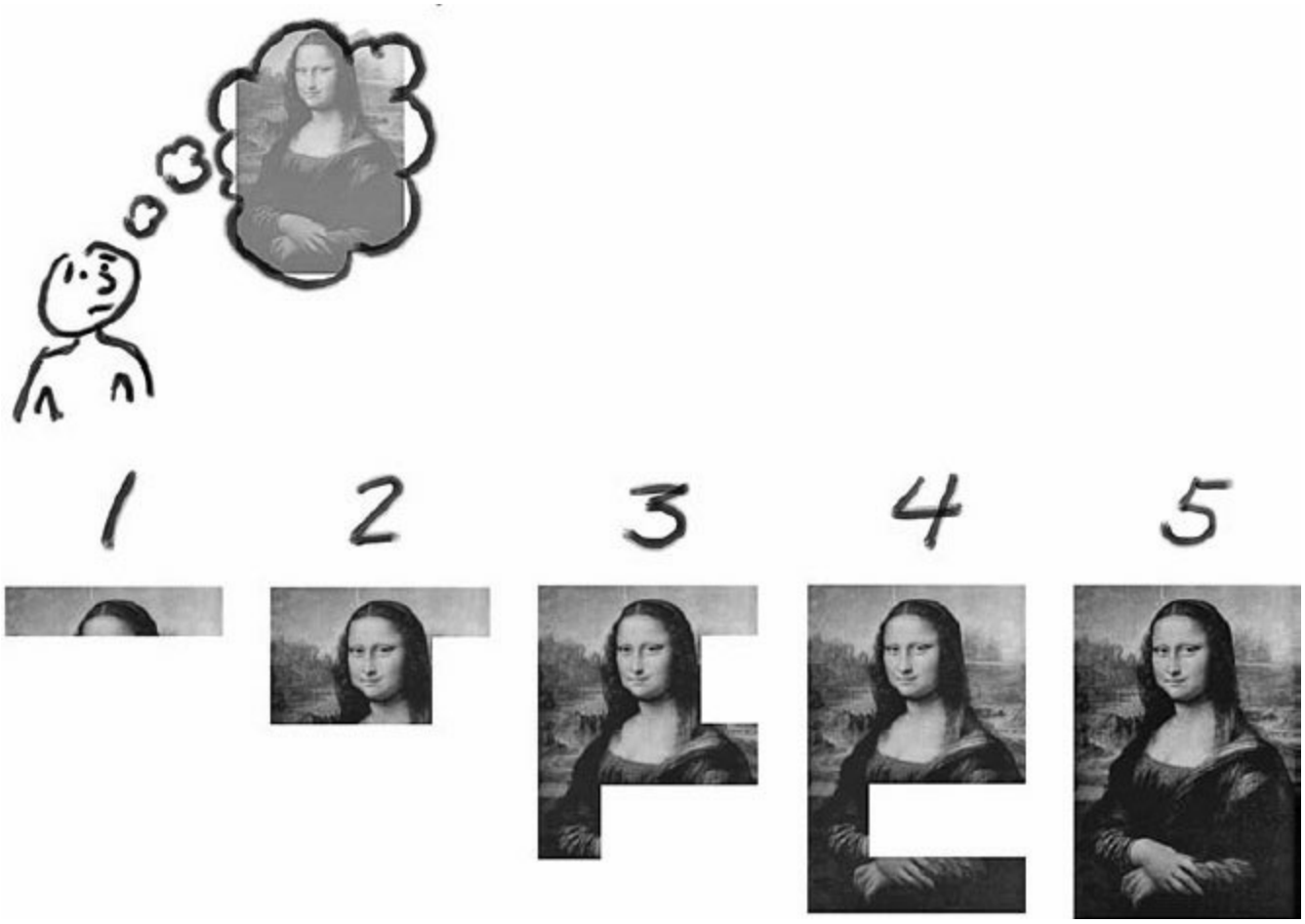
К концу дня вся команда была убеждена, что снижение частоты создания карт историй и добавление рискованных моментов – отличные способы приближения карты историй к реальности.

**Что сделал бы да Винчи?** Я часто задаю себе этот вопрос. Ладно, хорошо, не задаю. Но, возможно, мне следовало бы так поступать.

По сути Майк и Аарон выбрали ту же стратегию, которой пользуются художники, когда хотят завершить свои произведения вовремя. Точно такой же подход я применяю при создании программного обеспечения на протяжении многих лет. А когда я познакомился со своими будущими друзьями в Globo.com, оказалось, что и у них такой же подход, потому что, как я уже говорил, даже если они запоздают с разработкой нового интерактивного продукта к Олимпиаде, Олимпийский комитет не станет переносить открытие Игр. Я также думаю, что и вы привыкли следовать той же стратегии, даже не задумываясь над этим.

Начну объяснение с того, чего не делал Леонардо да Винчи, но, к сожалению, часто пытаются проделать обычные люди, занимающиеся созданием программного обеспечения.

Представьте себе, что вы Леонардо да Винчи и собираетесь написать картину, следуя той же стратегии, к которой традиционно прибегают большинство команд в мире разработки[11]. Вы можете начать работу с четким представлением о картине, по крайней мере вам так кажется. Вы разбиваете картину на несколько частей. Допустим, на эту работу отведено пять дней и каждый день вы рисуете по одной части. А к концу пятого дня – бах! – вы закончили! Что может быть проще?

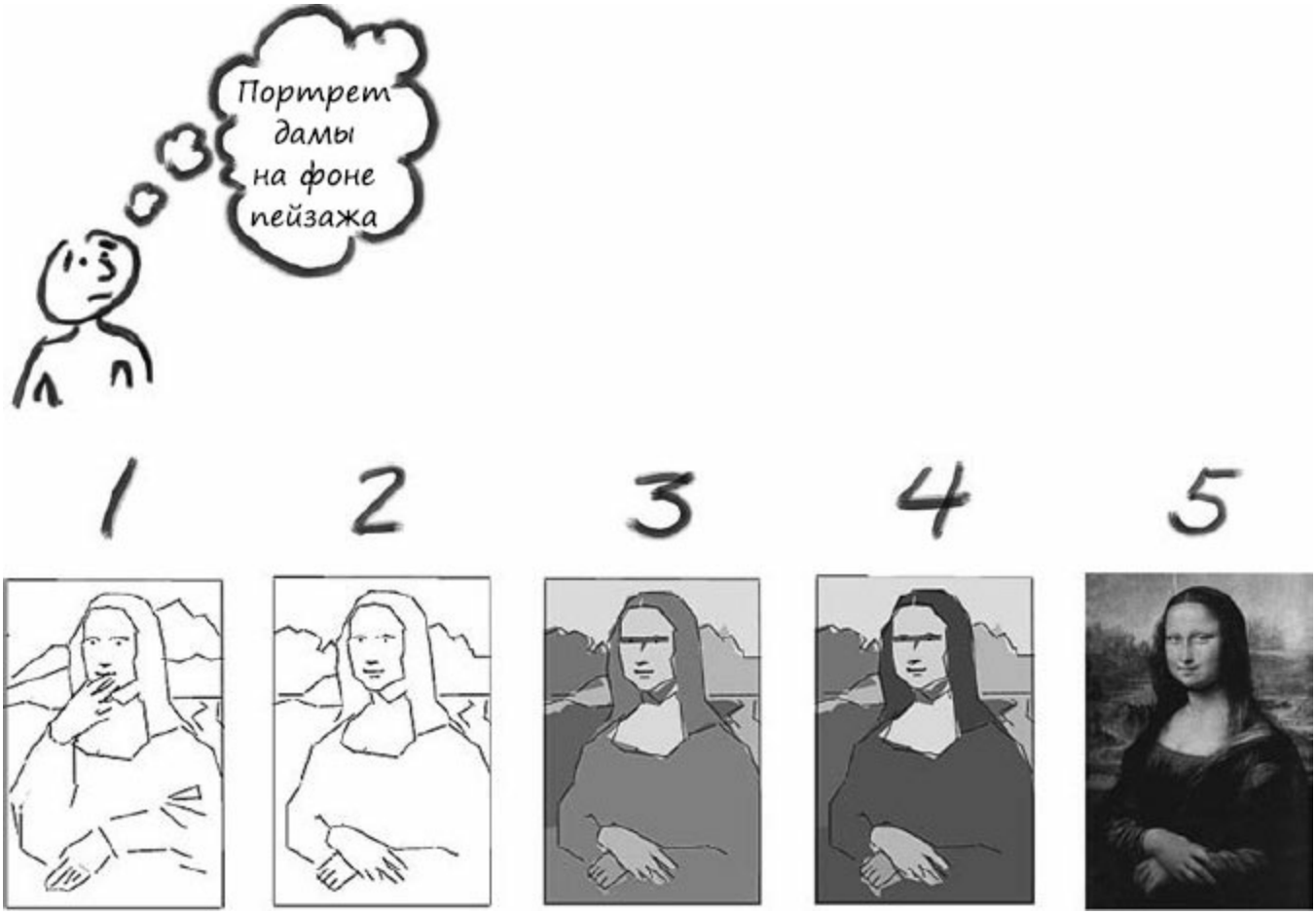



Проблема лишь в том, что в реальности это не работает, по крайней мере для художников. Что-то создать таким образом можно, если предположить, что изначальное видение точно и правильно. Необходимо также, чтобы автор был высокопрофессионален и способен точно выделить части работы без визуального представления их в контексте. Если вы поступаете подобным образом при разработке программного обеспечения, это называется *стратегией постоянного прироста (инкрементальной)*. По сути, таким образом каменщик строит стену, и этот способ хорошо работает, если размеры всех кусочков одинаковы и хорошо определены – как у кирпичей.

Когда в детстве я учился рисовать, то тоже наступил на эти грабли. Я любил изображать животных и, как правило, начинал с головы. Я работал над головой, пока она не становилась, по моему мнению, идеальной, а затем переходил к рисованию остальных частей тела – ног, туловища и пр. Но когда работа подходила к концу, мне становилось очевидно, что пропорции животного на рисунке совершенно не согласованы. Голова была слишком велика или слишком мала по сравнению с телом, ноги соединялись с ним под неестественным углом, а поза животного была вообще ни на что не похожа. Но все это можно было списать на особое видение талантливого шестилетнего художника, ведь, как мы знаем, все шестилетние художники очень талантливы.

Намного позже я осознал ценность предварительного наброска всей композиции. В данном примере я должен был сначала верно определить пропорции, уточнить позу, а может быть, даже изменить первоначальный замысел рисунка.

Мне, к сожалению, не приходилось наблюдать за Леонардо, но, я полагаю, он делал что-то очень похожее.




Даже Леонардо да Винчи, вероятно, признал бы, что его первоначальное видение картины не было идеальным, а он кое-что понимал в рисовании. В первый день, как мне представляется, он рисовал эскиз композиции или, может быть, делал легкий набросок. Я уверен, что на этом этапе он вносил в композицию небольшие изменения: «Хм, кажется, здесь самое важное – ее улыбка. Надо убрать руку от губ. А эти горы на заднем плане… Кажется, они слишком высокие, недостает пространства».

К середине недели Леонардо уже нанес на картину большую часть форм и красок, но все еще мог вносить изменения и делал это. К концу недели, видя, что не укладывается в назначенный срок, Леонардо прикладывает все усилия к проработке деталей картины. Некоторые до сих пор гадают, почему у Моны Лизы нет бровей: это осознанное решение художника или ему просто не хватило времени проработать эту особенность?

> *«Работу над великим произведением искусства невозможно завершить – ее можно только прервать» (Леонардо да Винчи).*

Эту цитату приписывают Леонардо да Винчи. Фраза означает, что мы можем продолжать добавлять и улучшать что-то бесконечно, но в какой-то момент все-таки придется прерваться и выпустить наше произведение в мир. И если Леонардо и множество других великих художников могут служить для нас примерами, мы – люди, которые смотрят на их работы, – даже не догадываемся, что работа была прервана. Для нас она выглядит завершенной.

# Итерации и прирост

Любой художник или писатель работает именно так. Люди, которые совмещают чтение утренней газеты и просмотр вечерних новостей, по сути, следуют этому принципу. Люди, играющие в театре, следуют этому принципу. Каждому, кто должен выпустить какой-то продукт в точные сроки и приобретает знания по ходу дела, без сомнения, знакома такая стратегия.

> *Итеративно мыслите, чтобы верно оценивать, и вносите изменения в то, что вы уже сделали.*

В области разработки программного обеспечения термин *«итерация»* имеет два значения. С точки зрения процесса он означает повторение одних и тех же действий снова и снова. Вот почему отрезок времени, в течение которого идет разработка, в методологии Agile часто называют *итерацией*. Но, используя этот термин в отношении программного обеспечения, которое вы уже создали, вы подразумеваете его оценку и изменение. Часто необходимость внести изменения во что-то уже существующее выглядит как провал: если бы требования были прописаны лучше или границы проекта не расширялись постоянно, то люди, принимающие решения, что и как программировать, не ошибались бы и ничего исправлять бы не пришлось. Но, как мы знаем, на самом деле исправления – всего лишь неизбежный результат увеличения знаний.

> *Итеративно мыслите, чтобы добавлять новое.*

К сожалению, завязнуть в водовороте итераций очень легко, поэтому держите в поле зрения календарь и продолжайте постепенно внедрять новое. Художники совершенствуют свою работу, не только добавляя в картину какие-то ранее отсутствующие детали, но и прорабатывая то, что уже написано.

Вы можете следовать тому же принципу при разработке программного обеспечения, для начала создав простую версию функциональности без каких-либо деталей. Можете рассматривать ее как набросок, эскиз. Некоторое время поработав с этой простой версией, можно ее усовершенствовать – добавить дополнительные функции. Еще через некоторое время она развивается настолько, что работу можно считать законченной – она соответствует вашему первоначальному видению. А если вы все делали правильно, то, возможно, результат работы будет отличаться от изначального замысла в лучшую сторону, ведь идея получает развитие, потому что по ходу работы вы узнаете и понимаете много нового.

# Дебют, миттельшпиль, эндшпиль

Возможно, это немного взорвет ваш мозг, но я собираюсь объединить несколько метафор. Лично мне нравится аналогия между созданием программного обеспечения и шахматной партией. Сразу говорю: я небольшой мастер игры в шахматы и в общем умею только двигать фигуры, поэтому, если мои метафоры неверны, не стоит писать мне и поправлять. Независимо от размера продукта или функциональности я предпочитаю делить релизный бэклог на следующие три части.

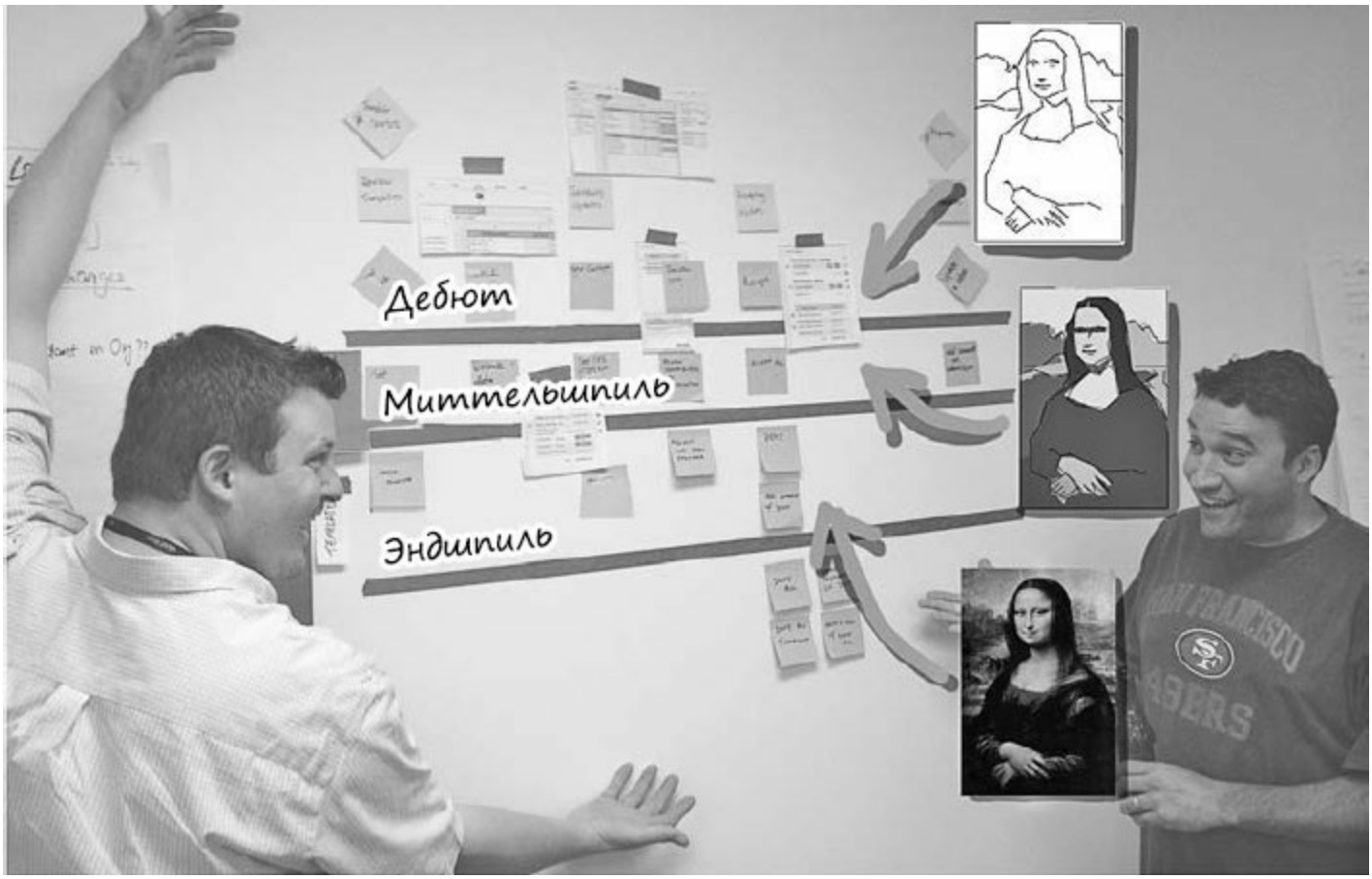
**Дебют.** Сфокусируйтесь на самых необходимых функциях или пользовательских шагах, которые затрагивают весь продукт. Сфокусируйтесь на самых трудных или рискованных частях работы. Отложите различные варианты действий пользователей. Отложите замысловатые бизнес-правила, которые понадобятся вам позднее, перед релизом. Создайте только то, что необходимо для базовой работы продукта от начала до конца.

**Миттельшпиль.** Дополняйте и прорабатывайте функции. Добавьте код, поддерживающий различные варианты действий, которые может выполнить пользователь. Реализуйте строгие бизнес-правила. Если вы хорошо поработали на стадии дебюта, то можете начать тестировать основной процесс продукта на соответствие требованиям производительности, масштабирования нагрузки и удобства использования – все это показатели качества, которые нелегко обеспечить. Обязательно отслеживать их постоянно.

**Эндшпиль.** Наведите на свою работу блеск. Сделайте ее приятной и эффективной в использовании. Поскольку сейчас вы можете протестировать продукт на реальных данных и в реальном масштабе, вы обязательно обнаружите возможности для улучшения, которые невозможно было заметить на стадии прототипа. На этом этапе также можете получить обратную связь от пользователей и внести необходимые изменения.

# Разделите стратегию разработки прямо на карте

Если вы определились, что именно войдет в первый жизнеспособный релиз, который можно показать заказчикам и пользователям, вместе с командой начните работу, чтобы разделить истории этого релиза на части, которые войдут в дебют, миттельшпиль и эндшпиль. Никто не сумеет оценить риски и благоприятные возможности для исследований лучше людей, которые создают продукт, кроме того, они будут ощущать большую заинтересованность в выполнении плана, который составлен с их участием.




Именно так и поступили Майк с Аароном и вся команда разработчиков. Теперь вы поняли, почему они так рады?

# На самом деле суть в рисках

В главе 3 Эрику пришлось много возиться, при этом он рисковал неверно определить продукт. Он прибегнул к стратегии выделения отдельных релизов, что позволило ему всякий раз показывать заказчикам весь продукт.

В этой главе Аарон и Майк сконцентрировались на технических рисках – факторах, которые могли нарушить график разработки или непредсказуемо увеличить затраты. Они не демонстрировали результаты своей работы пользователям и заказчикам в конце каждого цикла, так как это никак не помогло бы снизить риск, но очень строго и придирчиво рассматривали их самостоятельно и с помощью команды, а затем использовали то, что удавалось узнать, для безопасной разработки продукта.

Возможно, при чтении главы 2 вы обратили внимание на один тонкий момент – Эрику понадобились два двухнедельных спринта, чтобы создать очередной минимально жизнеспособный продукт-эксперимент. Ему пришлось решать, какие функциональности необходимо включить в первый спринт, а какие – во второй. Для этого он использовал примерно такой же подход. Он и его команда обработали самые рискованные части в первую очередь, чтобы как можно скорее увидеть их в работе и, если понадобится, внести изменения по ходу разработки, прежде чем их увидят заказчики.

# Что теперь?

Вы познакомились с несколькими отличными примерами создания и применения карт для различных целей. Но карты можно использовать и для других задач – мы рассмотрим их в следующих главах. Прежде чем идти дальше, хочу продемонстрировать вам свою любимую хитрость, которую я применяю при обучении других использованию карт. Уверяю: стоит вам ее освоить – и вы прямо сейчас сможете создавать карты на экспертном уровне.

Встретимся в главе 5.

# Глава 5. На самом деле вы уже всё умеете!

Если вы думаете, что создание карты историй – это сложный, загадочный или в каком-нибудь еще смысле трудный процесс, то разрешите мне прямо сейчас доказать, что это не так. На самом деле у вас уже сейчас есть все, чтобы разобраться в базовых понятиях, необходимых для создания карты. Давайте прямо сейчас рассмотрим пример, взятый из обычной жизни. А чтобы было интереснее, возьмем его из *вашей* жизни. По ходу дела я познакомлю вас с самыми важными понятиями процесса составления карт, суть которых вы уже понимаете.

Приготовьте пачку стикеров, ручку – и начнем. Не спешите, приступим, когда у вас будет достаточно времени. Я подожду.

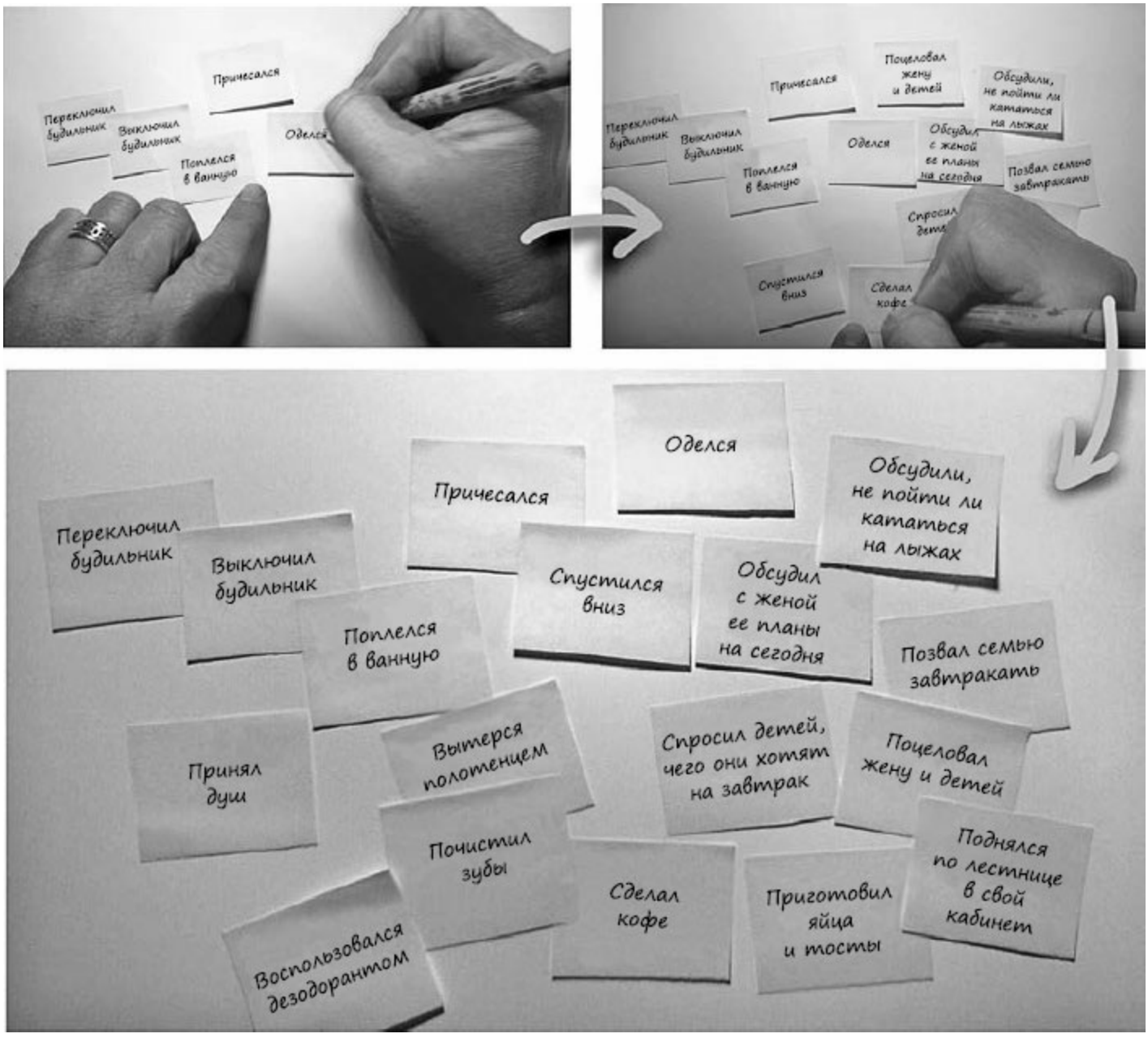
Готовы?

# 1. Запишите свою историю по одному шагу за раз

Закройте глаза и вернитесь к моменту пробуждения сегодня утром. Вы же *проснулись* сегодня утром, правда? Что вы сделали прежде всего? А сейчас откройте глаза и запишите это действие на стикере. Я тоже сделаю это: мое первое действие сегодня утром было «Переключил будильник», что я и написал на бумажке. Не слишком интересно, правда? Хотя бывают дни, когда мне приходится переключать будильник два или три раза.

А сейчас оторвите этот стикер и приклейте его на стол перед собой. Затем подумайте о следующей вещи, которую вы сделали. Вспомнили? Теперь запишите ее на следующем стикере и приклейте его, поместив справа от предыдущего. Продолжайте в том же духе. На моих следующих стикерах написано: «Выключил будильник» и «Поплелся в ванную».

Продолжайте записывать свои действия на бумажках, пока не доберетесь до выхода из дома на работу или куда-то еще, куда вы собирались сегодня. Я обычно заканчиваю на «Подошел к машине» и перехожу к пути на работу. Думаю, вам понадобится не больше трех-четырех минут, чтобы написать все эти стикеры.



## Задачи – это то, что мы делаем

Посмотрите на то, что у вас получилось. Вы заметили, что все или почти все стикеры начинаются с глагола? Короткие глагольные фразы, вроде «почистил зубы» или «принял душ» – это *задачи*, или попросту действия, необходимые для достижения какой-то цели. Когда мы описываем задачи людей, использующих наше ПО и выполняющих по порядку какие-то действия, чтобы достичь определенной цели, мы называем их *пользовательскими задачами*. Это самое важное понятие в создании хорошей карты историй, не говоря уже о написании и изложении хороших историй. Вы убедитесь, что почти все стикеры в картах историй содержат короткие глагольные фразы, обозначающие действия, которые делают люди при использовании вашего ПО.

А сейчас остановимся на минуту – подумайте, было ли это сложно? Я попросил вас записать то, что вы делали, и задачи легко возникли у вас в голове. Не правда ли, хорошо, что самое важное в процессе происходит так естественно?

Поэтому не стоит бояться слова *«задача»*. Если вы менеджер проекта, то должны знать, что проектные планы полностью состоят из задач. Если вы когда-либо использовали истории в разработке Agile, то знаете, что планирование включает в себя написание большого количества задач на разработку и тестирование. Если же вы не менеджер и не программист, то будьте осторожны, употребляя слово *«задача»*: другие люди могут понимать под ним то, что привычно для *них*, как в примерах, приведенных ранее, и могут сказать, что вы ошибаетесь.

> *Пользовательские задачи – основной строительный материал для карты историй.*

А сейчас посчитайте, сколько задач вы записали.

Большинство людей в этом упражнении записывают от 15 до 25 задач. Если вы записали больше, это поразительно. Если меньше – что ж, можно позавидовать простоте вашей жизни, я бы тоже хотел собираться по утрам так быстро. Но, может быть, вам стоит еще раз посмотреть на свой список и проверить, не пропустили ли вы чего.

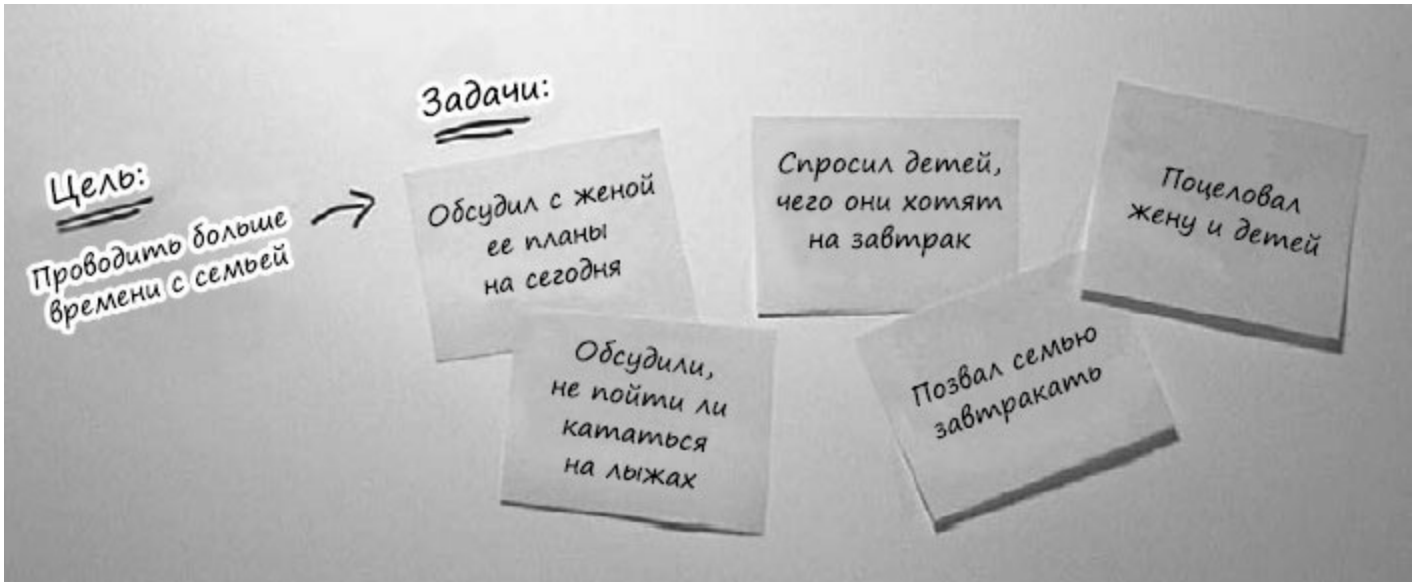
## Мои задачи отличаются от ваших

Думаю, для вас не станет новостью то, что люди отличаются друг от друга. Эти различия хорошо заметны в способах, которые люди выбирают, чтобы чего-то добиться.

Например, многие достаточно мотивированы и дисциплинированны, чтобы каждое утро делать зарядку. Если у вас есть хоть один стикер, связанный с физкультурой, жму вам руку! Я все еще работаю над собой в этом смысле.

У других людей больше утренних обязанностей по сравнению с вами из-за условий, в которых они живут. Например, если у вас есть дети, могу поспорить, что вы записали несколько задач, которые людям бездетным и в голову не придут. Если есть собака, то, должно быть, найдется задача-другая, связанная с заботой о животном.

Не забывайте об этом, думая о людях, которые будут использовать ваше ПО. Они могут работать с приложением, имея различные цели. Они могут применять его в разных обстоятельствах, из-за чего им приходится учитывать интересы других людей или процессов.



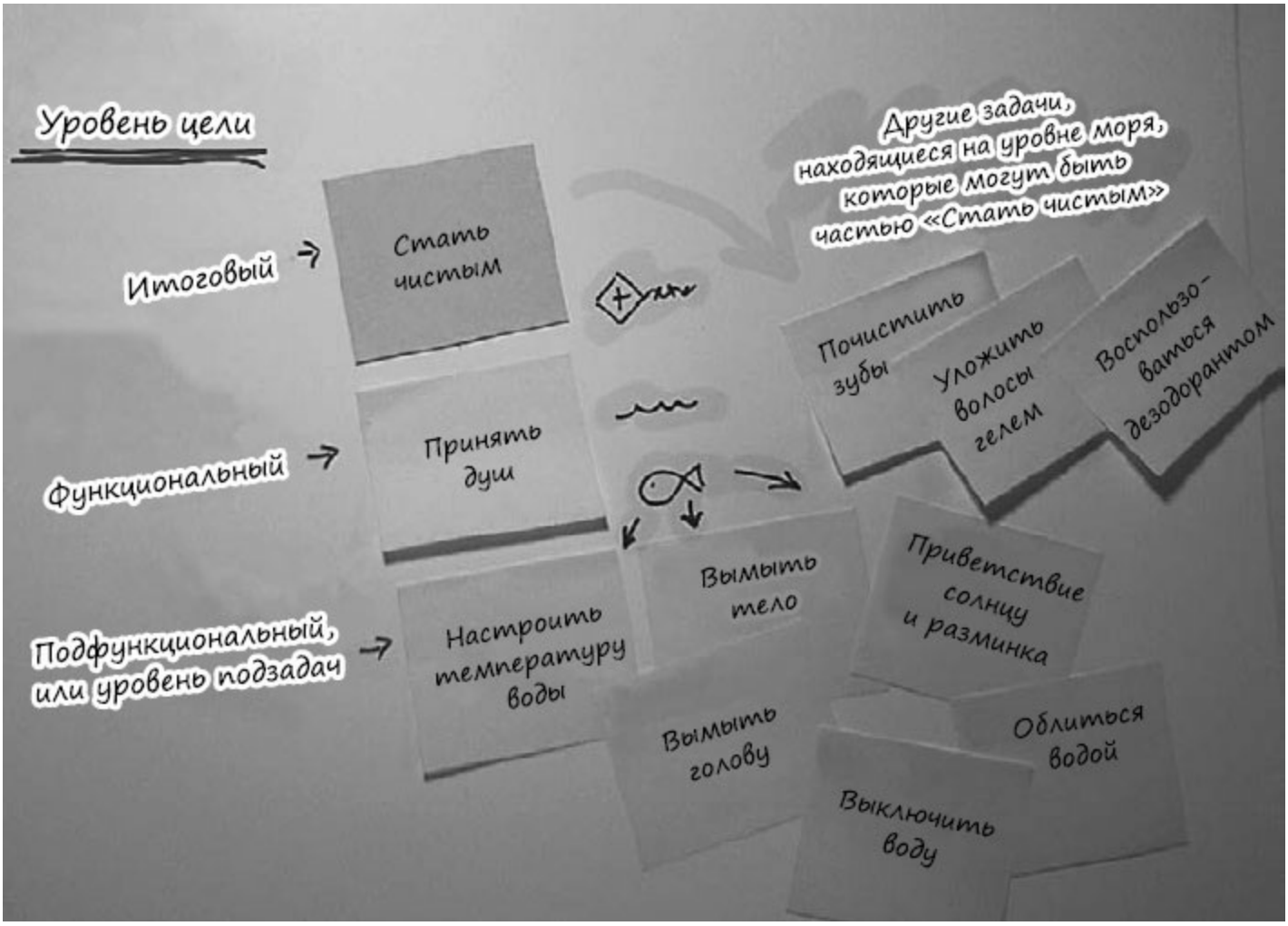

## Я ориентирован на детали

Как правило, выполняя это упражнение, некоторые люди записывают больше деталей, чем другие. Вместо краткого «Приготовил завтрак» они могут написать нечто вроде «Положил хлеб в тостер», «Налил сок в стакан» или, если это моя жена, «Добавила в смузи кале[12]», что создает некоторые проблемы в наших отношениях.

Задачи похожи на камни. Если вы возьмете большой камень и стукнете по нему молотком, он разобьется на кучку осколков, но эти осколки – такие же камни. То же самое происходит с задачами. Я не сумею определить, достаточно ли камень велик, чтобы считать его валуном, или достаточно мал, чтобы считать его галькой, но способ отличить крупные задачи от мелких существует.

Мой друг Алистер Коберн описал концепцию *целевых уровней* в книге «Написание эффективных пользовательских ситуаций» (*Cockburn Alistair,* Writing Effective Use Cases, издательство Addison-Wesley Professional). Не волнуйтесь, расписывать пользовательские ситуации мы сейчас не будем – это просто концепция, очень полезная при обсуждении поведения людей.

Алистер пользуется метафорой уровней высоты: уровень моря находится в середине, а все остальное – выше или ниже его. Задача на уровне моря – это то, что мы собираемся сделать, прежде чем намеренно остановимся и перейдем к чему-то еще. В ваших задачах наверняка есть «Принял душ». Это и есть задача на уровне моря – вы же не выключаете воду, думая: «Этот душ, похоже, затягивается, пойду-ка я выпью чашечку кофе, а потом закончу мыться». Алистер называет такие вещи *задачами функционального уровня* и помечает их значком океанской волны. Но я их называю просто задачами.




Задача «Принять душ» может быть разбита на множество маленьких подзадач вроде «Настроить температуру воды» или «Вымыть голову», а также, как у моей жены, чего-нибудь включающего надраивание мочалкой из люфы. Помните, что люди разные, и вы заметите различия в их поведении, касающиеся того, как они подходят к решению задач. Алистер помечает такие вещи маленькой рыбкой, потому что они находятся ниже уровня моря.

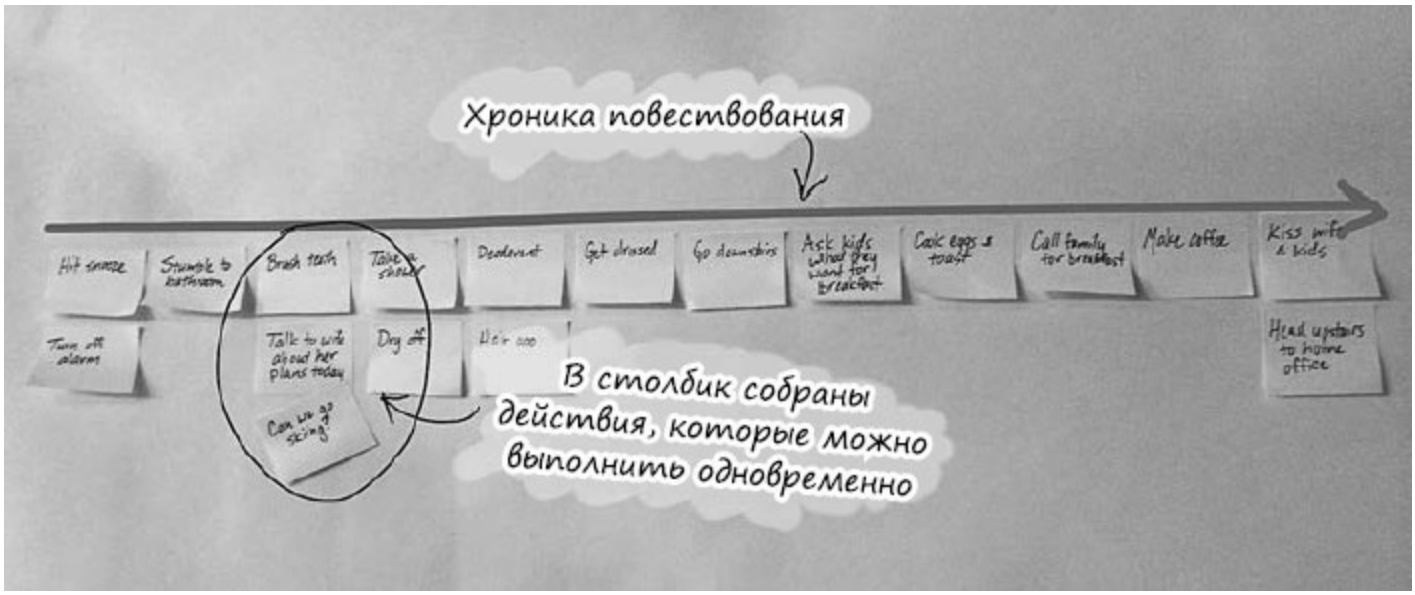
И наконец, мы переходим к ряду *задач итогового уровня*. Принять душ, побриться, почистить зубы и все остальное, что вы проделываете по утрам, поднявшись с постели, в конце концов сводятся к одной задаче-итогу. Я не уверен, как точно сформулировать ее название. «Стать чистым»? «Утренняя гигиена»? Мне не очень нравится слово «гигиена». Не будем его использовать.

# 2. Организуйте свою историю

Если вы этого еще не сделали, расположите свои задачи слева направо так, чтобы действия, которые вы выполняете первыми, располагались слева, а выполняемые последними – справа.

Попробуйте изложить историю, начиная с первого стикера: «Сначала я делаю это…», а затем перейдя к следующему со словами «а потом это». Продолжайте таким образом двигаться слева направо и рассказывать свою историю. Вы можете рассматривать каждый стикер как один шаг и переходить от одного шага к другому с помощью простой конструкции «а затем я…».

Порядок слева направо я называю *хроникой повествования*, но это просто красивое название порядка изложения истории слева направо. То, что у нас получилось, можно назвать *картой*, и хроника повествования – ее горизонтальная ось слева направо.




Оказывается, мое повествование получилось довольно длинным. Я также начал собирать вместе события, которые происходят примерно в одно и то же время. По мере выкладывания стикеров в последовательную историю я понял, что пропустил несколько деталей, и сейчас пытаюсь решить, стоят ли они того, чтобы их добавить.

*Карты организуются слева направо согласно хронике повествования – порядка, в котором вы рассказываете историю.*

**Добавьте пропущенные детали.** Приятная особенность составления карт с помощью стикеров заключается в том, что увидеть и оценить общую картину очень легко. А видя историю, организованную по порядку хроники повествования, вы обязательно заметите, что каких-то частей истории не хватает.

Посмотрите на свою карту и подумайте, не пропустили ли вы какие-нибудь шаги.

Я добавил всего несколько. Ниже уровня моря находится множество деталей, которыми я решил пренебречь: если попытаться записать их, получится несколько сотен стикеров.

# 3. Исследуйте альтернативные истории

Пока все довольно очевидно, правда? Знания, которые вы получили, вряд ли стоят использованных бумажек. Но подождите, сейчас будет кое-что интересное.

Задумайтесь на минуту о том, что вы делали *вчерашним* утром. Если найдутся какие-то действия, отличные от того, что вы делали сегодня утром, запишите их на стикерах и добавьте на карту.

Бывает, что утром все катится кувырком. Что вы будете делать, если у вас в доме внезапно отключат горячую воду? Если внезапно не окажется молока, или овсяных хлопьев, или чего-то еще, что вы привыкли есть на завтрак? Если дочка внезапно вспомнит о домашнем задании, которое нужно было обязательно сделать к сегодняшнему дню (у нас это происходит регулярно)? Что тогда? Запишите возможные действия и добавьте их на карту.

А сейчас подумайте об идеальном утре. Что бы сделало ваше утро безупречным? Мне, например, хотелось бы сделать небольшую зарядку, а затем не спеша позавтракать, читая что-то интересное. Но для этого пришлось бы вставать намного раньше и не переключать будильник.

Обратите внимание на то, что вы можете разместить некоторые задачи в столбик, во-первых, чтобы сэкономить место, а во-вторых, потому, что эти задачи аналогичны тем, которые вы решаете регулярно. Например, если вы добавили задачи по приготовлению изысканного завтрака, то можете их поместить в одну колонку с задачами приготовления быстрого завтрака, который едите обычно.


Мой друг Дэвид Хассман называет этот процесс игрой во «Что, если» – вы, наверное, помните эту фразу из глав 2 и 3. К сожалению, в эту игру можно играть бесконечно, в результате чего карта разрастется до невероятных размеров. Я добавил в свою карту только то, что когда-либо происходило на самом деле – я делал зарядку или позволял себе немного расслабиться за завтраком. Кроме того, добавил несколько распространенных вариантов развития событий, которые, как правило, могут наблюдаться по утрам.

*Основная часть карты заполняется деталями, альтернативами, вариантами и исключениями.*

**Сохраняйте порядок.** Наверное, вы обратили внимание на то, что, добавляя новые задачи, вам, скорее всего, пришлось переставить старые. Мне вот, например, пришлось: зарядку нужно было поместить между подъемом с кровати и утренним душем. Кроме того, еще раньше пришлось добавить «Надеть спортивную форму», ведь это не то же самое, что предусматривает задача «Одеться», стоящая после душа.

Если вы расслабитесь и интуитивно поставите все задачи на те места, где они выглядят наиболее естественно, то получите верную последовательность действий. Если попробуете рассказать свою историю сейчас, то сможете сделать это несколькими разными способами. Вы можете изложить историю *типичного* дня, историю *потрясающего* дня, а также историю дня, *когда вы встали не с той ноги*. В каждом из них по ходу продвижения слева направо будут использоваться разные стикеры. Попробуйте применить другие конструкции для объединения задач. Вы можете сказать: «Обычно я делаю это, а иногда – то» или «Я делаю это или это, а затем то» (вместо слов «это» или «то» подставьте реальные действия – я ведь не знаю, на что вы ссылаетесь в данный момент).

В моем детстве была популярна серия книг для детей под названием «Выбирай свое приключение». Может быть, и вы их помните. Идея книжек заключалась в том, что, дочитав до конца главы, вы получали набор вариантов действий, которые мог предпринять герой в продолжении истории. Каждому варианту соответствовал номер страницы. Сделав выбор, нужно было пролистать книгу до этой страницы и продолжать чтение с этого места. Мне эти книги не особенно нравились: независимо от того, что я выбирал, я всегда оказывался в одном и том же месте, да и вариантов по-настоящему крутого приключения было маловато. Карта работает почти так же, как эта книга, только лучше. У вас бесконечное количество способов прохождения карты, особенно если говорить об использовании вашего продукта реальными людьми для решения реальных задач.

Если вы хотите сделать это упражнение *по-настоящему* сложным и интересным, выполните его вместе с парой коллег. Уверяю: вы узнаете о них больше, чем когда-либо хотели, а кроме того, составлять хронику повествования, которая устроит каждого, будет очень интересно. Я имею в виду, вам будет интересно, если вы любитель подискутировать. Всегда найдутся те, кто сперва завтракает, а потом принимает душ, и те, кто поступает наоборот. Широкие возможности для спора открывает чистка зубов – вот вы, например, чистите зубы до или после завтрака? А может, и до и после?

Расслабьтесь.

То, о чем вы спорите, скорее всего, не имеет особого значения. К примеру, разместите вы завтрак до или после душа, зависит только от ваших предпочтений. Остановитесь на том, что привычно для большинства в вашей группе. О том, что *на самом деле важно*, люди, как вы убедитесь, не спорят. Не понадобится даже говорить о том, что задача «Одеться» должна находиться после душа – если, конечно, никто еще не приходил в ваш офис в мокром насквозь костюме.

# 4. Уберите всё лишнее, чтобы выделить каркас

К этому этапу ваша карта должна была здорово увеличиться в ширину, а если вы продумали некоторое количество вариантов – еще и в глубину. Скорее всего, в ней около 30 задач и выглядит она как скелет с ребрами, принадлежащий какому-то доисторическому животному.

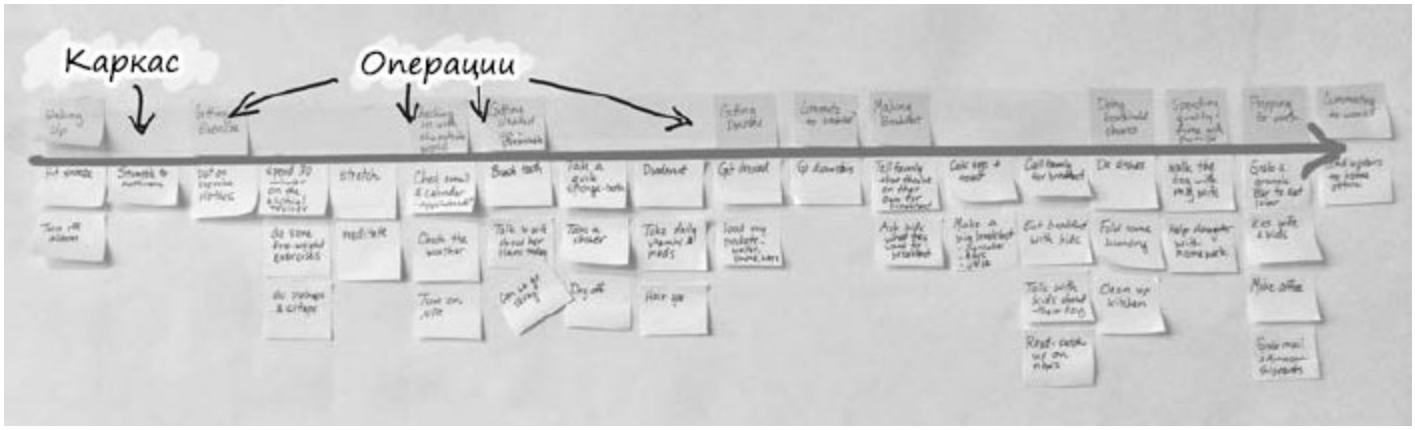
Если вы, сделав шаг назад, оглядите свою карту слева направо, то заметите, что там есть пачки задач, которые можно объединить между собой, – к примеру, все, что вы проделываете в ванной, чтобы собраться, или все операции по приготовлению завтрака на кухне, или другие мелкие действия: заглянуть в прогноз погоды, прихватить плащ, приготовить сумку с ноутбуком или другими вещами – словом, все, что вы делаете перед выходом из дома. Выделяются ли подобные действия, помогающие достичь более крупной цели, в вашей карте?

Если да, поместите над каждой пачкой стикеров с группирующимися действиями стикер другого цвета. Напишите на нем глагольную фразу, коротко формулирующую то, что объединяет задачи, находящиеся в этой пачке.

Если у вас не оказалось под рукой стикеров другого цвета, вот маленькая хитрость: на самом деле в каждой пачке стикеров находятся листки двух различных форм! Просто поверните стикер на 45°, и – оп! – он из квадрата превратится в ромб. Вы можете использовать этот простой способ всегда, когда нужно заставить стикер выглядеть иначе.

Стикеры, где записаны более высокоуровневые задачи, представляют на карте *операции*. Операции составлены из набора задач, которые похожие люди выполняют примерно в одно и то же время, чтобы достичь определенной цели. Если вы прочтете список операций, расположенный вдоль карты, то тоже получите хронику повествования. Строка с этими наклейками представляет собой каркас карты. Если у вас есть карта с множеством стикеров и вы хотите с кем-то ею поделиться, лучше всего для начала изложить высокоуровневую историю. Просто прочитайте каркас карты, вставляя связку «а потом они…» между соседними операциями.

*Операции объединяют задачи, которые направлены на одну и ту же цель.*




А вот и моя разросшаяся карта с добавленными операциями, составившими ее каркас. Таким образом карту проще читать и находить в ней нужные элементы, по крайней мере для

меня. Кроме того, теперь есть цельная картина того, что происходит со мной в течение утра.

*Операции и высокоуровневые задачи формируют каркас карты историй.*

Часто смысл операций, в отличие от задач, нелегко передать простым языком. Как, например, вы назовете тот набор мелочей, которые проделываете перед выходом из дома: приготовить сумку, захватить список покупок, заглянуть в прогноз погоды и при необходимости взять зонтик? Я, например, условно называю это «собраться перед выходом». Можете придумать собственное название.

Когда будете делать карту для своих продуктов, назовите операции так, как их называют ваши *заказчики*, которые пользуются этими продуктами.

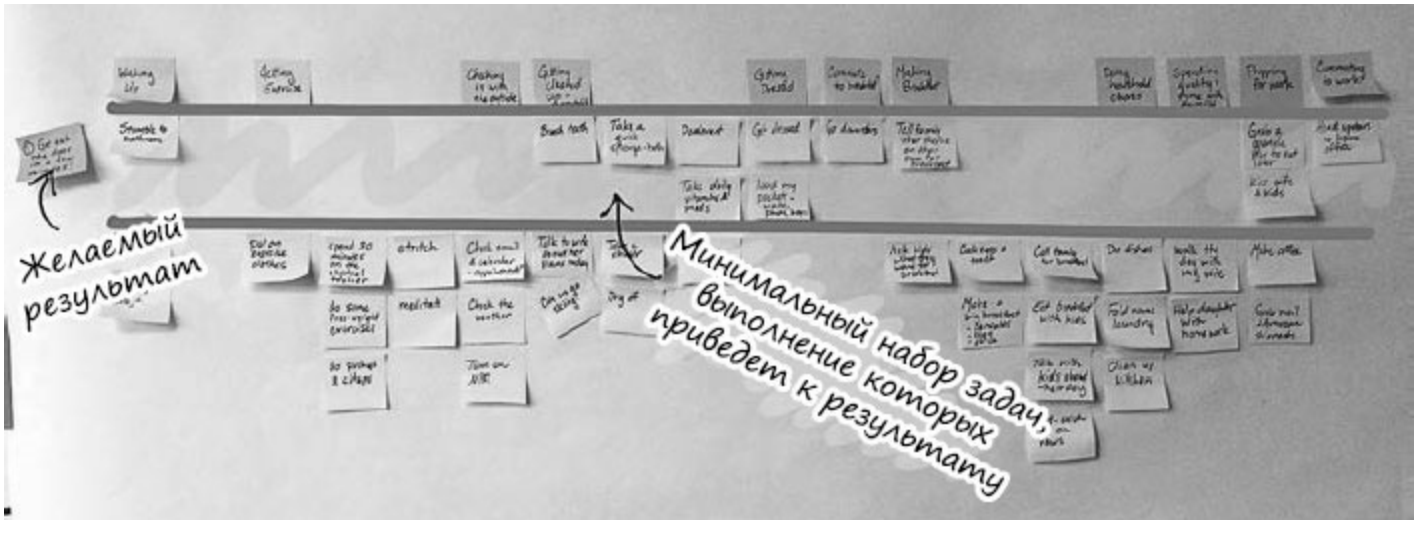
# 5. Выделите задачи, необходимые для достижения особенных целей

А сейчас перейдем к действительно интересному этапу – вы попытаетесь использовать карту для того, чтобы представить себе что-то, чего не случилось.

Например, в созданной вами карте в левом верхнем углу наверняка имеется нечто вроде «Прихлопнуть будильник» или «Переключить будильник». Представьте, что было бы, если бы сегодня утром вы пропустили этот шаг? Это могло произойти, к примеру, если вы забыли включить будильник накануне вечером. Вы открываете глаза, смотрите на часы и с ужасом понимаете, что вам нужно прибыть куда-то уже через несколько минут! Вы страшно опаздываете! Не нужно, впрочем, по-настоящему впадать в панику – пока все происходит только в воображении.

Запишите на листке «Приготовиться к выходу за пару минут» и приклейте его слева от карты чуть ниже верхней части. А сейчас представьте себе проходящую слева направо примерно посередине карты линию наподобие разделительной ленты. Переместите ниже этой линии все задачи, которые вы не будете выполнять, чтобы достичь указанной цели. Но не трогайте операции, даже если не осталось никаких связанных с ними задач. Наличие операции без задач позволяет вам понять, чего вы не собираетесь делать, чтобы достичь цели сегодня утром.

Скорее всего, в верхнем срезе останется всего несколько задач. Теперь снова пройдитесь по всему процессу и добавьте пропущенные задачи: что-то, что вы делаете, только когда сильно опаздываете. Например, если при нормальном ходе вещей вы принимаете душ, то в случае опоздания можете добавить задачи «Плеснуть водой в лицо» или «Протереть салфеткой самые грязные части тела». Кстати, когда это упражнение выполняют группы программистов, я часто вижу задачу «Побрызгать побольше дезодоранта». Я никого не осуждаю, если что – просто рассказываю.




Это моя карта с задачами, выполнив которые я за считаные минуты оказываюсь готов к выходу.

Можете опробовать этот прием на различных целях, которые могут оказаться с левой стороны. Например, «Провести утро мечты» или «Утро в середине двухнедельного отпуска». Как вы убедитесь, хроника повествования останется практически неизменной, но вам

придется добавлять или убирать задачи, чтобы обеспечить достижение различных целей.

*Используйте срезы, чтобы выделить все задачи и детали, соответствующие достижению определенного результата.*

# Вот и всё! Теперь вы знаете всё, что важно

Было ведь несложно, правда? По мере построения карты вы должны были усвоить следующее.

*Задачи* – это короткие глагольные фразы, описывающие то, что делают люди.

• Всем задачам соответствуют разные *уровни целей.*

• На карте задачи образуют хронику, изложенную *слева направо.*

*Глубину* карты создают различные варианты и альтернативные задачи.

• Задачи образуют *операции* на протяжении всей карты.

• Операции формируют *каркас карты*.

• Вы можете выделить на карте *срезы*, куда войдут задачи, необходимые для достижения какого-то особого результата.

# На самом деле попробуйте – дома или на работе

Не думайте, что вам удалось меня провести. Я прекрасно знаю: большая часть из вас просто прочитали эту главу, а составлять карту даже не пробовали. Если вас тоже одолела лень и вы не составили карту своего утра, обещайте мне, что сделаете это. Это мой самый любимый и очень простой способ обучения составлению карт. Если вы хотите опробовать методику составления карт в своей организации, соберите небольшую группу людей и попробуйте выполнить упражнение вместе – таким образом вы изучите основные понятия. А кроме того, заложите основы умения составлять карту *чего угодно*.

### *А ты принимаешь душ перед работой?*

*Рик Кьюзик, Reading Plus, Winooski, Vermont*

Мы проделали упражнение с утренней картой в таком составе: четверо программистов, владелец продукта, тестировщик, начальник группы UX, а также два менеджера. Разделившись на две команды, мы быстро выяснили, что включает в себя утро каждого, а затем многократно тасовали карточки, пока не пришли к усредненному утру. Всем очень понравилась работа по составлению карты, несмотря на то что раньше они никогда не делали ничего подобного или рассматривали этот процесс лишь как работу владельца продукта.

С моей точки зрения, целями этого упражнения было показать эффективность визуализации нашей работы, продемонстрировать, как составление карты помогает развитию одинакового понимания, а также доказать полезность видения пользовательского опыта в удобном формате. Кроме того, обнаружился такой приятный побочный эффект, как сплочение команды, – при совместной работе над проектом это получается само собой, работа вызывает интерес и усиливает чувство сопереживания друг другу. «Я и не знал, что перед работой тебе приходится отвозить детей в школу!» «Ты каждое утро занимаешься йогой?» «А я не могу выйти из дома без завтрака, от меня просто не будет никакого толку!»

Некоторые затруднения вызвали случаи, когда несколько событий происходили

одновременно или одно событие зависело от другого: «Если я пью кофе и читаю газету – это один стикер или два?» Или: «По пятницам детей в школу отвозит жена, как я должен это отразить в истории?» Кроме того, учитывая линейный характер времени, на карте, располагающейся слева направо, оказалось затруднительно отразить все варианты – это также привело к замешательству. Как координатору мне было приятно видеть, что процесс размышления идет прямо во время упражнения, даже если у меня не было готовых ответов на эти вопросы.

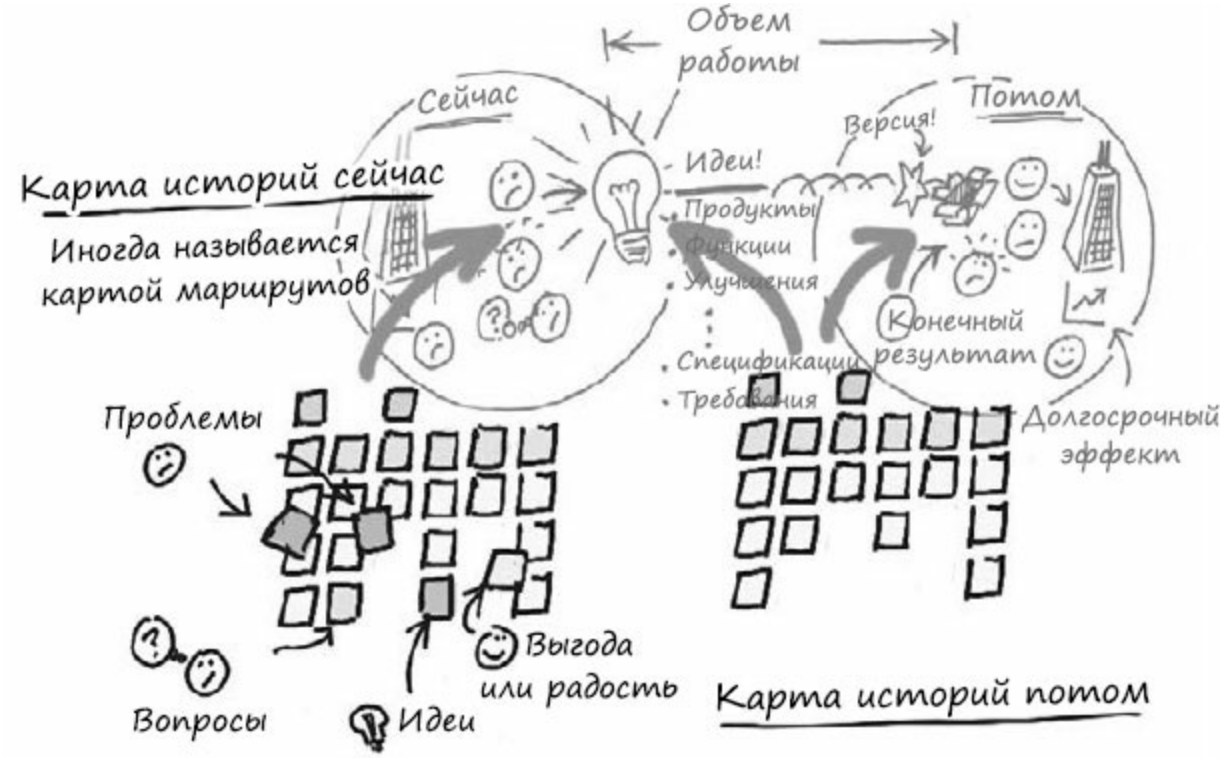
После того как мы расставили операции по приоритету, принятие некоторых сложных решений привело к комическим ситуациям. Звучали, например, такие «серьезные» вопросы: «А что, душ перед работой *действительно* необходим?» «Независимо от того, что еще мы выбросим, нужно просто подняться с кровати, одеться и поехать на работу, – рассуждал один из участников. – Хотя… можно ведь просто работать из дома!»

Вскоре после выполнения этого упражнения карты историй стали нашим любимым способом обсуждать пользовательский опыт взаимодействия, расставлять пользовательские истории по приоритету, а также планировать итерации и релизы. Эта практика прочно вошла в культуру разработки в компании и в профессиональный жаргон, и мы пользуемся ею до сих пор.

Один из уроков, которые я усвоил, выполнив это упражнение несколько раз со многими командами в нашей организации, – необходимо замечание, снимающее скованность, какое-то средство, способное запустить мыслительный процесс у участников. Обычно в начале сессии я прошу каждого участника записать что-то одно, что он или она делает по утрам между пробуждением и приходом на работу. Потом я спрашиваю каждого, почему именно он (-а) делает это. Как я убедился, это закладывает основу мыслительного процесса, который позднее проявляется в сессиях по планированию: «В чем ценность этой истории? Что заставляет наших пользователей это делать?»

# На карте настоящее, а не будущее

Думаю, что для многих из вас это очевидно, но на всякий случай подчеркну: карта, которую мы только что составили, имеет фундаментальное отличие от карт, упомянутых в первых четырех главах. Карты Гэри, Globo.com, Эрика и Майка с Аароном представляют собой воображаемые способы работы пользователей с продуктом *в будущем*, после того как его выпустят. Они записывали задачи и операции, которые, по их предположению, пользователи будут выполнять, работая с продуктом. А карта, которую вы создали, касается ваших *теперешних* задач – тех, которые вы выполняли этим утром. И оказывается, что основа и концепция одинаковы в обоих случаях. Поэтому будьте уверены: вы потратили время не зря.




Одно из самых крутых преимуществ составления карт историй настоящего заключается в том, что, построив их, вы гораздо лучше осознаете, как люди работают сейчас. Вот вы только что тщательно изучили, как собираетесь на работу по утрам. А можете открыть для себя еще больше, если добавите на карту кое-что еще. Это могут быть, например:

• *проблемы* – что-то, что не работает, вызывает трудности, что люди ненавидят;

• *выгоды или радости* – то, что доставляет удовольствие и вдохновляет;

• *вопросы* – «Почему люди делают это?», «Что происходит, если они делают это?»;

• *идеи* – то, что люди могли бы сделать, или возможные способы устранить проблему либо увеличить эффекты удовольствия и радости.

Этот метод уже довольно давно используют многие специалисты UX, чтобы лучше понять своих пользователей. Иногда они называют его *картами взаимодействия*, но основная идея абсолютно та же.

# Испробуйте метод в реальности

В начале 2000-х я руководил небольшой компанией по разработке продуктов под названием Tomax. Мы работали над программой для розничных магазинов – тех самых, куда ходили за покупками, прежде чем с головой погрузились в Интернет. У нас появился новый клиент – крупная сеть магазинов лакокрасочной продукции и предметов интерьера. Мы немного разбирались в розничной торговле, а также в людях, которые обслуживают покупателей в магазине и работают с оборудованием, но совсем ничего не знали о многих вещах – в основном о деталях, характерных для торговли красками и предметами интерьера. Например, нам были неизвестны особенности продажи краски индивидуальных оттенков или жалюзи на заказ. А учиться нужно было быстро.

Чтобы ускорить процесс, мы попросили этих трех дам нам помочь. Они не имели никакого отношения к разработке программного обеспечения – они дизайнеры интерьеров, работающие в компании, которая хотела заказать у нас программу. С их помощью мы от начала до конца изучили процесс продажи жалюзи на заказ. Мы попросили их записать все, что они обычно делают, от появления клиента до установки готовых жалюзи к его удовольствию. Вероятно, вы уже догадались: мы попросили их сделать то же самое, что делали вы, составляя карту своего утра. Все происходило точно так же. Они легко называли все, что делали в процессе продажи жалюзи, так же, как вы перечисляли свои утренние действия при сборах на работу. А когда мы расположили их задачи на карте, оказалось, что существует не единственный способ достижения цели: все трое выполняли задачи немного по-разному или в разном порядке. Вы заметите то же самое, если попробуете составить карту сборов на работу вместе с небольшой группой разных людей.

После выполнения несложного упражнения по составлению историй и организации их в карту все мы сформировали единое понимание того, как происходит процесс *сейчас*. Теперь можно было приступать к преобразованию этой карты в программное обеспечение, где точно такой же процесс должен был происходить *в будущем*.

# С программами труднее

Я не хочу вас обманывать. Если вы профессионал в области разработки программного обеспечения, возможно, вам потребуется некоторое время, чтобы перейти от обсуждения экранов и функций к записи коротких глагольных фраз, которые формулируют то, что пытаются в реальности проделать люди. Продолжайте практиковаться. У вас получится.

Но если вы не знаете, кто ваши пользователи, чего они хотят достичь и как с этим справляются, будет гораздо труднее. К сожалению, попытка построить карту в этих условиях приведет лишь к выявлению того, что вам неизвестно. Если вы оказались в такой ситуации, лучшее, что вы можете сделать, – это изучить пользователей, их цели и задачи. Еще лучше, если вам удастся напрямую поработать с ними и построить карту вместе.

## *Шесть простых шагов к составлению карты историй*

Я хочу свести главное из предыдущих четырех глав в шесть простых шагов. Вы можете задаться вопросом: «Почему он не сделал этого с самого начала?» Что ж, я мог бы пропустить все эти рассуждения и истории и просто перечислить требования. Но толку от этого не было бы никакого.

Мне известно множество работающих способов построения и использования карты историй, поэтому я вывел такой шестишаговый процесс, лучше всего подходящий для меня.

1. Сформулируйте проблему: для кого вы это делаете, зачем вы это делаете?
2. Обрисуйте общую картину. Концентрируйтесь на широте, а не на глубине, продвигайтесь на милю вперед и на дюйм вглубь (или на километр вперед и на сантиметр вглубь – для моих неамериканских друзей). Если у вас нет готового решения или даже если, как вам кажется, оно есть, попробуйте построить карту по состоянию на сегодня, включая проблемы и преимущества, которые отмечаются у ваших пользователей.
3. Исследуйте. Углубляйтесь и обсудите другие типы пользователей и то, как еще они могут выполнять свои задачи, а также варианты их действий, если (или, скорее, когда) что-то пойдет не так. Для пущей уверенности нарисуйте эскизы, составьте прототипы, протестируйте их, усовершенствуйте идеи решений, по ходу дела меняя и расширяя карту.
4. Выделите релизную стратегию. Помните: времени и ресурсов всегда не хватает. Концентрируйтесь на том, чего хочет достичь ваш бизнес, а также на людях, для которых создаете продукт. Выбросите вон все, что не нужно для реализации минимальных решений, которые, с одной стороны, удовлетворят нужды людей, с другой – помогут вашей организации достичь ее целей.
5. Выделите исследовательскую стратегию. Можете сформулировать то, что, по вашему мнению, является минимально жизнеспособным решением, но не забывайте, что это только гипотеза, пока не убедитесь в обратном. Используйте карту и обсуждения, чтобы выявить самые рискованные элементы. Разделите карту на совсем маленькие срезы минимально жизнеспособных продуктов-

экспериментов. Вы сможете показать их группе пользователей и выяснить, что на самом деле для них полезно.

6. Выделите стратегию разработки. Если вы отбросите все, что *не должны* предъявить пользователям, останется то, что *должны*. Теперь разделите минимально жизнеспособное решение на части с точки зрения последовательности разработки. Сконцентрируйтесь на реализации пораньше – так вы быстрее обнаружите технические проблемы и риски разработки.

# Карта только начало

Построение карты помогает вам охватить взглядом общую картину, иными словами, увидеть лес за деревьями. Это одно из самых больших преимуществ построения карты историй. Но если вы один из ответственных за посадку леса, у вас нет иного пути, кроме как сажать одно дерево за другим. Вы уже знаете два важных правила работы с картами историй.

• При изложении историй используйте как слова, так и рисунки, для того чтобы выработалось одинаковое понимание.

• Не говорите только о том, что нужно разработать, – обсуждайте, кто будет использовать продукт и почему таким образом вам удастся минимизировать объем работы и обеспечить максимальный реальный результат.

Держите это в уме, и все мало-помалу получится.

Мы уже обсудили несколько тактик использования историй, чтобы избежать ошибок. Поговорим еще о нескольких моментах, которые помогут вам правильно применять истории.

**Карты пользовательских историй в SAP – суть в масштабировании**

*Андреа Шмиден*

Когда Джефф впервые представил свою концепцию карт пользовательских историй, мы сразу увидели, какую пользу она может принести SAP. Создалось впечатление, что этот простой, но мощный метод может помочь трансформировать видение продукта в бэклог, а также разобраться, что мы разрабатываем, для кого и зачем. Поэтому мы решили его опробовать.

Однако, как мы вскоре убедились, то, что легко и просто для одиночного разработчика или отдельной команды, работающей по Scrum, совершенно иначе выглядит для группы по разработке продукта, состоящей из *нескольких* Scrum-команд. В SAP, огромной организации, где работают около 20 000 разработчиков, привычны огромные команды, работа которых зависит от других команд. Здесь это норма, а не исключение. Нам нужен был работающий способ масштабировать карты пользовательских историй для такой большой организации.

**Задача**

Сложная задача, стоящая перед нами, включала в себя два аспекта.

• Как составить карты сложных продуктов и не утонуть в куче стикеров?

• Как внедрить метод в организации по разработке и научить людей его использовать?

**1. Карты пользовательских историй для крупных продуктов**

В поиске ответа на первый вопрос мы решили, что самым лучшим будет провести несколько пилотных семинаров по проработке существующих проектов. Начали с небольшой команды самых заинтересованных сотрудников и приблизительно десяти пилотных проектов, над самым большим из которых работали 14 Scrum-команд. В этой пилотной фазе мы опробовали метод по-разному в нескольких видах: формат проведения семинара, содержание, фазы проекта, формат карты и т. д. Проведя несколько раундов с получением обратной связи и определения итераций, мы выделили набор хороших практик, которые, как оказалось, неплохо работают при разработке проектов большого размера.

**Ключевые хорошие практики**

Если команда пробует использовать карты историй в первый раз, мы рекомендуем привлечь к делу опытного тренера. На встрече с приглашающей стороной тренер обсуждает цели семинара, приглашенных, повестку дня, соответствующие вводные данные и т. д. Обычно он проводит однодневный семинар со всей командой, а затем при необходимости мы устраиваем более короткие сессии.

Работу в день семинара мы, как правило, начинаем с упражнения по видению продукта вроде хорошо известных «речи в лифте» или «темы номера», когда члены команды описывают, что бы они хотели прочитать о своем продукте в статье отраслевого журнала через год. Таким образом становится хорошо видно, есть ли у команды общее понимание основного направления работы, или необходимы дополнительные исследования (например, дополнительные интервью, создание прототипов, тестирование прототипов и т. д.).

На следующем шаге в фокусе оказываются типичные пользователи продукта. Если цель семинара – получить детализированный бэклог, пользовательские роли или персонажи должны быть получены на стадии пользовательского исследования. Если же проект находится на ранней фазе, команда записывает свои предположения, которые потом будут проверены при исследовании пользователей. Мы уже убедились в том, что это отличная подготовительная работа для исследования, – вот еще один аспект, когда дизайнерское мышление и практики составления карт пользовательских историй отлично дополняют друг друга.

После этого переходим к составлению пользовательских историй, последовательно проходя три уровня.

1. Сначала определяются высокоуровневые последовательности шагов по использованию продукта.

2. Затем последовательности шагов разбиваются на более четкие операции согласно пользовательским ролям.

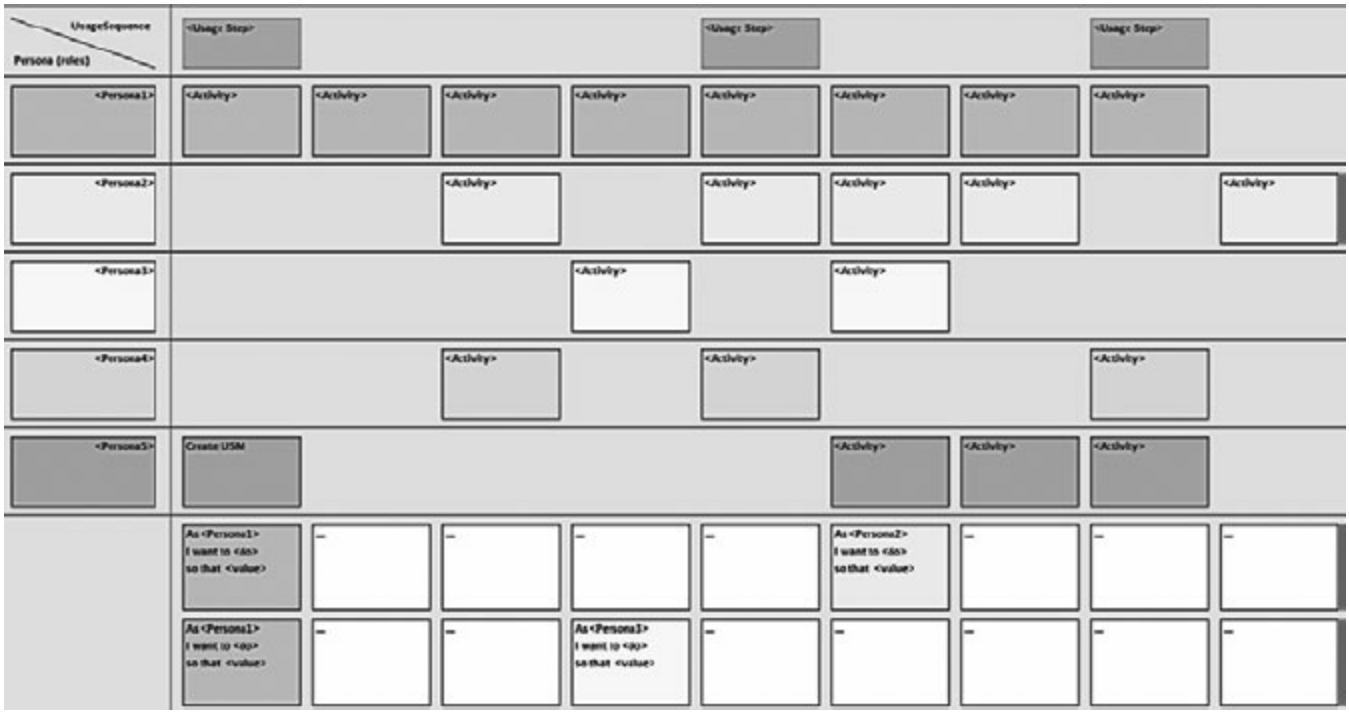
3. Из пользовательских ролей выделяются конкретные пользовательские истории в формате «как *<роль>*, я хочу *<функциональность>*, что даст мне *<выгоду>*». Эти пользовательские истории включаются в первый вариант продуктового бэклога.

Такой трехуровневый процесс особенно полезен для больших продуктов. На каждом уровне команда может решить, когда имеет смысл погрузиться в обсуждение деталей и где следует рассмотреть зависимости от других команд. Этот подход позволяет сконцентрироваться на ключевых задачах разработки, держа в уме общую картину.

Чтобы читать карту было проще, мы используем стикеры определенных цветов для операций, а также для пользовательских историй, относящихся к определенным персонажу или роли, как вы можете видеть на графике.

Часто во время работы команды над картой выявляются дополнительные аспекты, так называемые белые пятна – области, где нужно выполнить больше исследований, или вопросы без ответов, зависимости, пробелы. Чтобы выделить эти проблемы, мы используем стикеры определенного цвета или размера. На первый взгляд может показаться, что помещать открытые проблемы на карту

неудобно, но, как мы убедились по опыту, это один из самых полезных аспектов в процессе составления карт историй: вы получаете ясное и осязаемое представление элементов, которые требуют дальнейшего уточнения. После того как все проблемы оказываются на доске, обрабатывать их становится намного проще.




Как только команда обработала все детали (в разумной степени), мы расставляем пользовательские истории в бэклоге по приоритету. В зависимости от размера проекта и его фазы иногда это делается скорее на уровне пользовательских операций, а не пользовательских историй. Обычно мы пользуемся простыми техниками голосования, например точечной техникой, но иногда применяем для голосования упрощенную модель Кано, согласно которой команды помечают пользовательские истории как «обязательно», «желательно» и «опционально». Результаты этого простого голосования также являются хорошей основой для окончательной расстановки историй с участием других заинтересованных сторон, конечных пользователей и заказчиков.

Как прокомментировал один из владельцев продукта, «владельцы продукта часто сталкиваются со сложной задачей – им необходимо поместить множество требований в очень узкие временные рамки. Мы пригласили наших заказчиков на однодневный семинар по картам историй, и оказалось, что это очень эффективный и продуктивный способ выработки одинакового понимания наших приоритетов».

Детали, подробные оценки затрат и другие мелочи обычно не входят в семинар, но, как правило, обсуждаются в меньших группах позднее.

**2. Масштабирование карт пользовательских историй**

Чтобы масштабировать и развернуть процесс, команда тренеров, которая начинала работу, приготовила материалы: шаблоны карт, сделанные в программе Excel, шаблоны для персонажей, стандартную повестку дня семинара, выдержки из

документации, а также шпаргалки с описанием метода. Кроме этого, было разработано собственное внутреннее приложение для составления карт пользовательских историй.

Но материалы – это одно, а проведение семинара – совершенно другое. Поэтому мы снова настоятельно рекомендуем привлечь к процессу опытного тренера. Чтобы обеспечить организацию достаточным количеством тренеров, первичная тренерская команда подготовила специалистов. Эти начинающие тренеры провели несколько семинаров под присмотром наставников, помогали на индивидуальных сессиях и наконец были готовы руководить семинаром абсолютно самостоятельно. Мы также проводили семинары и тренировки тренеров в других офисах SAP по всему миру. Чтобы обрести уверенность, что мы учимся друг у друга, и для обмена опытом мы запустили огромную сеть для хранения документации и общения, где тренеры могли опубликовать свои вопросы и полезные практики. И конечно, очень многое мы почерпнули из опыта Джеффа.

Наши усилия по масштабированию пользовательских карт историй в конце концов принесли плоды. Мы провели более 200 семинаров под руководством тренеров в различных офисах во многих местах, и сейчас большинство команд способны продуктивно работать с картами пользовательских историй самостоятельно.

# Глава 6. Реальные примеры применения историй

Карты историй – невероятно простая идея. Применяя простую карту, вы можете работать с другими людьми, чтобы изложить историю использования продукта и увидеть результат своей работы в целом. Затем вы дробите эту цельную картину на части, чтобы правильно распланировать разработку. Основу всего составляет простая концепция историй в Agile.

# Убийственно простая идея Кента

Изначальная идея историй принадлежит невероятно умному парню по имени Кент Бек. В конце 1990-х Кент и его коллеги работали в области разработки программного обеспечения. Кент обратил внимание на то, что одна из крупнейших проблем в разработке программ берет свое начало из традиционной работы с бумажными документами, где подробно описано, чего мы хотим, – их принято называть требованиями. К настоящему моменту вы уже знаете, что с ними не так: разные люди могут читать один и тот же документ и понимать его совершенно по-разному. Они могут даже подписаться под этим документом, свидетельствуя, что пришли к соглашению.

Лишь позднее, когда мы погружаемся в глубину разработки программы – или даже еще немного позднее, когда она предъявлена пользователям, – выясняется, что на самом деле мы думали не об одном и том же. И конечно, большинство людей объясняют отсутствие общего понимания неудачными требованиями.

Давайте задержимся здесь на минуту. Я имел удовольствие работать с множеством команд. Очень часто работа начиналась с обсуждения наших самых больших затруднений. И разумеется, чаще всего я первым делом узнавал о «неудачных требованиях». Каждый тычет пальцем в этот несчастный документ. Автор документа готов сгореть со стыда – наверняка он должен был написать больше или меньше либо, может быть, использовать какие-то новейшие техники составления документации. Те, кто подписал этот документ, ощущают себя не лучше и поэтому занимают оборонительную позицию: «Вы что, ожидали, что я вчитаюсь в каждую деталь? В конце концов мы обсуждали это целыми днями! Я думал, вы поняли, что я сказал. Я вообще не понимаю, откуда взялся этот дурацкий документ!» А те, кто работал над самим программным продуктом, и вовсе сбиты с толку – они выполнили свою задачу, руководствуясь этим злосчастным документом, но оказалось, что все по-прежнему неправильно. В общем, все хотят отправить эти бумаги в печку и немедленно написать новую, лучшую документацию.

> *Мы оба можем прочитать один и тот же документ, но понять его по-разному.*


Но разное понимание документа – это лишь половина проблемы. Мы тратим уйму денег и времени, воплощая в жизнь то, что описывает этот документ, и все только для того, чтобы позднее обнаружить: для решения изначально поставленной задачи нужно совершенно иное. Да, вы поняли меня правильно – все эти документы, как правило, содержат совершенно неверные вещи. Документы обычно описывают то, *что* нам нужно, но не *почему* оно нужно. Если некто работающий над программным продуктом может просто поговорить с кем-либо, кто хорошо разбирается в будущих пользователях этого продукта и мотивах их работы с продуктом, чаще всего он найдет гораздо более эффективный и экономичный способ удовлетворить этих пользователей. Не нужно уточнять, что чаще всего у нас попросту нет такой информации.

> *Лучшие решения – результат сотрудничества между людьми, у которых есть какие-то проблемы, и другими людьми, которые могут их решить.*

Простая идея Кента заключалась в том, чтобы *прекратить* это – перестать тратить силы на написание идеальной документации, а вместо этого собираться вместе и *рассказывать истории*. Истории получили свое название не потому, что их нужно было записывать, а из-за того, *как* они должны были использоваться. Разрешите мне подчеркнуть это еще раз для закрепления. Прямо сейчас отложите все дела и громко прочитайте вслух.

> *Название «истории» произошло от того, как они должны использоваться, а не из-за того, что вы должны их записывать.*

Идея Кента очень проста. Если мы соберемся вместе и обсудим проблему, которую решаем с помощью программного продукта, а также поговорим о том, для кого и почему мы это делаем, то в конце концов придем к решению и выстроим одинаковое понимание.

# Просто – не значит легко

Некоторое время назад я стал замечать, что развитие работы с историями ушло немного в сторону: множество людей пишут книги, преподают и используют эту технику, концентрируясь на написании историй. Если бы я получал десять центов каждый раз, когда меня спрашивают, как писать хорошие истории, десятицентовиков у меня было бы больше, чем в той куче, которую я упомянул несколькими главами ранее.

Поскольку все вокруг говорили только о написании историй, мне пришлось задать Кенту вопрос: не пропустил ли я чего? Во время длинного обсуждения по электронной почте Кент объяснил мне происхождение идеи: *«Я хотел развить известную ситуацию, когда пользователи сами излагают истории работы крутых функций, которые умеет выполнять их рабочая программа. [Например,] когда я впечатываю в поле почтовый индекс, а программа автоматически заполняет город и штат, мне не приходится нажимать ни одной кнопки.*

*Я думаю, именно этот пример натолкнул меня на мысль. Если вы можете рассказывать истории о том, что делает программа, вызывая интерес и живой отклик у слушателя, почему же не рассказать эти истории еще до того, как программа сможет это делать?»* (Кент Бек в личной переписке, август 2010 года).

Как видите, идея – в рассказе, и вы сами знаете: если рассказ правильный, то у слушателей пробуждаются интерес и энергия, предмет рассказа предстает перед ними как живой. Не правда ли, здорово? Да и звучит намного интереснее, чем просто чтение типичного документа с требованиями[13].

Однако многие из тех, кто начинает использовать истории в разработке программного обеспечения, но все еще подразумевает традиционную модель процесса, стараются больше фокусироваться на этапе написания. Мне случалось видеть команды, которые заменили написанием историй традиционный процесс составления документов с требованиями и, конечно, впадали в бешенство, стараясь с помощью историй точно сформулировать, что должно быть реализовано в программе. Если сейчас то же самое происходит у вас, остановитесь.

> *Если вы не собираетесь вместе, чтобы всесторонне обсудить свои истории, то на самом деле вы не используете истории.*

# Рон Джеффрис и три «П»

В книге «Утвержденное экстремальное программирование»[14] Рон Джеффрис с соавторами описывают, как работать с историями наилучшим образом.

**Пишем.** Запишите то, что вы хотели бы видеть в программном продукте, на нескольких карточках.

**Проговариваем.** Соберитесь вместе и всесторонне обсудите программный продукт, который создаете, и все, что будет в нем реализовано.

**Подтверждаем.** Договоритесь о способе подтвердить то, что программный продукт готов.


## 1. Пишем

Допустим, вы ответственны за работу команды, которая получила задание создать какой-то программный продукт. Представьте этот продукт как можно яснее. Затем запишите на карточке каждое действие, которое пользователи могли бы проделать с применением вашего продукта. В результате вы получите стопку карточек. Сначала Кент предлагал делать записи на картотечных карточках, потому что проще расположить их на столе, расставить в соответствии с приоритетом или составить из них структуру, которая позволит увидеть общую картину – это, разумеется, карта историй.

Набор карточек, который целиком описывает продукт или все изменения, которые мы хотим внести в уже существующий продукт, называется *продуктовым бэклогом*. Этот термин произошел из процесса Scrum в методологии Agile. Один мой знакомый однажды признался: «Я ненавижу термин *“бэклог”*. Это слово звучит так, будто мы, даже не начав работать над программным продуктом, уже все завалили!»[15] У меня, к сожалению, нет лучшего названия для этой пачки историй, но если вы можете придумать что-то более позитивное, пожалуйста, используйте свой термин и напишите мне об этом.

## 2. Проговариваем

Обсуждение начинается, когда вы описываете свои идеи и задумки, а слушатели осмысливают их, основываясь на том, что слышат. Зачастую дать идеальное описание нелегко, а вот вообразить на основе услышанного нечто основанное на собственном прошлом опыте легче легкого, поэтому слушатель чаще всего представляет себе что-то совершенно отличное от того, что подразумеваете вы. Но все-таки волшебство происходит именно сейчас.

Как всегда при обсуждениях, слушатели могут задавать вопросы. Эти уточнения скорректируют их восприятие и помогут в конце концов прийти к верному единому пониманию.

В традиционном процессе разработки программ человек, составляющий требования, обычно хочет лишь корректно записать их на бумаге, а тот, кто собирается приступить к разработке программного продукта, должен их правильно понять. А если процесс управляется историями, у всех есть совершенно другая общая цель. Ваша цель – работать вместе, чтобы разобраться в проблеме, от которой можно избавиться с помощью программного продукта, а затем решить ее наилучшим из возможных способов. Разумеется, вы все должны договориться, что именно будете создавать и что, по вашему мнению, наиболее полезно тем, кто станет работать с продуктом.

Разрешите мне повторить это важное утверждение.

> *Обсуждать истории нужно для того, чтобы, работая вместе, найти наилучшее решение проблемы, которую мы все понимаем одинаково.*

## 3. Подтверждаем

Все эти разговоры хороши, но все-таки рано или поздно надо начать программировать, верно? Что ж, когда мы, кажется, нашли правильное решение, нужно сконцентрироваться на следующих вопросах.

*Если мы реализуем то, что мы все согласились делать, как проверить готовность и правильность полученного продукта?*

Ответ на этот вопрос обычно представляет собой короткий список проверок. Его еще иногда называют *критериями приемки* или *тестами историй*.

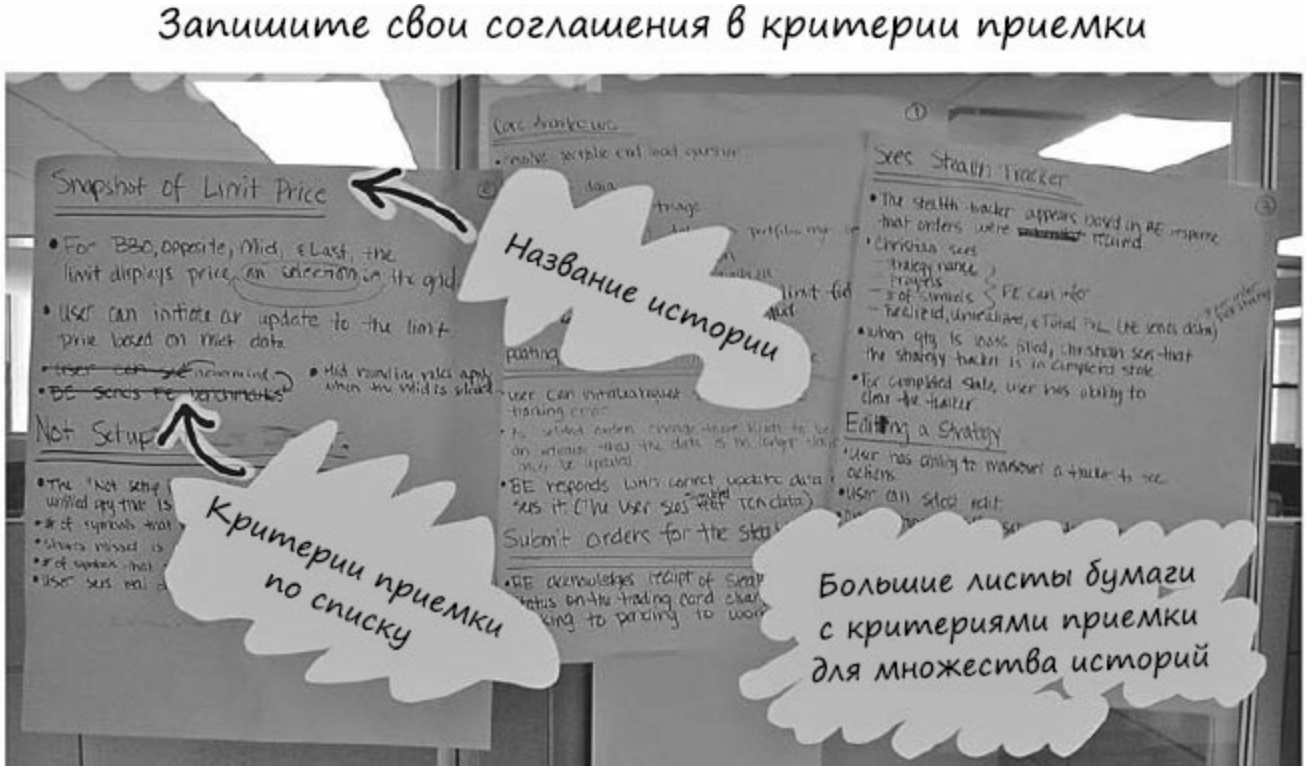
*Когда придет время показать результат нашей работы на демонстрации продукта, как это сделать?*

Чаще всего ответ на этот вопрос выявляет несколько пробелов в планировании. Например, вы можете закончить разработку, но для демонстрации нужны какие-то реальные данные. Обсуждение демонстрации может добавить еще несколько пунктов в список критериев приемки.

# Слова и картинки

На пути к соглашению недостаточно вооружиться одной карточкой и много размахивать руками. Обсуждение идет лучше всего, если у вас есть множество подручных средств, например простые изображения персонажей, диаграммы рабочего процесса, наброски интерфейса и прочее, что обычно используется в традиционном подходе к разработке ПО для того, чтобы лучше объяснить что-то. Таким образом, вы не будете просто размахивать руками, а конкретно укажете на предмет обсуждения. Что бы ни стало предметом обсуждения, это всегда можно пометить, записать, скорректировать или изменить. Мы можем создать множество разных вещей прямо на ходу, во время обсуждения. Активно используйте доску или большие листы бумаги. Не забудьте также сделать несколько «фото из отпуска» перед тем, как команда разойдется. Снимки того, что вы сделали, помогут вам припомнить все детали обсуждения, записать которые на бумаге нелегко.


Хорошее обсуждение историй всегда требует множества слов и картинок.




Во время обсуждения держите перед глазами критерии приемки, к которым вы пришли. Эта команда использует большие листы бумаги для записи хода обсуждения.

# Вот и всё

Собственно, на этом все. В этом и заключалась убийственно простая идея Кента. Но если вы воплотите ее в жизнь, обещаю, вам удастся изменить многое.

А если нет?

У людей, привыкших к традиционным способам, обсуждение может пойти по-настоящему плохо. Они мыслят стереотипно, ожидая, что один человек будет старательно излагать какую-то информацию, пытаясь как можно точнее донести до других людей, чего он от них хочет, а остальные будут внимательно слушать и затем обнаружат бреши в том, что он говорил. Если они слишком долго общаются таким образом, то в конце концов все начинают чувствовать себя так, будто в них самих пробили дыры, что никак не способствует успешной реализации намеченных планов.

Есть несколько полезных приемов, о которых нужно помнить, чтобы правильно направлять и регулировать обсуждение. Поговорим о них в следующей главе.

# Глава 7. Как нужно рассказывать истории

Идея историй на самом деле очень проста. Может быть, даже слишком проста. Множество людей, занятых в разработке программного обеспечения, воспринимают обсуждения как что-то очень непривычное… и, может быть, даже слегка неприятное. В результате многие скатываются к простому обсуждению требований, к которому они привыкли.

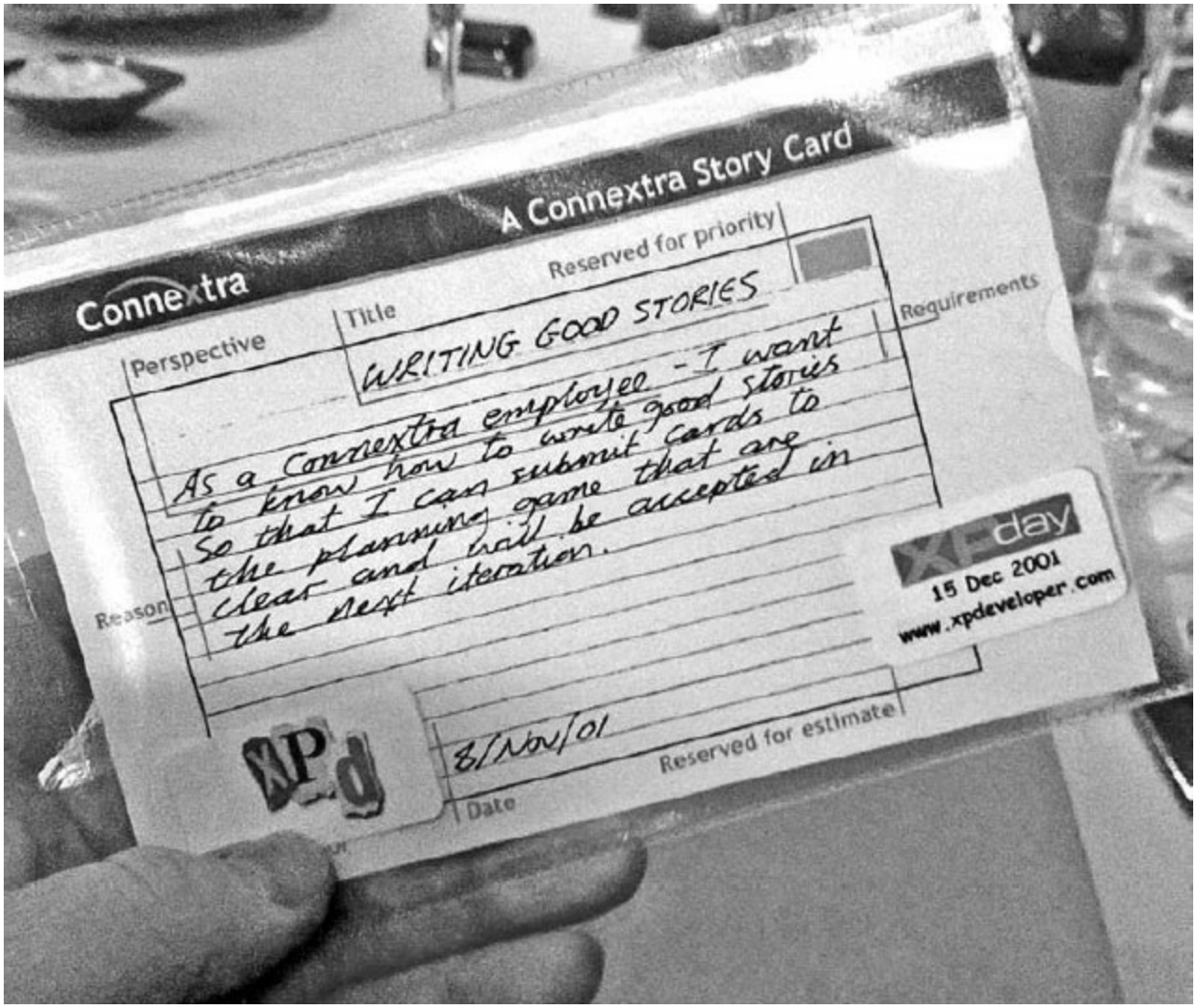
Когда Кент Бек впервые сформулировал идею историй, он не называл их *пользовательскими историями*, а просто употреблял слово *«истории»* – именно их, как он надеялся, вы будете составлять. Но вскоре после того, как были изданы его первые книги по экстремальному программированию, термин *«истории»* получил полезное уточнение *«пользовательские»* – таким образом, мы не забываем, что обсуждение должно вестись с точки зрения людей, находящихся вне программного продукта. Но, как оказалось, недостаточно лишь изменить имя.

# Классный шаблон Connextra

Это моя подруга Рэчел Дэвис. У нее в руках карточка историй.

В конце 1990-х Рэчел работала в компании под названием Connextra, которая была одним из первых адептов экстремального программирования, процесса Agile, откуда берут свое начало истории. Когда работники Connextra начали использовать истории, почти сразу обнаружились несколько распространенных проблем. Большинство людей, которые составляли в Connextra истории, работали в отделах продаж и маркетинга, поэтому они были склонны просто записывать функциональности, нужные пользователям. Но когда приходило время дискуссий между разработчиками, им приходилось искать того, кто предложил функциональность, и начинать настоящее обсуждение, включающее все «Для кого?» и «Почему?». Простая запись названия функциональности никак не помогала им понять, с какими людьми следует поговорить, или начать правильное обсуждение. Кроме того, в то время не было достаточного количества информации о том, что может или должно быть на карточках. Шаблон развивался в Connextra постепенно, с течением времени. Он был детищем не одной Рэчел, а скорее являлся следствием желания всей организации создавать по-настоящему значительные вещи.


Ребята из Connextra использовали этот шаблон некоторое время, а потом захотели поделиться своим замечательным изобретением с другими. Они напечатали стопку карт-образцов, которые продемонстрировали на XPDay 2001, небольшой конференции в Лондоне. На фотографии Рэчел держит именно такую карточку. Это последняя карточка из того набора и, возможно, последняя из существующих, поэтому ее можно назвать историческим памятником.




Вот как выглядит этот шаблон:

Как *[тип пользователя*], я хочу *[сделать то-то и то-то*], таким образом я смогу *[получить такую-то выгоду*].

Ребята из Connextra использовали этот простой шаблон для составления *описаний* своих историй. Они обнаружили, что написание этого небольшого дополнения перед началом изложения истории помогает им остановиться на минуту и сформулировать для себя: для кого, что, почему. И если выяснится, что данных недостаточно, составление истории оказывается под вопросом.

Когда приходит время начать обсуждение истории, ребята берут эти карточки и читают описания, которые дают отправную точку для дискуссии.

*Используйте простой шаблон историй, чтобы открыть обсуждение.*

Если у вас есть отдельная карточка, не принадлежащая карте историй, то шаблон – отличный способ завязать обсуждение. Помните, в главе 1 я упомянул одну из карточек в карте Гэри, где было написано: «Загрузить изображение»? Эта карточка была одной из деталей шага «Отредактировать рассылку». Я всегда пишу на карточках, составляющих карту, такие короткие глагольные фразы – они означают действия, которые люди выполняют, используя мой продукт. Но когда я рассматриваю какую-то карточку в отдельности, мне нужно увидеть историю внутри большого путешествия пользователя по продукту, которое представляет собой карта. Поэтому я могу сказать: «Как администратор музыкальной

группы, я хочу загрузить изображение и отредактировать свою рассылку таким образом».

Это очень эффективный прием. Выше карты могут находиться листочки с действиями разных типов пользователей, а цель пользователя записывается на карточке, находящейся выше той, на которую вы смотрите в данный момент.

Но не забывайте, это только запуск обсуждения. Продолжиться оно должно примерно так.

• «Почему администратору группы может понадобиться отредактировать рассылку?»

• «Ну, наверное, потому, что на рассылке должно быть фото группы, а оно там не появляется автоматически. Кроме того, администратор, вероятно, хочет, чтобы его рассылка выделялась и привлекала внимание – кому интересны стандартные рекламки?»

• «Это звучит разумно. А где, как вы думаете, люди наподобие администраторов хранят такие фотографии?»

• «Да где угодно. Они могут быть на жестком диске компьютера, в аккаунте Flickr и вообще в любом месте в Интернете».

• «Да? Признаться, я об этом не думал, я предполагал, что они хранят их только на жестких дисках».

• «Нет, много людей, с которыми мы говорили, хранят фотографии в самых разных местах. Наверное, стоит это учесть».

Именно так и выглядели большинство моих дискуссий с Гэри. По мере обсуждения мы делали записи на доске или на самих карточках, а вскоре начали даже рисовать эскизы наших идей.

Как видите, этот короткий шаблон совершенно не годится для того, чтобы служить спецификацией. Но когда мы используем его, чтобы начать обсуждение, оно идет куда живее и активнее, чем если бы мы просто начали говорить о загрузке файлов.

### *Как вновь открыть для себя ценность коротких обсуждений*

*Мэт Кроннер, ThoughtWorks*

Я работал в компании ThoughtWorks бизнес-аналитиком на проекте для правительственного агентства Великобритании. Мы несли ответственность не только за предъявление продукта, но и за обучение команды клиента некоторым практическим приемам методологии Agile. Это было не так-то просто, ведь у нас была большая команда – около 25 технологических или бизнес-специалистов. Представьте себе 25 человек в одной комнате, у каждого свои пожелания о режиме кондиционера, и вы поймете, в какой ситуации мы оказались.

Мы с владельцем продукта договорились, что напишем истории, а потом в начале двухнедельной итерации вся команда соберется и распланирует разработку. В силу необходимости на этом совещании присутствовало очень много людей, что и привело к полному провалу.

«Почему мы должны делать вот так?»

«Эти истории слишком длинные/короткие».

«В этом нет никакого смысла!»

«Я настаиваю, чтобы техническая реализация данного аспекта выполнялась вот так и никак иначе».

Это был долгий и утомительный спор. Честно признаюсь, уходя с совещания, я чувствовал себя ужасно и воспринял все происходившее как личное поражение.

Однако следовало что-то предпринять, чтобы все исправить, и мы решили от одного общего совещания, где обсуждалось бы все и сразу, перейти к небольшим, более фокусированным обсуждениям. Например, на первой неделе итерации мы привели в порядок бэклог. Это сделала небольшая группа людей (владелец продукта, менеджер проекта, бизнес-аналитик и технический архитектор), которые выступали в роли пользователей различных историй. В результате, когда мы позднее формировали итоговый бэклог с участием всей команды, недоумения и недовольства было куда меньше. На этих встречах мы скорректировали и усовершенствовали наши истории, а, например, расстановку приоритетов и оценку сложности реализации проигнорировали. Это сработало.

Мы также старались, чтобы процесс создания историй был более конструктивным. Если у меня возникали какие-то идеи по поводу истории, над которой я работал, я размещал их в верхней части доски с карточками в колонке «Анализ». Каждый день на совещании нашей команды я рассказывал о своей работе над конкретной историей и мог попросить кого-то из команды разработчиков уделить мне время. Мы садились и обсуждали, в каком направлении пойдет работа, как правило затрагивая технические аспекты, а затем записывали все это на бумаге. Мы не использовали Trello, где обычно хранилась наша доска с карточками в электронном виде, а вместо этого концентрировались на личных обсуждениях, иногда с помощью доски для записей. Работа в маленькой группе с глубоким погружением в детали очень полезна, а поскольку это обсуждение редко продолжалось больше 20 минут, оно никому не доставляло неудобств. Всем очень понравилось сотрудничать, содействуя общему делу.

К счастью, огромные совещания по наполнению бэклога ушли в прошлое. Кроме того, мы обнаружили, что размер историй постепенно становится постоянным. В результате этого планирование итераций проходило куда легче, а обсуждения стали более активными. Соответственно, возросло качество документирования историй и качество всей нашей работы.

# Шаблонные зомби и снегоочиститель

Термин *«шаблонные зомби»* впервые появился в книге Тома Де-Марко и его соавторов «Адреналиновые наркоманы и шаблонные зомби: понимание паттернов поведения проектов»[16]. Название говорит само за себя, но я приведу и авторское определение.

> *«Шаблонный зомби: проектная команда допускает, чтобы работа управлялась шаблонами, а не направляет процесс осознанно, чтобы обеспечить выпуск продукта наилучшим образом».*

Мы, в принципе, склонны злоупотреблять простотой использования таких вещей, как шаблон. Я неоднократно наблюдал, как люди мучаются, изо всех сил пытаясь впихнуть в шаблон идеи, которые в него не помещаются. Истории работы серверной части или функций, обеспечивающих безопасность данных, не так-то легко составить. Я также видел, как люди пишут и думают, ориентируясь на собственное представление о чем-то, а не на точку зрения людей, которые должны получить выгоду: «Как владелец продукта, я хочу, чтобы вы сделали кнопку для загрузки файла, таким образом мы удовлетворим требования заказчика». Просто ужас.

И это еще не самое худшее! По мере роста популярности универсальных шаблонов у некоторых людей складывается впечатление, что история не история, если она не записана по форме. Многие даже отказываются от использования заголовков и пишут на каждой карточке только длинные предложения по шаблону. Представьте себе, как бы вы пытались прочесть список историй, записанных таким образом. Представьте, как кто-то читает историю с помощью карты историй, где так выглядит каждый стикер. Ни один мозг этого не выдержит.

Все это меня бесконечно огорчает. Ведь истинная ценность историй не в том, что записано на карточке, – она заключается в том, что мы изучили и осознали, излагая историю.

> *История может быть историей, даже если она составлена не по шаблону.*

Человек на этой фотографии учиться кататься на горных лыжах[17]. Если вы когда-нибудь тоже этому учились и вам помогал инструктор, то вы хорошо знаете, что делает этот человек. Он выполняет прием торможения «плугом» – сводит носки лыж вместе и скользит на внутренних ребрах. Это самый простой способ контролировать скорость и оставаться в вертикальном положении, несмотря на то что к вашим ногам прикреплены две скользкие доски. Это самый лучший прием для начинающих лыжников. Но это не просто самый лучший прием – это самый лучший *обучающий* прием. В программе соревнований горнолыжников на Олимпийских играх нет упражнения «плуг». Ни один катающийся на склоне горы не будет восхищен вашим великолепным исполнением «плуга». Но, разумеется, стесняться здесь нечего. Видя, что вы катаетесь таким способом, все понимают, что вы новичок и пока учитесь.

Мне кажется, шаблоны историй работают точно так же, как «плуг». Используйте их, чтобы составлять описания к своим первым историям. Читайте их вслух, чтобы инициировать обсуждение. Но не слишком расстраивайтесь, если однажды обнаружится, что они работают не всегда. Как и техника «плуга» для горнолыжников, это не лучший выбор для сложного пути.

Мой любимый шаблон: если я записываю истории на стикерах или карточках и они не отправляются сразу же на большую сложную карту, то я даю им простые короткие названия, а затем записываю под ними:

Кто:

Что:

Почему:

Под каждый ответ отводится несколько строчек, ведь нужно перечислить каждого конкретного «Кто?», немного рассказать о «Что?», а также привести несколько различных причин для «Почему?». Кроме того, я люблю оставить на карточке место, чтобы добавить еще немного информации после того, как мы начнем обсуждать эту историю. Меня ужасно раздражает, когда некоторые пишут название истории прямо посередине карточки, ведь теперь на ней нельзя сделать никаких пометок! Но, конечно, тут я уже придираюсь.

## Чек-лист: о чем говорить

*По-настоящему обсудите «Кто?».* Пожалуйста, не говорите о пользователях вообще. Будьте конкретны. Ясно называйте тех пользователей, которых подразумеваете. Например, с Гэри мы говорили об администраторе группы или ее фанатах.

Говорите о разных типах пользователей. У многих видов программного обеспечения, особенно коммерческого, существует множество типов пользователей, работающих с одной и той же функциональностью. Обсудите функциональность с точки зрения разных пользователей.

Говорите о заказчиках. В случае разработки коммерческих продуктов заказчик (или лицо, принимающее решение о покупке) зачастую может быть одновременно и пользователем. Но, рассматривая корпоративные продукты, необходимо обсуждать также тех, кто принимает решение, организацию в целом и выгоды, которые они получают.

Говорите обо всех заинтересованных сторонах. Обсуждайте людей, спонсирующих покупку программы. Говорите о других людях, которые могут сотрудничать с вашими пользователями.

Случаи, когда можно учитывать только одного пользователя, очень редки.

*Как следует обсудите «Что?».* Я очень люблю, когда истории начинаются с пользовательских задач – действий, которые пользователи хотят выполнить, применив мой программный продукт. Но что же делать с сервисами, работа которых не видна в пользовательском интерфейсе, – теми, которые авторизуют вашу кредитную карту для покупки или пускают вас на сайт страховки? Вашим пользователям не приходится явно делать выбор, чтобы авторизовать свои кредитные карты или подтвердить введенные данные для авторизации. Поэтому не только можно, но и нужно обсудить самые разные сервисы и системы, которые могут их вызывать. Стоит также рассмотреть особые компоненты UI и поведение экрана. Просто не упускайте из виду заинтересованных людей и их мотивы.

*По-настоящему обсудите «Почему?».* Обговорите самые разные заботы и нужды пользователей. Как следует погрузитесь в «Почему?», потому что, как правило, их несколько и они накладываются друг на друга. Можно очень долго «тыкать их палочкой», чтобы наконец выявить истинные мотивы, лежащие в основе всего.

Обсудите мотивы других пользователей. Обсудите мотивы компании, в которой работают пользователи. Обсудите мотивы всех сторон, заинтересованных в бизнесе. Исследуя «Почему?», можно сделать множество интереснейших открытий.

*Обсудите то, что происходит за рамками программного обеспечения.* Подумайте, где и когда люди могут использовать ваш продукт, как часто они это делают. Представьте себе, кто еще может присутствовать при использовании продукта. Все эти факторы могут вам помочь нащупать наилучшее решение.

*Обсудите, что произойдет, если все пойдет не так.* Что будет, если произойдет какой-то сбой? Что случится, если система вдруг прекратит работу? Какими альтернативными способами пользователи могут достичь своих целей? Как они удовлетворяют свои нужды сейчас?

*Обсудите вопросы и предположения.* Если вы проговорите все это, то, скорее всего, столкнетесь со многими вещами, о которых не знаете. Сформулируйте свои вопросы и обсудите, насколько важно получить ответы перед тем, как приступить к работе над

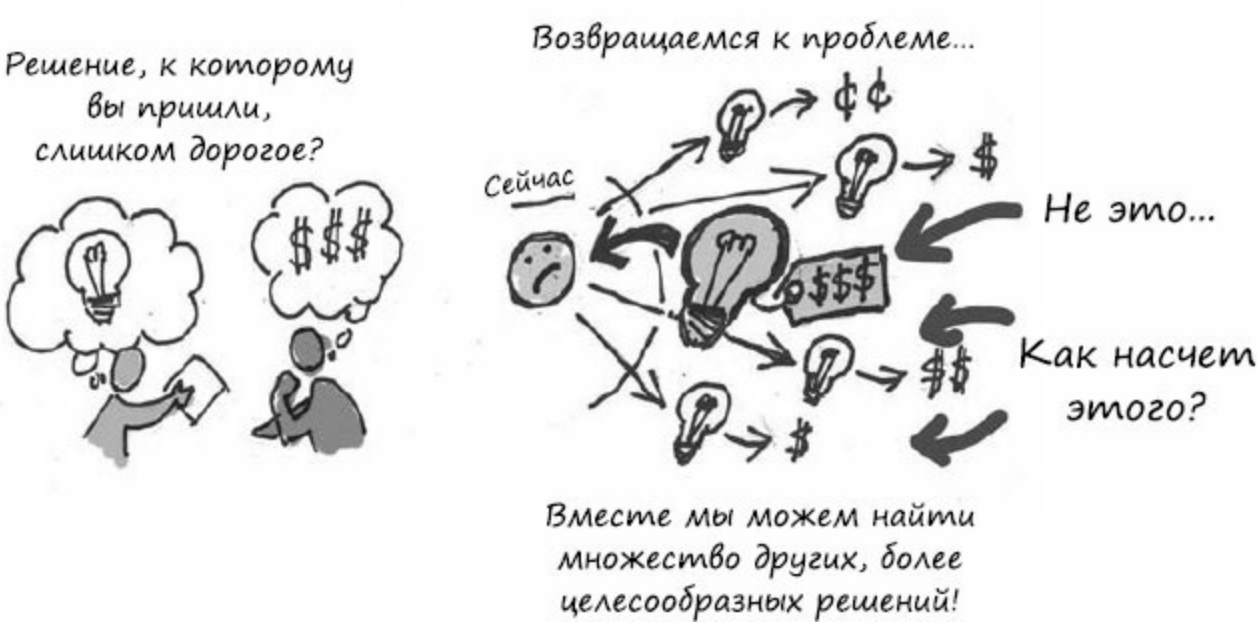
программным обеспечением. Нужно решить, кто проделает все необходимое для получения информации, и сохранить вопросы до следующего обсуждения. Как вы убедитесь, порой для полного рассмотрения некоторых историй требуется очень долго их обсуждать.

Запланируйте время на проверку своих предположений. Действительно ли вы понимаете пользователей? Вы точно знаете, чего они хотят? Реальны ли их проблемы? Смогут ли они, используя ваше решение, справиться с этими проблемами?

Рассмотрите также свои технические предположения. На какие базовые системы мы полагаемся? Будут ли они в самом деле работать так, как мы планируем? Есть ли какие-то технические риски, которые необходимо учесть?

Все эти вопросы и предположения могут потребовать дополнительного исследования или изучения. Составьте для этого особый план.

*Обсудите вероятность нахождения лучшего решения.* По-настоящему крутые результаты получаются тогда, когда в ходе обсуждения истории участники отклоняют первоначальные предположения о том, каким должно быть решение, возвращаются к рассматриваемой проблеме, а затем все вместе приходят к другому решению, более эффективному и экономичному в реализации.




*Обсудите «Как?».* Очень часто, присутствуя на обсуждении историй, я слышу чей-то раздраженный голос: «Мы должны обсуждать, что мы делаем, а не как!». Под этим обычно подразумевается, что нужно обсуждать действия и мотивы пользователей, а не код, который необходимо написать. А я чувствую точно такое же раздражение, когда мы говорим только о «Что?», не затрагивая «Почему?». В действительности при хорошем обсуждении мы стремимся к оптимизации всех трех составляющих. На самом деле нехорошо, когда кто-то предполагает, что определенное решение или какой-то способ технической реализации является требованием. Поэтому без явного обсуждения «Как?» (а если вы разработчик, вы неизбежно думаете об этом аспекте) очень трудно оценить стоимость решения. А слишком дорогое решение никак не может быть наилучшим.

Уважительно относитесь к опыту и компетентности других участников обсуждения. Не стоит указывать высококвалифицированному техническому специалисту, как ему делать свою работу. Не говорите кому-то хорошо знакомому с пользователями и их работой, что он

ничего не понимает. Задавайте вопросы и искренне пытайтесь научиться чему-то друг у друга.

*Обсудите, сколько времени понадобится.* В конце концов все-таки приходится принять несколько решений, чтобы продвинуться вперед и что-то построить или не построить. Эту ситуацию можно сравнить с решением о покупке товара, не имеющего ярлычка с ценой.

В мире программного обеспечения понятие стоимости в первую очередь связано с временем, которое потребуется на написание кода. На ранних этапах вполне допустимо использовать понятия «очень долго» или «пару дней». Еще лучше сравнить то, что вы обсуждаете, с чем-то уже законченным: «Это похоже на те улучшения для комментариев, которые мы реализовали в прошлом месяце». По мере того как мы все ближе подбираемся к началу разработки, все дольше обсуждаем и принимаем все больше решений, оценки трудозатрат должны становиться все более точными. Но не забывайте: в любом случае мы говорим лишь об оценках, а не об обязательствах.

# Сделайте «фото из отпуска»

Поскольку нужно очень многое обсудить и хорошо бы ничего не забыть, убедитесь: вы записали самое важное, что позволит позднее припомнить принятые решения, а также вопросы и предположения, в которых следует хорошенько разобраться. Не забудьте озвучивать ход своих мыслей, чтобы и все остальные понимали, что записывается.

Если вы пишете на бумаге, прихватите ее с собой и заглядывайте в нее позднее. Если записи сделаны на стене, можно просто обращаться к ним по мере надобности. Кроме того, если вы общаетесь как настоящая команда, то обнаружите, что вам не нужно беспрестанно все повторять. Люди и так будут помнить подробности обсуждения, особенно если вы сопровождали его документами и простыми рисунками… теми самыми «фото из отпуска».


Эта небольшая группа сейчас обсуждает историю. По мере обсуждения они визуализируют свои идеи и делают заметки о решениях.

Мой любимый подход именно такой. Я делаю записи на большом листе бумаги или доске прямо во время разговора. Я люблю сделать прямо на доске пометку о том, кто присутствовал на обсуждении, а затем фотографирую свои записи. Затем любым способом отправляю фото всем присутствующим. Я хорошо знаю, что в любую минуту могу припомнить детали или записать их более формально, если вдруг понадобится. Если у меня не получится точно припомнить, что было сказано, наверняка это вспомнит другой участник обсуждения: я всегда знал, что записывать имена участников – хорошая идея.

# Придется о многом позаботиться

Да, обсуждение истории включает в себя множество составляющих, и это может немного пугать. В этот момент вы, возможно, с тоской вспоминаете старые добрые времена, когда все, что вам нужно было, – это правильно понять требования, и хотите вернуться туда, где вам не приходилось по-настоящему решать задачи. Вы просто должны были создать, что сказано, а о том, правильны ли инструкции, заботился кто-то другой. Но я уверен, что на самом деле и вам, и почти всем прочим людям нравится решать проблемы. Так что это ваш шанс.

Возможно, обсудив все, о чем мы здесь говорили, вы получите огромный объем информации, которую необходимо хранить и отслеживать, а на стикеры или карточки она никак не помещается. Все в порядке. Так и должно быть. Давайте поговорим о том, что необходимо писать на карточке, а чем можно пренебречь.

# Глава 8. Не бросайте на карту всё

Да, короткие заголовки историй на карточках помогут вам запланировать и провести множество обсуждений между людьми, которые умеют писать программы, и людьми, которые разбираются в проблемах, для решения которых предназначаются программы. Но, к сожалению, выпуск полностью готового программного продукта требует участия гораздо большего количества людей.

В обычной команде вы, скорее всего, увидите менеджеров проекта, менеджеров продукта, бизнес-аналитиков, тестировщиков, специалистов по пользовательскому взаимодействию, технических писателей и многих других, о которых я наверняка забыл. Все они могут рассматривать одни и те же карточки, но обсуждения будут различаться, так как эти люди отвечают за решение различных проблем.

# Разные люди, разные обсуждения


Если я владелец или менеджер продукта и отвечаю за его успех, мне нужно немного больше, чем другим, знать о целевой аудитории. Мне нужно сформулировать гипотезы о том, как много людей купят или будут использовать этот продукт или как это повлияет на прибыль моей компании. И говорить я хочу об этом.

Если я бизнес-аналитик, я должен погрузиться в море деталей, понять, как должен выглядеть пользовательский интерфейс, а также вникнуть в бизнес-правила компании, которые лежат в его основе.

Если я тестировщик, мне нужно думать о том, в каком случае продукт может дать сбой. Я хотел бы обсудить, что мне нужно учесть при составлении плана тестов.

Если я дизайнер UI, я хочу слушать о том, как должен выглядеть графический интерфейс, не больше, чем разработчик – о том, как должен быть написан код. Мне надо знать, кто будет пользоваться этим интерфейсом, что и почему делают эти люди, в результате чего я смогу создать эффективный и удобный пользовательский интерфейс.

И наконец, если я менеджер проекта, ответственный за координацию действий специалистов, мне нужно уделить внимание всем, поучаствовать во всех обсуждениях, чтобы быть в курсе деталей. Мне нужно учесть зависимости, график и статус разработки, когда она начнется.

Как много обсуждений! Некоторые из них должны пройти раньше остальных. Многие будут повторяться больше одного раза. Поэтому, чтобы быть точными, мы должны добавить еще несколько дюжин «П» к нашим трем «П». Но, к счастью, если вы постараетесь и проведете действительно предметные обсуждения, чтобы по ходу дела родилось одинаковое понимание, то избежите множества недоразумений и корректировок.

*Каждую историю придется обсуждать с разными людьми и с разных точек зрения.*

# Нам понадобится карточка побольше

Я надеюсь, что, читая заголовок, вы вспомнили старый фильм «Челюсти», где, впервые увидев приближающуюся огромную акулу-людоеда, шериф Броуди говорит Квинту: «Кажется, вам понадобится лодка побольше».

Как видите, изначальная идея все та же: я могу взять карточку и написать на одной стороне название. После обсуждения я переворачиваю ее и записываю все детали, наши решения и соглашения, к которым мы пришли. Я могу набросать там же пользовательский интерфейс и разместить много другой информации. На некоторых проектах это и в самом деле может работать. Здорово, если и у вас получится, но, как правило, так бывает в маленьких командах, члены которых тесно связаны друг с другом, а информация известна всем. В таких командах не приходится много записывать, чтобы помнить.

Но я не думаю, что даже Кент и его ребята, которые впервые сформулировали концепцию историй, действительно предполагали, что материалы всех обсуждений, куда вовлечено столько людей, могут храниться на единственной карточке. Такого, как правило, не бывает.

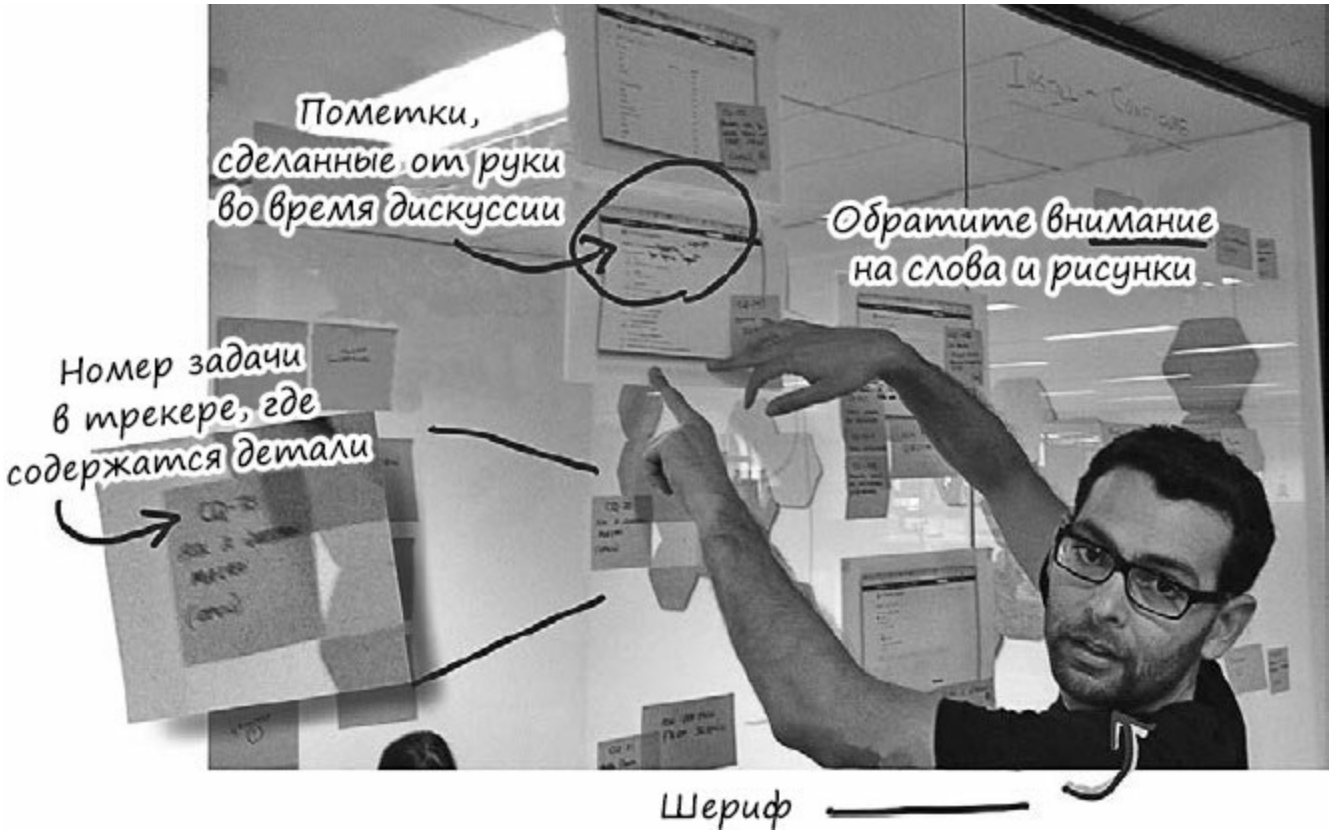
Метафора, которую я хорошо воспринимаю: карточка в библиотечной картотеке (впрочем, если вы не настолько стары, чтобы застать времена, когда в библиотеках были *настоящие* картотеки, вам не удастся ее оценить). Истории, записанные на карточках, работают примерно так же.


Если я возьму из картотеки карточку, на ней будет ровно столько информации, сколько нужно для поиска книги. Скорее всего, там указаны название, автор, аннотация, количество страниц, жанр – скажем, «научно-популярная литература», – а также код (помните десятичную классификацию Дьюи?[18]), с помощью которого можно найти экземпляр этой книги в хранилище. Карточка – просто символ, который легко хранить и просто найти. Никто не перепутает карточку с книгой. Библиотечная картотека очень удобна, так как занимает намного меньше места, чем тысячи книг. А карточки можно сортировать множеством способов, например по авторам или по темам.

Ваши истории должны работать примерно так же: вы можете записать их на карточках, разместить на большом листе либо занести в свой любимый инструмент для планирования или систему, которую используют все в вашей компании (периодически ругая на все корки). Находясь в библиотеке, вы знаете, что там есть нужная книга, и если нашли в картотеке нужную карточку, то и книгу с легкостью найдете. То же с историями: вы знаете, что у вас есть растущий объем информации. Она развивается и увеличивается с каждым обсуждением. И если в вашей компании хоть сколько-нибудь ценится прозрачность сведений, найти ее тоже будет легко.

Если же вы истинный консерватор, то можете хранить подробности всех обсуждений на больших листах бумаги, развешанных на стене, так что обсуждение можно продолжить в любой момент. Но помните: однажды придется снять их, если вы закончите работу или место на стене закончится. Поэтому не забудьте сфотографировать листы и сохранить снимки на будущее.

### *Как создатели инструментов проводят удачные обсуждения историй*




Это мой друг Шериф, который работает в компании под названием Atlassian. Шериф занимается Confluence – популярной вики-системой, используемой огромным количеством организаций для хранения информации, с которой они работают, и управления ею наряду с прочим. Кроме того, они создали JIRA – один из самых популярных инструментов для управления разработкой по методологии Agile. Разумно предположить, что компания, которая специализируется на создании инструментов для электронного хранения информации и управления ею, и сама пользуется своей продукцией – как говорится, «сами ешьте еду своей собаки»[19], – и в Atlassian это действительно так. Но кроме того, в этой компании понимают, как важны плодотворные обсуждения историй лицом к лицу.

Осматривая офис Atlassian в Сиднее, я видел стены, увешанные стикерами, разрисованными досками и эскизами экранов. Присмотревшись, на стикерах можно было заметить номера задач в системах для управления работой, которыми пользовалась команда. Ребята ловко перемещались от систем к физическому пространству и обратно. Когда Шериф показал мне, как и что они хранят в Confluence, я был поражен сочетанием фотографий, коротких видео, а также множества обсуждений.

# Борьба и единство радиатора и морозилки

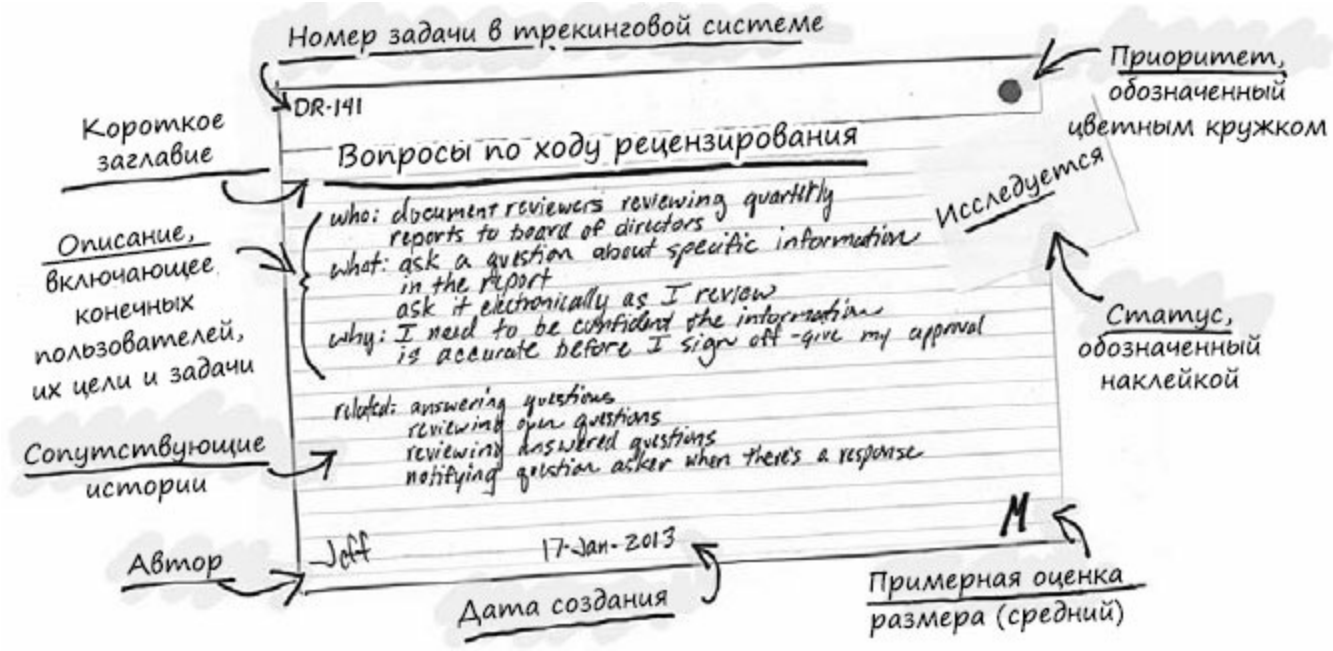
В своей книге «Создание программного обеспечения как коллективная игра»[20] Алистер Коберн употребляет термин «информационный радиатор», чтобы описать, как информация, размещенная в комнате на видном месте, помогает работающим там действовать эффективно. Люди постоянно подходят, чтобы посмотреть на нее. Если эта информация актуальна и полезна, около стены разворачивается множество обсуждений, во время которых люди могут обращаться к записям, дополнять и развивать их.

Когда я захожу в комнату, где стены совершенно пусты или украшены пейзажами и натюрмортами, а то и кошмарными мотивационными плакатами, я здорово огорчаюсь. С помощью этих стен сотрудники могли бы развернуть эффективнейшую совместную работу! Но если то, что висит на стене, является информационным радиатором, то, очевидно, должны существовать средства, являющиеся *информационным морозильником* – местом, где информация должна храниться и, возможно, надолго оставаться погребенной под коркой льда, как продукты в морозилке у вас дома (я, например, регулярно нахожу там что-то неожиданное).

В Atlassian мне прежде всего бросилось в глаза то, что вся имеющаяся у них информация – как внутри, так и вне их инструментов – актуальна и полезна.

## *Что на самом деле должно содержаться на карточке истории*

Представьте себе карточку из библиотечной картотеки. На ней имеется ровно столько информации, сколько нужно вам для управления карточками и подтверждения того, что вы обсуждаете нужную книгу. Хорошая карточка выглядит примерно так.



В общем случае на карточке, скорее всего, будет следующее.

• *Короткое заглавие.* Им легко оперировать во время обсуждения, когда вы говорите об этой истории. Хорошее название – самая полезная часть истории. Не стесняйтесь изменить его, если первоначальный вариант получился не совсем понятным.

• *Описание.* Одно-два предложения, описывающие то, как вы представляете себе ход истории. Хорошая идея описать здесь «Кто?», «Что?» и «Почему?»: кто использует эту функцию или нуждается в ней, что он будет с ней делать и какую пользу от этого получит.

По мере обсуждения историй вы будете добавлять информацию, которая подводит итог некоторых дискуссий. Здесь будут следующие элементы.

• *Номер истории.* Когда историй у вас будет много и вы захотите перенести их в трекинговую систему, номер поможет вам найти нужный, так же как десятичный код Дьюи помогает найти книгу в библиотеке. Но, что бы вы ни делали со своими историями, *пожалуйста*, не называйте их по номерам! Если вас так и тянет называть номер вместо названия – это верный симптом того, что название неудачное. В конце концов, библиотекари же не называют книги по номеру в десятичной системе Дьюи!

• *Оценка трудозатрат, размер или бюджет.* Когда вы начнете обсуждать историю, вам непременно понадобится прикинуть, сколько времени займет разработка этого программного обеспечения. Эту величину можно назвать по-разному, например *оценка времени* или *трудозатрат, размер* или *бюджет*. Пользуйтесь термином, принятым в вашей организации.

• *Важность.* Возможно, вы будете долго спорить о важности одной истории по сравнению с другой. Одним нравится использовать числовую шкалу, а другим – помечать карточки наклейками с текстом «*Высокая*», «*Средняя*», «*Низкая*».

• *Параметры*. Если вам и в самом деле важны результаты, укажите особые параметры, которые будете отслеживать после выпуска программного продукта, чтобы убедиться в его успешности.

• *Зависимости.* Другие истории, от которых данная история может зависеть или быть с ними тесно связанной.

• *Статус*. Запланирована ли история для определенного релиза? Началась ли работа над ней? Идет ли разработка? История завершена?

• *Даты*. Как на библиотечной карточке имеется дата издания книги, так и на вашей карточке нужно фиксировать даты, когда история была добавлена, принята в разработку и завершена.

Вы можете делать на карточке любые нужные вам пометки. Можно также перевернуть ее и на другой стороне что-то записать или перечислить критерии приемки.

Единственная строго обязательная надпись на карточке – это хорошее название. Вся остальная информация может быть полезной, но только вам и вашей команде решать, нужна ли она.

На карточке умещается не так уж много сведений, и это в общем хорошо. Помните: это только символ, который вы применяете для планирования. Вы можете использовать карточки или стикеры. Наличие бумажных карточек упрощает

обсуждение: рассказывая, вы можете просто говорить: «вот это» или «то» – и указывать на карточку на столе или на доске. Бумажные карточки можно перемещать по столу, сортировать по важности, развешивать на стенах. Можете размахивать ими во время разговора, чтобы подчеркнуть важность того, что говорите. Если вы попробуете проделать это с увесистым многостраничным документом, то, скорее всего, случайно ударите кого-то, а то и самого себя. И конечно, кучу карточек можно организовать в карту историй, чтобы составлять более длинные и сложные истории.

## Эта штука здесь не поможет!

Дровосек встречает в лесу человека, который пытается срубить дерево и для этого изо всех сил бьет по нему молотком.

«Эй! – говорит дровосек. – Кажется, у тебя неподходящий инструмент, дружище. Попробуй-ка это!» – и протягивает человеку пилу.

Человек сердечно благодарит дровосека, и тот продолжает свой путь, радуясь, что удалось сделать доброе дело. А человек тем временем берет пилу и начинает изо всех сил, как молотком, бить ею по дереву…

Этот анекдот учит нас, что мы можем, с одной стороны, использовать для работы *неверный инструмент*, а с другой – использовать *инструмент неверно*.

Когда я рассказываю, как в компаниях наподобие Atlassian применяют рабочие инструменты, обычно все удивляются. Так происходит потому, что, как правило, все пытаются заменить доски и стикеры электронными инструментами. И разумеется, ни к чему хорошему это не приводит. Это может происходить и потому, что не тот инструмент используется не для той работы, и потому, что правильный инструмент используется неверно. Чтобы понять, где ошибка, нужно в первую очередь сконцентрироваться на работе, а не на инструменте.

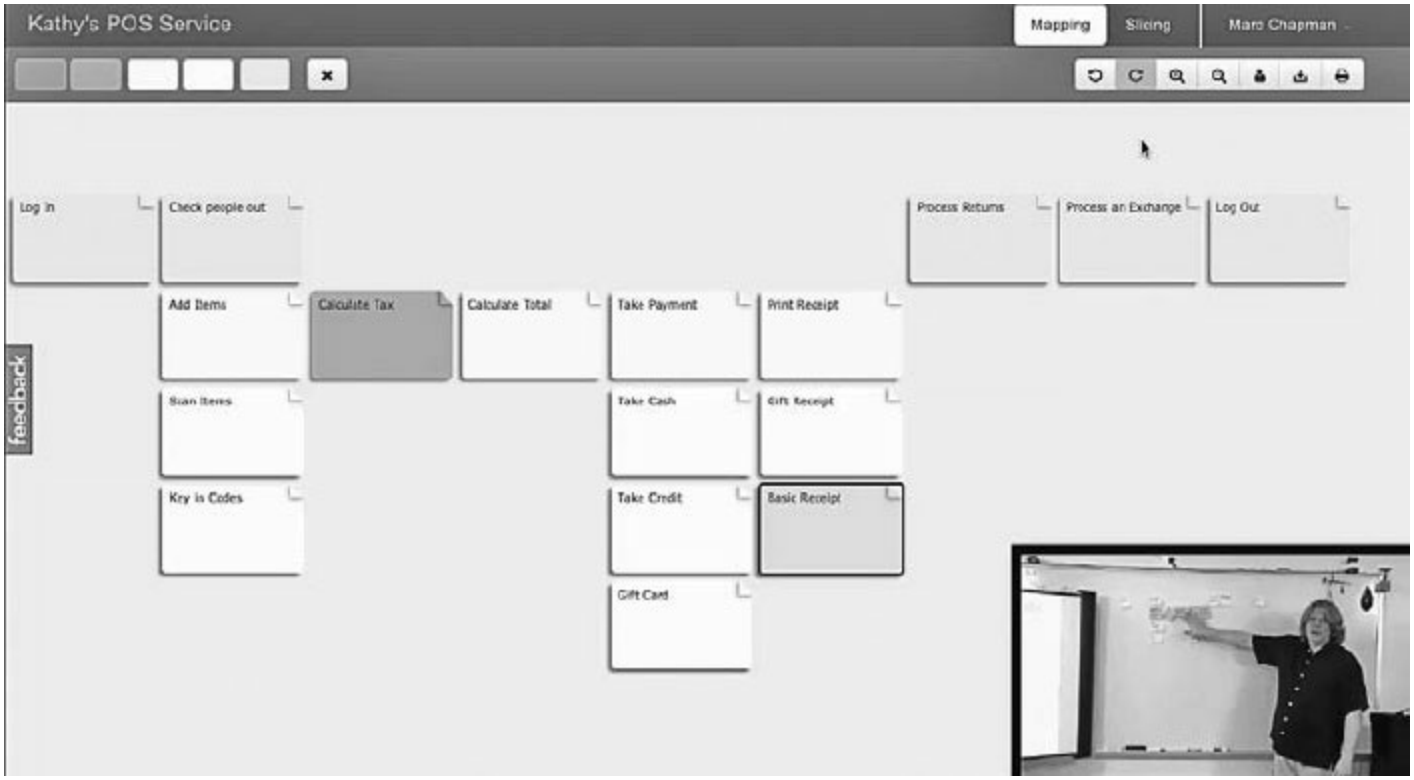
## Выработка единого понимания

Когда мы вместе работаем, чтобы составлять истории и принимать решения о том, что необходимо построить, наша первейшая цель – выработать одинаковое понимание. На данном этапе вынесение наших мыслей на всеобщее обозрение и их правильная организация критически важны! Ничто не заменит совместную работу около доски с пачкой стикеров в руках. А если вы пытаетесь выполнить эту задачу, работая удаленно, скорее всего, вам будет нелегко. Видеоконференции, где вы можете видеть лица коллег, не особенно помогут, ведь нужны-то вам не лица, а идеи, которые в виде записи можно поместить для обсуждения на стол или стену.

> *Во время веб-конференции используйте настольную или веб-камеру, которые будут транслировать всем стену или стол, где происходит работа.*

Мне приходилось работать в командах, где на всем протяжении видеоконференции использовали камеры, которые фокусировались на модели, растущей на стене, а не на лицах участников.

Если вы используете для визуализации какой-то инструмент, то идеально, когда все участники могут добавлять и двигать элементы точно так же, как если бы они работали все вместе перед доской. Вот так выглядит приложение под названием Cardboard.




Сотрудник составляет карту в Cardboard одновременно с тем, как Дэвид Хассман, один из создателей Cardboard, формирует карту на стене. Остальные, кто принимает участие в составлении карты удаленно, видят и то и другое в реальном времени. Они могут добавлять, удалять и менять карточки, каждый видит все, что делает другой. Вы можете виртуально отступить назад и окинуть взглядом картину в целом, что очень удобно. Компьютерный экран в данном случае всего лишь портал к тому, что вы могли бы увидеть, работая у стены.

> *Если некоторые ваши коллеги работают удаленно, используйте инструменты, позволяющие каждому видеть, дополнять и изменять модель в реальном времени.*

К счастью, я вижу, что на рынке сейчас появилось очень много инструментов, которые поддерживают совместную работу и облегчают выстраивание одинакового понимания. Это не может не радовать.

## Запоминание

После того как все мы вместе отлично поработали, чтобы прийти к общей точке зрения, нужно сохранить копии всех созданных моделей или примеров, чтобы использовать их в качестве «отпускных фотографий». В дальнейшем они помогут нам припомнить дискуссию в деталях. В продуктах наподобие Atlassian Confluence предлагается вики-движок с широкими возможностями, где вы можете использовать не только слова, но и картинки или видео. Самые быстрые способы создания документации – это фотографии или короткие видео по результатам совместной работы.

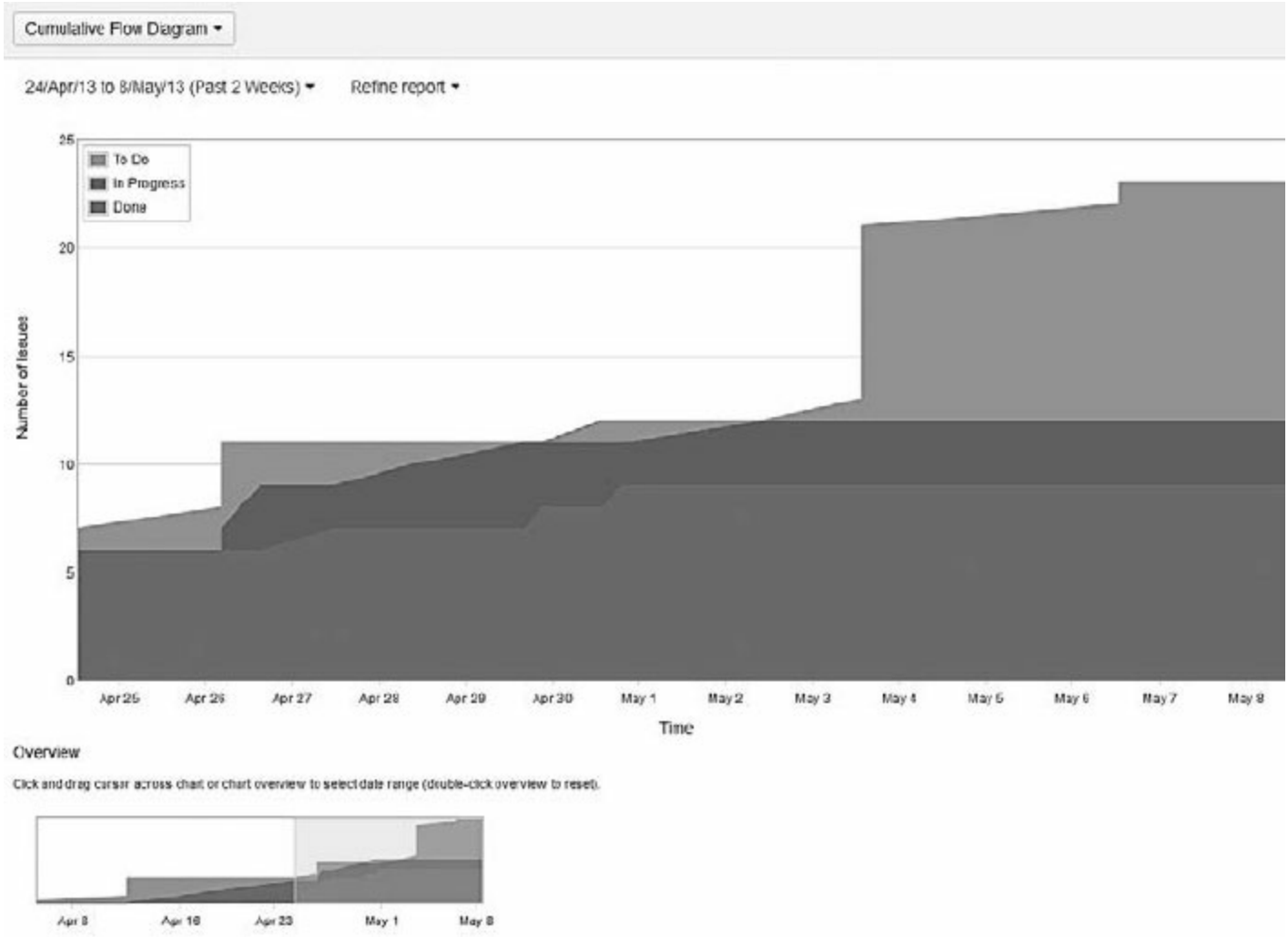

В Atlassian так и делают: фотографируют стену с результатами работы, а затем загружают фотографию на вики для хранения.

> *Используйте инструменты, позволяющие размещать фотографии, видео и текст, которые помогут вам сохранить и вспомнить ваши обсуждения.*

Лично мне нравится, не делая лишних телодвижений, хранить информацию прямо на стене, но я фотографирую стену на тот случай, если уборщица ночью ликвидирует мою работу. Если я должен передать информацию людям, которые по каким-то причинам не смогли присутствовать при составлении модели даже виртуально, то снимаю короткое видео, проходя шаг за шагом всю модель, и размещаю его там, где все могут на нее посмотреть.

## Трекинг

Одни из лучших функций инструментов, которыми мы пользуемся, – это хранение информации о нашей работе и возможность отслеживать ее прогресс. Нам слишком обременительно держать в памяти данные, с которыми легко управляются программы, например точные даты начала и окончания работы, степень ее выполнения. Самые лучшие инструменты могут открыть нам то, о чем мы и не подозревали, в то время как мы просто отслеживаем свою ежедневную работу.




Это диаграмма объема сделанной работы, сгенерированная с помощью JIRA, – продукта Atlassian. На рисунке хорошо видны работа, которую мы делаем, и изменение ее статуса с течением времени. Делать такой график вручную – очень скучная и трудоемкая работа.

Для небольших команд, работающих в одном офисе, и маленьких проектов стены будет вполне достаточно. Но если вы работаете в большой команде, члены которой физически удалены друг от друга, а проекты довольно продолжительны, вам непременно нужна трекинговая система для отслеживания всех деталей.

*Используйте инструменты для организации работы, отслеживания и анализа прогресса.*

Суть здесь в том, чтобы правильно использовать правильные инструменты. Не пытайтесь применить даже самые лучшие трекинговые системы, чтобы выработать одинаковое понимание, но и мучиться, разворачивая на доске сложную аналитику, не стоит.

С точки зрения Agile идеальным вариантом было бы решать рабочие задачи быстро и просто, как можно дольше работая с карточками и досками. И я обещаю вам: если вы сумеете сохранить простоту и скорость, избегая избыточного использования инструментов, в итоге вы останетесь довольны. Помните: все эти инструменты – лишь средства для достижения цели. Далее мы поговорим о том, что происходит после того, как готовы карточки.

# Глава 9. Карточка – только начало

Наши три «П» только начало.

Помнится, я утверждал, что не чувствую себя обязанным цитировать манифест Agile, но, похоже, придется – хотя бы небольшую часть. Одно из важнейших утверждений манифеста гласит: «Работающий продукт важнее исчерпывающей документации». Я бы перефразировал его так: «Работающий продукт важнее исчерпывающего обсуждения», но смысл останется неизменным. Все обсуждения, а также документы, которые помогают нам хранить их результаты, – только средства для достижения цели. В конце концов нам нужно что-то создать.

Если мы закончим цикл, модель будет выглядеть так, как на стр. 158.

Существует несколько подводных камней, которые почти всегда обнаруживаются после того, как мы, казалось бы, выработали идеальное одинаковое понимание и пришли к соглашению о том, что мы делаем. Остерегайтесь их.

# Разрабатывайте, держа в голове ясную картину

После обсуждений и записи деталей, которые помогут нам припомнить, что было сказано, а также после подтверждения готовности мы наконец можем приступать.

• Программисты могут начать писать код.

• Тестировщики – составить план тестирования и начать его выполнять.

• Графические дизайнеры – создать детальные проекты UI и электронные ресурсы, если они еще этого не сделали в процессе выработки одинакового понимания.

• Технические писатели – написать или обновить файлы справки, а также другие текстовые документы.


Самое важное здесь, чтобы все эти люди *держали в голове одну и ту же картину,*

*которую они создали, разговаривая друг с другом.*

Здесь я хочу сделать паузу. Подумайте об этом.

А следующие предложения я хотел бы проговорить медленно, поэтому, пожалуйста, прочтите их не спеша.

> *Передача всех подробностей истории кому-то еще для реализации не работает. Не делайте этого.*

Если в результате совместной работы с коллегами было выработано единое мнение о том, что нужно разработать, и вы задокументировали все важные детали, которые необходимы каждому для работы над проектом, у вас, скорее всего, появится искушение передать их кому-то еще. В конце концов, вот вы смотрите на информацию и вам все абсолютно ясно! Но не стоит обманывать себя. Вам все ясно только потому, что в вашей умной голове полно деталей, которых нет в документации. Мозг с такой легкостью ими оперирует, что вам даже не сразу удастся определить, чего не хватает. Помните: все эти детали есть на *ваших*, а не на *чужих* «отпускных фото».

# Заложите традицию устного рассказа

Поделиться информацией об истории чрезвычайно легко. Любой, кто хорошо понимает историю и ориентируется в информации о ней, с легкостью перескажет ее другому человеку, который хочет ее узнать. Сейчас дело пойдет куда проще, чем на первых обсуждениях, где вам приходилось принимать основополагающие решения (которые, как вы надеетесь, не придется менять позднее). Используйте имеющиеся у вас записи для изложения истории. Рассказывая, указывайте на иллюстрации. Поощряйте вашего слушателя задавать вопросы и вносите изменения в рисунки – это облегчает запоминание. Помогите этому человеку создать свое собственное «отпускное фото» на основе имеющейся информации об истории.


Очень часто возникает вот такая неприятная ситуация: некоторые люди думают, что, раз любой член команды теоретически может подключиться к работе над какой-либо историей, значит, в каждом обсуждении должна участвовать вся команда. Если такое практикуется и в вашей компании, вы наверняка слышали об этом, ведь все вокруг беспрестанно жалуются на бесчисленные совещания, которые нужно посещать. Кроме всего прочего, слово «совещание» зачастую является всего лишь эвфемизмом непродуктивной совместной работы.

Дискуссия и принятие решений наиболее эффективны в группах от двух до пяти человек, которые похожи на компанию, беседующую за обеденным столом. Вы, наверное, замечали: когда друзья выбираются перекусить и за столом лишь несколько человек, беседа течет легко. Но если их больше пяти, поддерживать общий разговор становится трудно.

Позвольте небольшим группам работать вместе и принимать решения, а затем продолжите общение, в разговорной форме сообщив результаты всем остальным.

# Следите за результатами своей работы

Если вы команда, то вы работаете все вместе, вооружившись общим пониманием того, что и как вы создаете. По ходу работы вы продолжаете время от времени что-то обсуждать, так как никто и никогда не может предусмотреть всего сразу. А когда программный продукт закончен, вы снова собираетесь вместе и обсуждаете результаты.

Это подходящий момент, чтобы поздравить друг друга с отличной работой. Очень здорово видеть реальный прогресс. В традиционном подходе к разработке программного обеспечения возможность увидеть результаты тяжелой работы предоставляется куда реже и, как правило, не всегда доходит до команды. В типичном процессе Agile, например Scrum, оценка и обзор продукта делаются каждые несколько недель, в конце спринта. В самых лучших командах сотрудники часто собираются вместе, чтобы оценить, как они справляются со своей работой. Но не стоит ограничиваться простой демонстрацией: после взаимных поздравлений найдите время для короткого, но вдумчивого анализа качества работы, которую вы проделали.

Говоря о качестве, я начинаю дискуссию с трех аспектов.

**Качество пользовательского взаимодействия.** Посмотрите на свою работу с точки зрения конечного пользователя. Очевидно ли, как пользоваться продуктом? Приятно ли им пользоваться? Каков внешний вид? Нет ли несоответствий с вашей торговой маркой или другими функциональностями?

**Функциональное качество.** Выполняет ли программа свои задачи согласно вашим договоренностям, без багов и ошибок? Тестировщики и другие члены команды, вероятно, потратили много времени на тестирование, и вы, наверное, уже исправили все ошибки. Но хороший тестировщик скажет вам, что в продукте почти всегда скрываются баги, которые могут проявиться позднее. Впрочем, возможно, он заверит, что продукт надежен как скала.

**Качество кода.** Хорошо ли написана программа? Соответствует ли код нашим стандартам? Мы будем еще какое-то время иметь дело с этим продуктом, поэтому очень важно, чтобы легко было работать с кодом, развивать и поддерживать его. Или же у нас появилась очередная порция технического долга, с которым придется разбираться позднее?

У меня для вас плохая новость: скорее всего, найдется довольно много вещей, которые можно было бы сделать лучше.

Чтобы все были спокойны, стоит четко разделять два соображения. Первое: *сделали ли мы то, что договорились сделать?* Второе: *если мы видим именно то, что договорились сделать, нужно ли внести какие-то изменения?*

Все вы с самого начала напряженно работали над определением проблемы пользователей, старались понять, каким образом программный продукт может решить эти проблемы и как сделать его разработку максимально экономичной. Вы сделали все возможное, чтобы выявить признаки, свидетельствующие об успешном завершении работы. Проверьте все это и поздравьте друг друга с тем, что вам удалось многого достичь. Вы получили именно то, к чему договорились прийти.

В этот момент у меня в голове начинает звучать старая песня Rolling Stones. Если вы ее знаете, подпевайте: «You can’t always get what you want. But, if you try sometime, you just might find, you get what you need…»[21] Какая ирония – в мире программного обеспечения все происходит ровно наоборот.

Вы вместе работали над договоренностями о том, к чему вы хотите прийти. И если ваша команда достаточно компетентна, то, скорее всего, у вас это неплохо получается. Но порой, лишь увидев результат работы, вы можете точно понять, что же вам было нужно. Да, это огорчает. Но не обвиняйте себя – именно так все и работает.

Во всяком случае у вас есть возможность что-то исправить. А начать исправление нужно, записав на карточках свои мысли о том, что нужно изменить в программном продукте, чтобы довести его до совершенства. Конечно, все это очень обидно, ведь вы рассчитывали получить хороший результат с первого раза. Но, может быть, Мик Джаггер и прав. Возможно, на самом деле вам нужно было понять, что надеяться оказаться правым с первого раза довольно опрометчиво. Особенно в разработке программного обеспечения.

# Это не для вас

Простите меня. Я припас для вас еще немного новостей – снова плохих.

В реальности тот, кто пишет первую карточку и начинает весь этот цикл, – чаще всего это не тот, кто будет использовать программу каждый день. Поэтому те, кто делал надписи на карточках, и вся работавшая совместно с ними команда могут быть свято уверены в том, что создали идеальное решение проблем конечных пользователей.

Не обманывайте себя.

Если мы команда, особенно если хорошая команда, то мы берем свой продукт и отдаем его на тестирование конечным пользователям. Тестирование не ограничивается демонстрацией. Мы тестируем, наблюдая за тем, как люди пользуются продуктом, решая при этом свои реальные каждодневные задачи.

Случалось ли вам когда-либо наблюдать, как какой-нибудь пользователь работает с продуктом, в создании которого вы участвовали? Вспомните, как это было в первый раз. Как все прошло? Я, конечно, не присутствовал при этом, но почти уверен, что все было совсем не так, как вы ожидали.

Если вы когда-нибудь сидели за компьютером вместе с пользователем и наблюдали за тем, как он использует ваш продукт, вы понимаете, о чем я. Если у вас не было такого опыта, попробуйте.

Для таких тестов вам нужны люди, которые покупают, настраивают и используют ваш продукт с некоторой регулярностью. Часто я жду только, чтобы была готова хотя бы небольшая часть продукта, достаточная лишь для того, чтобы пользователи могли проделать что-то новое, чего нельзя было сделать раньше. Но независимо от того, какую частоту предпочитаете вы, старайтесь, чтобы взаимодействие реального пользователя с вашей системой тестировалось хотя бы раз в пару недель.

Вовсе не требуется, чтобы при пользовательском тестировании присутствовали все члены команды: пользователей будет смущать слишком большое количество наблюдателей. Но все-таки пользовательские тесты вызывают сопереживание – то, чего вы не сможете добиться никаким другим способом. Видеть пользователей, которые работают с вашим продуктом с трудом и без всякого удовольствия, хотя вы были уверены, что все хорошо, – мощнейший мотиватор. Если вы присутствовали при пользовательском тестировании, поделитесь с остальными тем, что видели, излагая им истории.

После такого тестирования вы составите перечень проблем, которые нужно исправить, а также улучшений, которые можно внести. Для каждого из этих элементов нужно приготовить карточку истории со своими идеями по усовершенствованию продукта.

# Разрабатывайте, чтобы изучать

Если вы твердо уверены в том, что использование историй предотвратит создание вашей командой плохих программных продуктов, то вы правы по меньшей мере наполовину. И в самом деле, если несколько умных людей собираются вместе, концентрируются на осознании проблем пользователей и обсуждают, как решить их с помощью программного продукта, который они создают, – это огромный шаг в сторону значительного улучшения этого продукта. Но все-таки надо признать, что разработка программного обеспечения – не работа на конвейере. Вы не просто создаете одинаковые виджеты один за другим, штуку за минуту. Каждая история, которую мы создаем и затем реализуем в наших продуктах, представляет собой что-то новое.

Гуру разработки Agile, уже упоминавшийся здесь мой друг Алистер Коберн, однажды сказал мне: «В бэклог историй нужно отправлять не одну написанную тобой историю, а три».

На мой вопрос «Почему?» он ответил: «Просто делай это».

«Что же мне писать в двух других историях?» – спросил я.

«Не имеет никакого значения».

«Что ты имеешь в виду? – удивился я. – Мне же нужно там *что-то* написать!»

«Ну хорошо, – сказал Алистер, – если тебе так уж нужно, запиши то, что хочешь разработать, на первой карточке, а на второй напиши: “Исправить первую карточку”. Потом на третьей – “Исправить вторую”. Если ты не пройдешь этот цикл трижды для каждой истории, то никогда ничему не научишься».

В традиционном процессе разработки учебу принято связывать с *расширением объема разработки* или *плохими требованиями*. В процессе Agile учеба – это ваша цель. Нужно запланировать, что вы изучите при каждой разработке. Кроме того, нужно заложить в план то, что время от времени вы будете терпеть неудачу.

Стратегия Эрика, которую мы рассматривали в главе 3, помогла ему реализовывать небольшие решения и выполнять итерации до тех пор, пока они не становились жизнеспособными. Эрик получал новые знания с каждым выпуском.

Стратегия *Моны Лизы*, использованная Майком и Аароном из главы 4, помогла им разделить каждую историю на маленькие компактные части, которые, правда, нельзя было представить пользователям, но они позволяли ребятам обучаться быстрее и управлять бюджетом разработки, в результате чего работа была завершена вовремя.

Это две замечательные стратегии обучения. Попробуйте их. Изобретите собственную. Только, пожалуйста, не думайте, что вы правы всегда и во всем! Уверяю, вас ждет великое разочарование.

# Не только программы

В 2011 году Кент Бек – создатель метода историй – начал одну из первых конференций Lean Startup с пересмотра манифеста Agile. Будь на его месте, например, я, это было бы, пожалуй, святотатством; но Кент, один из авторов манифеста, знал, что делает. Он пересмотрел принцип, касающийся работающего ПО, и отныне он читается так: «*Эмпирическое обучение* важнее работающего продукта».

Если вы помните из главы 3, эмпирическое обучение – важнейшая концепция, родившаяся из процессов Lean Startup. Ключевое слово здесь – обучение. *Эмпирическое обучение* означает, что сначала мы обсуждаем, что именно хотим изучить в процессе какой-то разработки, а затем возвращаемся назад и оцениваем *результаты*, по которым видно, узнали мы что-то полезное или нет. Кстати, как оказалось, совсем не обязательно для изучения разрабатывать программный продукт, но, как правило, что-то создать или предпринять какие-то действия все-таки нужно.

Мне нравится использовать истории для управления работой по созданию простых прототипов или планированием пользовательских тестов либо интервью. Мне нравится говорить о том, кто это делает, что нужно сделать и почему. Я люблю сначала договориться о том, что мы делаем, а уж потом приступить к работе. Закончив работу, я анализирую ее результаты, чтобы оценить, удалось ли изучить что-нибудь.

> *Попробуйте метод историй для организации любой работы независимо от того, касается она разработки программных продуктов или нет.*

# Планируйте изучение и учитесь планировать

Карты историй полезны для разбиения большого продукта или идеи функциональности на меньшие части. В главах 3 и 4 описано разделение этих меньших частей на фрагменты, удобные для разработки, притом что каждый фрагмент сконцентрирован на изучении чего-то. Но есть еще один способ дробления большого объема работы, вы должны познакомиться с ним и стараться держать его в голове. С помощью этого способа историю можно преобразовать в план работы, который в итоге обеспечит результат. Об этом мы поговорим в следующей главе.

# Глава 10. Истории готовят как торты

Две недели назад был день рождения моей дочери и нам нужен был торт. У нашей семьи есть собственный кондитер, у которой мы заказываем торты всегда. Не сказать, что мы очень богаты или не умеем печь торты сами, просто Сидни, наш кондитер, делает удивительные, невероятно вкусные торты. Я не знаю точно, какой кулинарной магией она пользуется, чтобы создать такое чудо, но, когда ни спросишь моих детей, какой торт они хотят на день рождения, ответ всегда один: «Мы хотим торт Сидни!» – и от заказа уже не отвертеться.

Чтобы у нас появился торт, я звоню Сидни по телефону. Она всегда спрашивает, для кого торт и по какому случаю. Две недели назад я сказал ей, что Грейс скоро исполнится 12. «А чем она сейчас увлекается?» – спросила Сидни. Мы немного поговорили о Грейс, ее вкусах и интересах. После этого мы обсудили формы тортов, имеющиеся у Сидни, а также подумали, как можно украсить торт, чтобы он был готов вовремя. В этот раз мы остановились на торте в форме птички.

Вот так и работают истории. Сидни задала много вопросов «Кто?», «Что?» и «Почему?». Она осведомилась о контексте – где и когда будет подан торт, сколько человек приглашено. Во время разговора мы рассмотрели несколько разных вариантов. Мы беседовали достаточно долго для того, чтобы выработать единое понимание цели. А поскольку это далеко не первый торт, который мы заказываем у Сидни, у нас уже есть некоторое общее представление о виде и вкусе десерта, который должен попасть к нам на стол. Если бы этого не было, возможно, нам нужно было бы просмотреть несколько фотографий или попробовать несколько кусочков разных тортов, для чего телефона было бы, конечно, недостаточно.

# Начните с рецепта

В ходе беседы Сидни обдумывала, как она будет делать торт. Это было необходимо, иначе она не смогла бы оценить вероятность успеть вовремя. Когда приходит время печь торт, у нее уже есть список шагов, которые нужно проделать: отмерить муку, сахар, масло, яйца, молоко и т. д. Сидни должна взбить, смешать, испечь, украсить и, наверное, выполнить еще много секретных действий, о которых я ничего не знаю. Думаю, у нее есть разные рецепты для разных видов тортов и перечень действий, которые надо выполнить для каждого торта, прежде чем он будет помещен в коробку с бантиком и готов к передаче заказчику. Если бы Сидни написала на бумаге, что ей нужно проделать, у нее получился бы рабочий план, состоящий из специфических шагов по изготовлению тортов.


То же самое происходит, когда кто-то приносит историю в команду разработки. Все вместе принимают решение, какой именно продукт нужно создать, и команда составляет план, содержащий много задач на разработку. В команду разработки, кроме программистов, входят тестировщики, графические дизайнеры, технические писатели и прочие специалисты, чьи навыки необходимы для создания программного обеспечения, поэтому далеко не все задачи связаны с написанием кода. Этот план никогда не составляется при обсуждении истории, так же как и Сидни ничего не планировала, пока не поговорила со мной по телефону. План появится после того, как члены команды ознакомятся с историей, сделают заметки, нарисуют эскизы и уточнят множество деталей. Во всяком случае, так все должно происходить.

Разговаривая с Сидни, я не рассказывал про стаканы сахара и муки. Я никогда не буду излагать истории процесса выпечки, если только не возьмусь разрабатывать духовку. Поэтому, когда вы составляете истории работы с программным продуктом и собираете перечень названий имеющихся историй, вы должны представлять, что ваш программный продукт уже готов. Сидни не погружалась в детали приготовления торта, а просто уточнила, для кого он, что любит моя дочка, сколько людей будет на вечеринке, и выяснила множество других подробностей, которые помогли нам вместе решить, какой торт подойдет лучше

всего. Сидни не просто поинтересовалась моими пожеланиями – фактически мы работали вместе, чтобы решить, как сделать самый лучший торт. Это было самое настоящее обсуждение истории.

# Разделим торт на части

Есть еще одна важная вещь. Проблемы возникают, когда, начав разговор с людьми, которые действительно способны воплотить в жизнь наши идеи, мы обнаруживаем, что программный продукт, описанный историей, *очень* большой. Да, карточка, на которой записана история, того же размера, что и другие, а цель, которую должны достичь пользователи, может быть не важнее остальных, но тем не менее при обсуждении выясняется, что реализация программного продукта, необходимого для достижения этой цели, займет очень много времени.

То же самое могло произойти, когда я обсуждал свой торт с Сидни. Я мог представлять себе какой-то особенный торт, форм для которого у Сидни бы не оказалось, или украшения, которые невозможно сделать. Я не смог бы позволить себе такой торт, а Сидни не смогла бы обещать, что изготовит его ко дню рождения моей дочери.

В главе 7 я сказал, что, если решение, которое мы приняли, оказалось слишком дорогим, нужно сделать шаг назад и реалистично оценить проблемы, которые мы пытаемся решить, и результаты, которые надеемся получить. Затем нужно рассмотреть имеющиеся альтернативы. Таким образом, у нас будет торт поменьше или, может быть, просто пирог.

> *Если решение, описанное в истории, слишком дорого, рассмотрите другое, которое поможет вам достичь цели.*

Но если мы все-таки способны реализовать решение, несмотря на его значительный размер, то в разбиении на более мелкие части особого смысла нет, не так ли? На самом деле не так. Последовательно выпуская небольшие части программного продукта, мы быстрее можем увидеть и проанализировать прогресс. Это позволяет людям, которые платят деньги, чувствовать себя чуть спокойнее. Кроме того, согласно *стратегии Моны Лизы* из главы 4, это помогает частично оценивать продукт на ранних этапах, чтобы понять, в правильном ли направлении мы движемся.

> *Если решение, описанное в истории, требует большого объема работы, но все же реализуемо, разбейте его на небольшие части, что позволит вам раньше оценить результаты и анализировать прогресс*.

В разбиении больших историй на меньшие части есть небольшая хитрость, которая позволяет мне придерживаться аналогии с тортом. Если вы любите торты, то, наверное, у вас уже текут слюнки? Наверняка вы представляете себе какой-то особенно вкусный десерт. Простите меня за это.

Так вот, давайте представим, что наша история описывает создание очень большого торта, например многоярусного свадебного, которым будут угощаться несколько сотен людей. Если это так, то речь идет не о стаканах муки и сахара, а о *мешках*. Большинство людей разделит программный продукт на части аналогичным образом. Получатся не мелкие задачи по графическому дизайну, бизнес-логике и взаимодействию с базой данных, а огромные пласты работы. Но вспомните: программный продукт все-таки не торт. Отмерить два фунта муки ненамного дольше, чем два стакана, но создание 20 экранов пользовательского интерфейса займет куда больше времени, чем разработка двух. Поэтому если команда

использует для разбиения подобную простую структуру, что на первый взгляд логично, то в результате получатся задачи на разработку бизнес-логики, интерфейса и прочего, каждая из которых требует нескольких недель труда. Прибегнув к такой стратегии, мы будем вынуждены ждать очень долго, прежде чем сможем, образно говоря, попробовать *хоть какой-нибудь* торт. Не поступайте так.

> *Не разбивайте большие задачи на большие части. Разбивайте большие задачи на мелкие части с компактными планами.*

Сейчас наша аналогия немного затрещит по швам, но вскоре будет восстановлена. Способ, который сработает в отношении большого программного торта, – это разбиение его на множество маленьких капкейков[22]. Каждый из них является законченным продуктом, а кроме того, рецепты всех примерно одинаковы: немного сахара, немного муки, одно-два яйца и т. д.


Ладно, давайте наконец настроимся серьезно. Программный продукт не торт. Он может разрастись до невероятных размеров, потребовать огромных денежных затрат и вызвать серьезные риски. Работая над этим текстом, несколько минут назад я услышал в утренних новостях по телевизору историю о том, как пользователи не могли зарегистрироваться на правительственном сайте США по вопросам здравоохранения. Конечно, критиковать постфактум легко, но все-таки нетрудно понять, что никто не попробовал этот торт, прежде чем он был подан на воображаемой свадьбе, никто не знал даже примерно, каков он на вкус. В результате этот полусырой десерт провалил всю вечеринку.

Если вы когда-нибудь участвовали в разработке программного обеспечения традиционным способом, то, наверное, привыкли разбивать большую работу на большие

задачи. У меня есть такой опыт. На первый взгляд кажется противоестественным разбивать что-то большое на небольшие части, которые не совсем похожи на конечный продукт, запланированный вами. Вы знаете, что для комбинирования всех этих частей в один продукт нужно будет немного поработать, слегка переписав и скорректировав каждый кусочек, чтобы их можно было объединить. Но помните: у этого подхода очень хорошие основы. Одна из главных, на которых базируется его выбор, – избегание рисков, связанных с тем, что вы видите, используете, пробуете новый продукт слишком поздно. Вы разбиваете продукт на много маленьких продуктов для того, чтобы начать изучение поскорее.

Если бы мне нужно было разбить на части большой торт, чтобы поскорей его попробовать и оценить внешний вид, мне отлично подошло бы изготовление маленьких капкейков, которые быстро дали бы результаты. Скажем, я испек бы несколько штук из разных видов муки, попробовал их все, отметил самый вкусный и был бы уверен, что сделал хороший выбор. Если бы меня волновали цвет и украшения, я хотел бы увидеть несколько капкейков, оформленных в разных стилях, и выбрал бы самый красивый из них.

В производстве программ капкейкам соответствуют порции работающего программного обеспечения, которые позволяют пользователям оценить, могут ли они эффективно решить свои задачи. Это могут быть фрагменты, позволяющие выявить технические риски. И каждая часть даст нам новые знания.

Но горка капкейков не может быть свадебным тортом… или может?[23]

Программный продукт не торт. Поэтому все кусочки программы, которые мы выпускаем, *складываются* в один большой работающий продукт, чего не получится в случае торта.

Мой друг Люк Хоуман привел забавную поговорку: «Половина готового торта и наполовину готовый торт – не одно и то же». Половины готового пирога может быть недостаточно для всех гостей на свадьбе, но ее хватит, чтобы все смогли отведать его и с нетерпением ждать добавки.

# Глава 11. Как дробить камни

Изначальная идея историй очень проста: запишите что-то на карточке, обсудите это, согласуйте, что будете разрабатывать. А после завершения разработки закончите цикл анализом того, что у вас вышло и чему это можно научить. Вот и все – не правда ли, просто? Если вы хоть немного работали в сфере разработки программного обеспечения, то знаете, что ничего простого тут нет. Истории проходят долгий пусть с множеством обсуждений, вовлечением множества людей, чтобы сначала продвинуть идею продукта, функциональности или улучшения, а потом продвинуть этот продукт на рынке. Хорошая новость в том, что вы можете использовать истории и их изложение на протяжении всего пути. И я обещаю, что, положившись на метод историй, вы *окажете* себе значительную услугу.

# Размер имеет значение

В предыдущей главе мы остановились на торте Сидни и идее разделения больших тортов на маленькие тортики. Но программный продукт – несколько менее осязаемая вещь, и, в отличие от торта, его нельзя измерить в дюймах, сантиметрах, унциях или граммах.

Изначальная идея заключалась в том, что пользователь или другое лицо, которое в чем-то нуждается, может записать свою потребность на карточке, а затем мы можем это обсудить. Тот, от кого поступил запрос, не заботился о том, много ли времени потребуется на решение его проблемы, – он просто сформулировал потребность как она есть.

> *С точки зрения пользователя размер истории верен, если он удовлетворяет его потребность.*

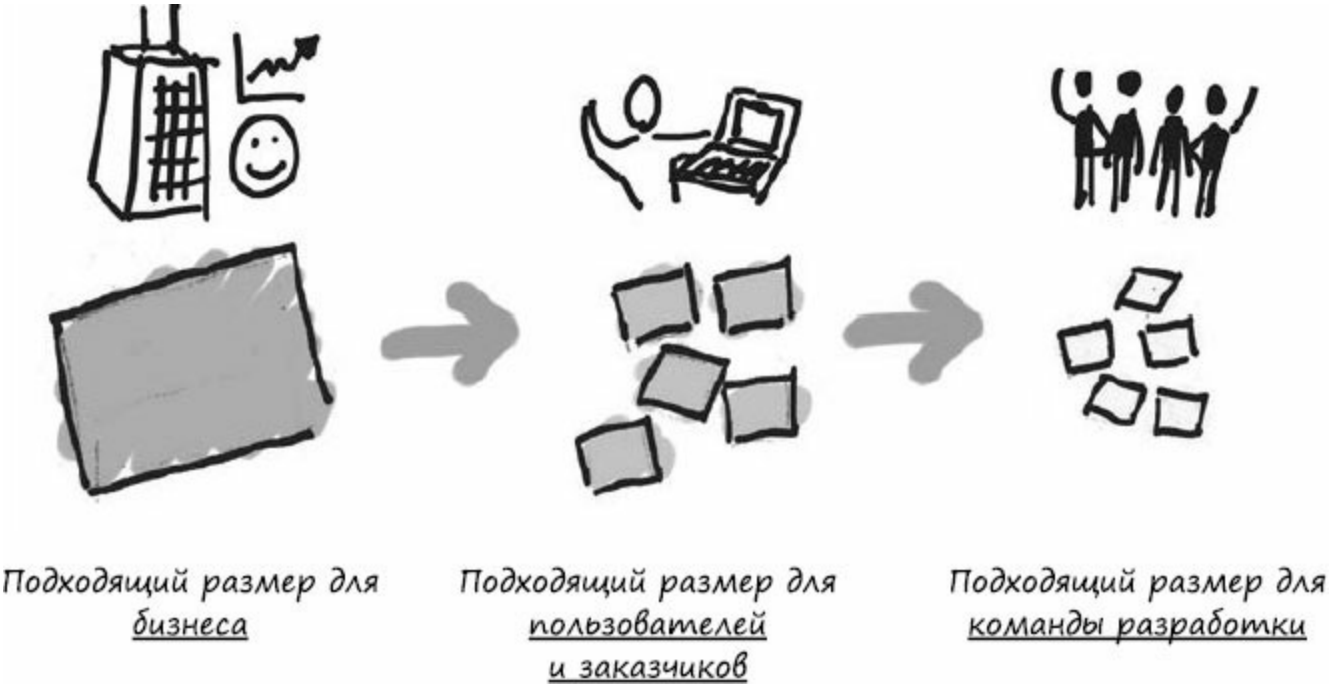
Когда приходит время писать код, можно оценить значительные преимущества программирования, тестирования и интеграции программного продукта по небольшим частям. Если я смогу вскоре увидеть и протестировать маленькие части, значит, смогу и судить о том, как быстро продвигается наша разработка и каков ее уровень качества. Если я могу разделить что-то большое на много маленьких частей, членам моей команды легче брать в работу и выпускать эти части одновременно. Можно взять за правило делить истории на части такого размера, чтобы на их программирование и тестирование требовалось не более нескольких дней.

> *С точки зрения команды разработки, размер истории верен, если программирование и тестирование занимает не более пары дней.*

Но с точки зрения бизнеса может быть более разумно предъявлять пользователям и заказчикам программное обеспечение, содержащее набор нескольких функциональностей. Если вы выпускаете совершенно новый продукт, первая версия может быть довольно большой. Эту версию я ранее называл минимально жизнеспособным решением, и основной целью его разработки было получение особых результатов от выделенной группы пользователей. В идеале бизнес должен стремиться к тому, чтобы почаще и побольше выпускать таких продуктов, размер которых оптимален для пользователей или для получения результатов, отвечающих потребностям бизнеса. Но если вашим продуктом пользуется большая разношерстная группа клиентов и у вас нет инфраструктуры или бизнес-модели, поддерживающей более последовательный процесс релиза, то, конечно, ваши бизнес-релизы могут быть довольно большими.

> *С точки зрения бизнеса размер истории верен, если он позволяет бизнесу достичь желаемых результатов.*

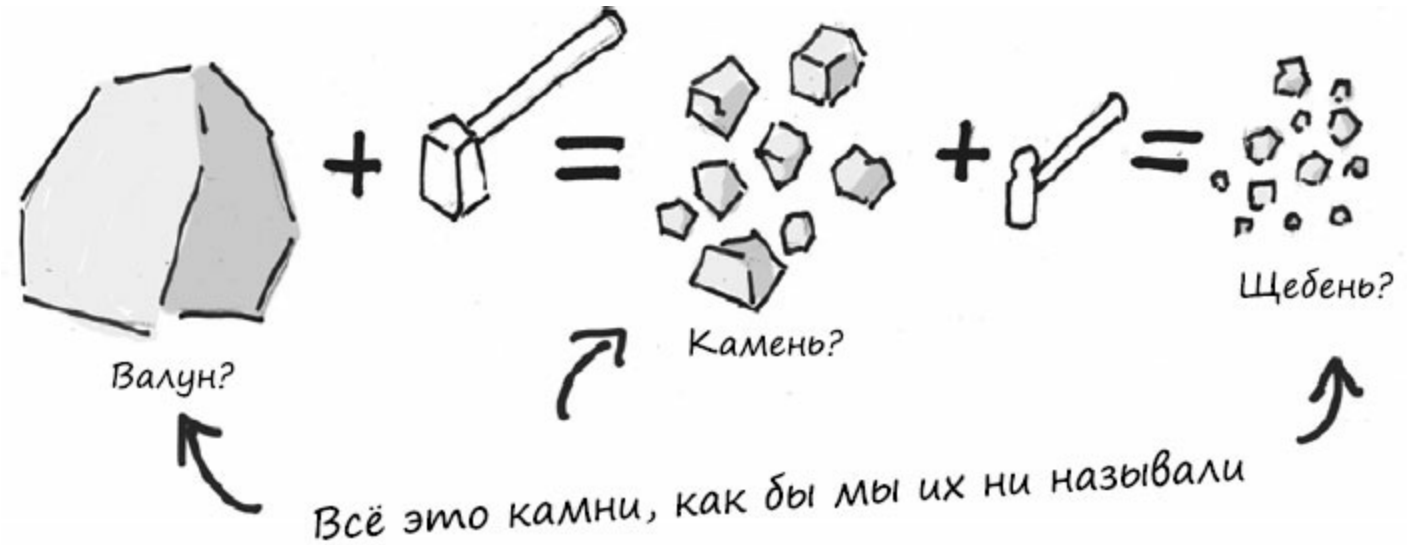
Я мог бы сказать, что правильного размера историй не существует, но это не так. Правильный размер – тот, который соответствует проходящему обсуждению.




Эти большие истории содержат много меньших, которые, в свою очередь, содержат много еще меньших историй. В зависимости от того, с кем вы говорите, возможно, придется перевести обсуждение на более высокий уровень.

# Истории похожи на камни

Воспринимайте истории как камни. Если я возьму очень большой камень, положу его посреди комнаты, а затем разобью молотом на 30 осколков, мы назовем их 30 осколками камня. Если вы возьмете один из этих меньших камней и тоже ударите по нему молотом, он разобьется на более мелкие части. Мы и их можем назвать осколками камня. Или подойти творчески и употребить, скажем, слова «валун» или «щебень». Но я не знаю, когда именно что-то перестает быть валуном и становится просто камнем. Он, может быть, и выглядит как простой камень, пока не упадет вам на ногу – тогда-то вы и почувствуете, что это настоящий валун.




Мой инструмент для разбивания камней – большой молот. И он отлично работает.

Большие истории разбиваются на меньшие истории, и эти меньшие истории тоже могут быть разбиты на совсем маленькие. Но любого размера, даже самого маленького, это все равно история. А какой инструмент лучше всего применить для разбивания историй? Правильно: обсуждение. Иногда достаточно просто немного поразмыслить, но если вы обсуждаете историю и работаете над ней совместно с кем-то еще, то одновременно расширяете одинаковое понимание.

> *Обсуждение – один из лучших инструментов для разделения больших историй на части.*

Многие работники сферы IT, и я не исключение, недовольны отсутствием точности в этом аспекте. В большинстве организаций, с которыми я работал, появлялся свой язык для классификации историй по размеру, но в таком случае мы опять-таки упираемся в проблему «валун или щебень». Точность особенно важна, когда этот камень прилетел в вас, что может объяснить, почему программисты так трепетно относятся к классификации своих историй.


Если вы работаете над языком классификации в своей организации, не старайтесь достичь большой точности. Определения истории и ее верного размера довольно неустойчивы, но это позволяет проявлять определенную гибкость при их использовании на протяжении всего цикла разработки.

# Эпики – большие камни, которыми иногда кидаются в людей

*Эпик* – общепринятый термин (я не знаю, кто первым употребил его) для описания крупных пользовательских историй, так же как слово «валун» используется для обозначения очень большого камня. Буду с вами честен: мне понадобились годы, чтобы воспринимать важные вещи, которые мы разрабатываем, как *истории*, но сейчас я понимаю, почему они так называются. Термин «эпик» смущает меня до сих пор. Насколько я помню, учитель английской литературы объяснял, что эпические поэмы обычно повествуют о героях, борющихся со злом, – Беовульфе, Ахилле, Фродо – обычно с помощью какого-то волшебного оружия или при поддержке богов. Но, кажется, я отвлекся…

> *Эпик – это история, размер которой считают значительным, и, значит, она должна быть разбита на меньшие части.*

Неплохо иметь специальное название для больших историй, но погодите немного. Термин «эпик» нередко используется в качестве оружия. Я нередко наблюдал, как член команды разработки убеждает представителя бизнеса, менеджера продукта, пользователя или кого-то еще, кто о чем-то просит, что его история – это целый *эпик*, а не просто история. Обычно это произносится таким тоном, словно автор истории сделал что-то не так и нанес члену команды разработки серьезное оскорбление. Поэтому, если вы член команды разработки, не используйте термин «эпик», чтобы кого-то пристыдить. Это плохое начало (и, вероятно, преждевременное окончание) того, что должно было быть продуктивным обсуждением.


Помните, эпики – это большие истории, размер которых может быть правильным с точки зрения бизнеса, пользователей или заказчиков, но не команды разработки. Поэтому поработайте вместе над их разбиением, но не откладывайте в сторону сам эпик. Он вам понадобится, когда вы станете обсуждать с другими людьми и его, и более детализированные истории, на которые он был разделен.

Если вы применяете электронную систему, которая поддерживает разработку Agile, то, скорее всего, там используется концепция эпика как большой родительской истории, на которой основаны более мелкие дочерние истории.

# Группу историй организует тема

Используйте термин «тема» для описания историй, которые удобно объединить в группу. Разбив камень – разделив большие истории на части и организуя их в продукты, которые люди хотят получить и использовать, – вы получите множество маленьких историй. Я воспринимаю тему как мешок, в который можно сложить набор историй, связанных друг с другом. Я могу использовать этот термин также для набора историй, необходимых для следующего релиза, являющихся частью одной функциональности, соответствующих определенному типу пользователя или связанных каким-то иным образом. Но тогда моя метафора не совсем верна, так как одна и та же история может относиться к двум разным темам, а вот камень лежать в двух разных мешках не может.

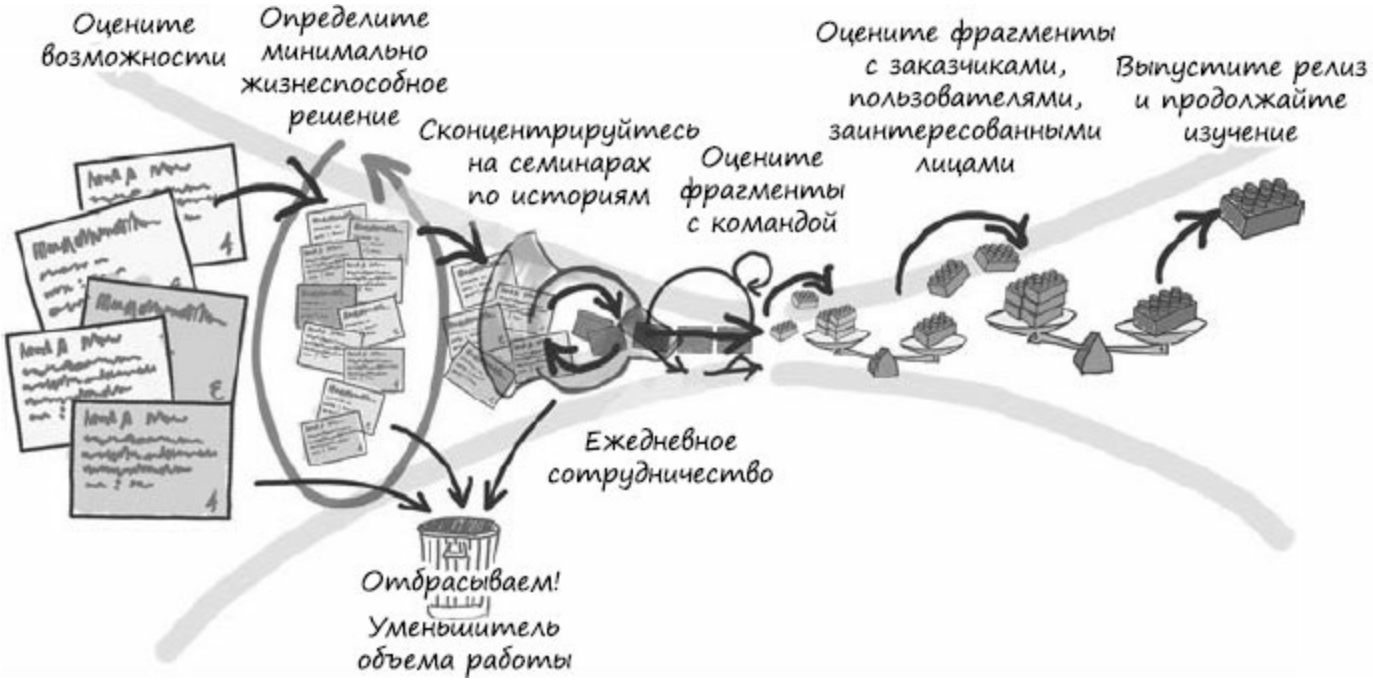

Если вы пользуетесь одним из доступных сейчас инструментов, которые поддерживают объединение в группы историй Agile, возможно, в них поддерживается и организация историй по темам. Вы можете ссылаться на тему, какой бы она ни была: следующий релиз, функциональность или истории, соответствующие определенному типу пользователей.

# Забудьте все эти термины и сконцентрируйтесь на изложении историй

Термины *«эпик»* и *«тема»* давно проникли в инструменты для управления жизненным циклом Agile, в некоторые подходы к разработке Agile со своими наименованиями, а также в общепринятый язык для обсуждения историй. По этой причине вы должны знать и понимать их.

Но сейчас давайте отойдем от этих терминов: забудьте, что я упоминал их, хотя бы ненадолго. Сделаем шаг назад и посмотрим на весь цикл разбиения камней. В этом цикле мы начинаем с великих идей, на которые можно смотреть как на большие камни, и проводим их по всему пути к небольшим фрагментам работающего программного продукта. Затем перебираем эти фрагменты программы в функциональности, продукты и релизы, которые нужны пользователям и заказчикам.

Вот как может выглядеть весь цикл разбиения камней.




А сейчас разберемся со всем этим подробнее.

# Начните с возможностей

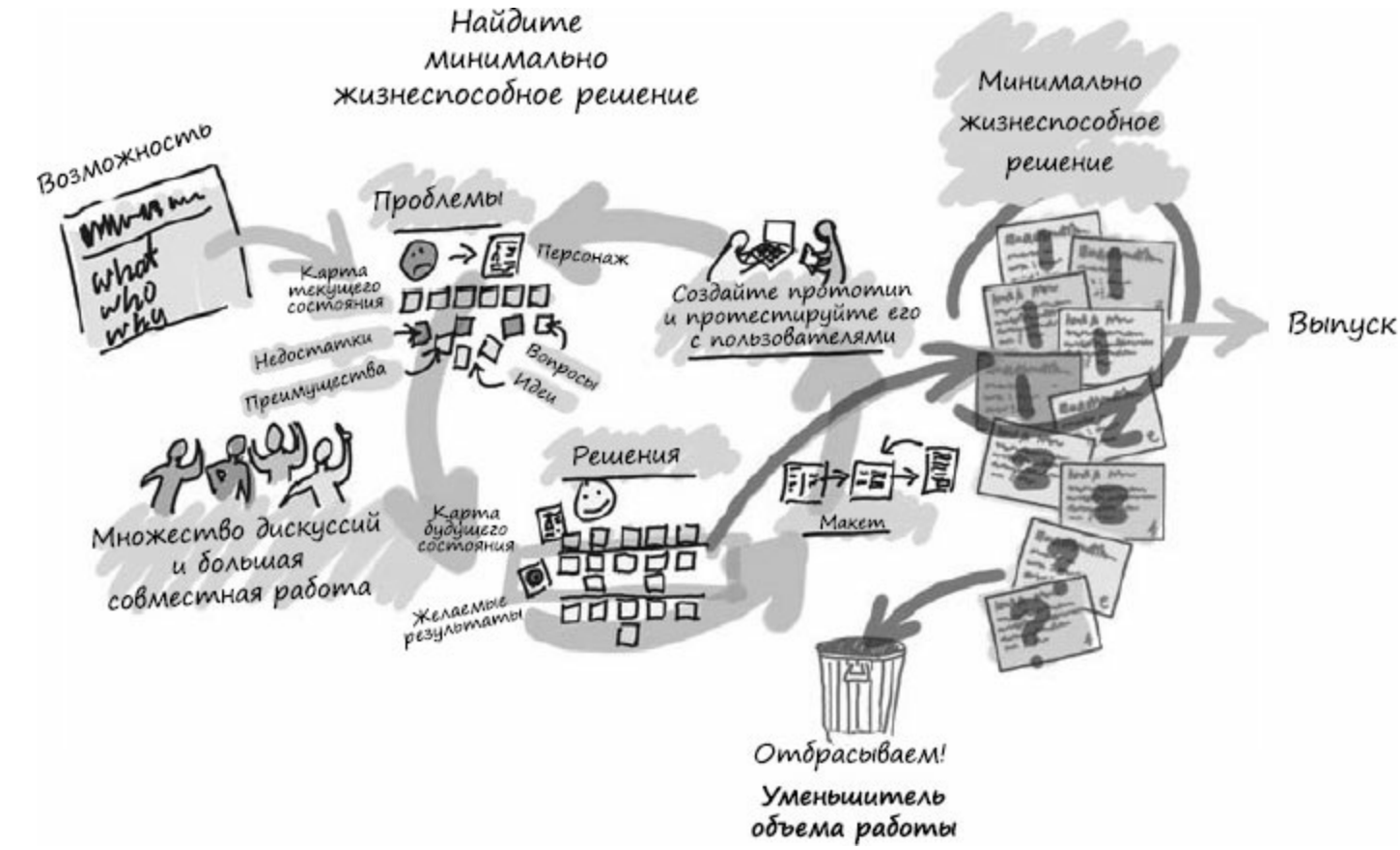
Странствие истории начинается с идеи. Это может быть мысль о новой функциональности или совершенно новом продукте. Это может быть какое-то изменение, улучшающее уже существующую функциональность. Это может быть проблема, которую нужно решить. Но я все же использую термин *«возможность»*, поскольку у нас есть благоприятная возможность получить какие-то выгоды в результате этих действий. Предлагаю вам составить список этих возможностей – я называю его *бэклогом возможностей*.


Первое качественное обсуждение истории – высокоуровневая («Кто?», «Что?», «Почему?») дискуссия, ее основная цель – принять решение: «да» или «нет». Причем «да» не означает, что мы немедленно приступаем к разработке. Это значит: пойдем дальше, проводя углубленные обсуждения, чтобы полностью понять историю. Но я не хочу двигаться вперед и тратить много времени на разговоры, если выяснится, что идея изначально для нас не годится. «Нет» – вежливый способ сказать: «Ерунда». Поэтому давайте назовем этот этап «решение того, развивать идею или отбросить ее». Помните: нам всегда требуется разработать больше, чем мы способны, так что отбросить бесперспективную возможность прежде, чем она съест слишком много времени и сил, – стоящее решение.

*Обсуждайте возможности, чтобы удостовериться в целесообразности решения проблемы и принять решение, развивать его или отбросить.*

# Найдите минимально жизнеспособное решение

После того как вы приняли решение двигаться вперед, настало время углубиться в исследования. Старайтесь узнать побольше, чтобы найти решение, достойное реализации. И не забудьте его всерьез минимизировать. Стремитесь как можно значительнее уменьшить объем требуемой работы, но при этом получить максимальный положительный эффект.




Во время исследования вы должны углубиться в следующие вопросы.

• Кто те заказчики или пользователи, которым, по вашему мнению, пригодится это решение?

• Как они решают свои задачи сейчас, пока продукта еще нет?

• Как изменится их мир, когда вы покажете свое решение?

• Какие внешний вид и способ взаимодействия должны быть у вашего решения?

• Сколько времени займет реализация?

Существует много практик, которые могут оказаться полезными в этой дискуссии, в частности карты историй. Они могут помочь вам осознать, как люди работают сегодня, а затем соотнести ваши идеи с состоянием всей системы после реализации решения.

Во время исследования очень важно проверить свои предположения и проделать работу по их подтверждению. Это можно сделать, глубоко проанализировав бизнес-правила или внешние предписания. Можно также поработать некоторое время непосредственно с заказчиками и пользователями, чтобы понять, как они это делают. В рамках этой работы прототипы вашего решения должны быть созданы и проверены на целевой аудитории. Кроме того, постройте технические прототипы для того, чтобы можно было выявить технические риски.

*Изучайте и обсуждайте, чтобы найти небольшое и жизнеспособное решение.*

Радуйтесь каждому пункту вашего решения, благополучно отправленному в корзину или обратно в бэклог возможностей, чтобы разобраться с ним в другой раз. Наши возможности могут представлять собой очень большие камни, но внутри этих камней порой таятся бриллианты и драгоценные металлы. Разбейте камни и извлеките оттуда действительно ценное, не обращая внимания на простую породу, а затем порадуйтесь, что отбросили все ненужное.

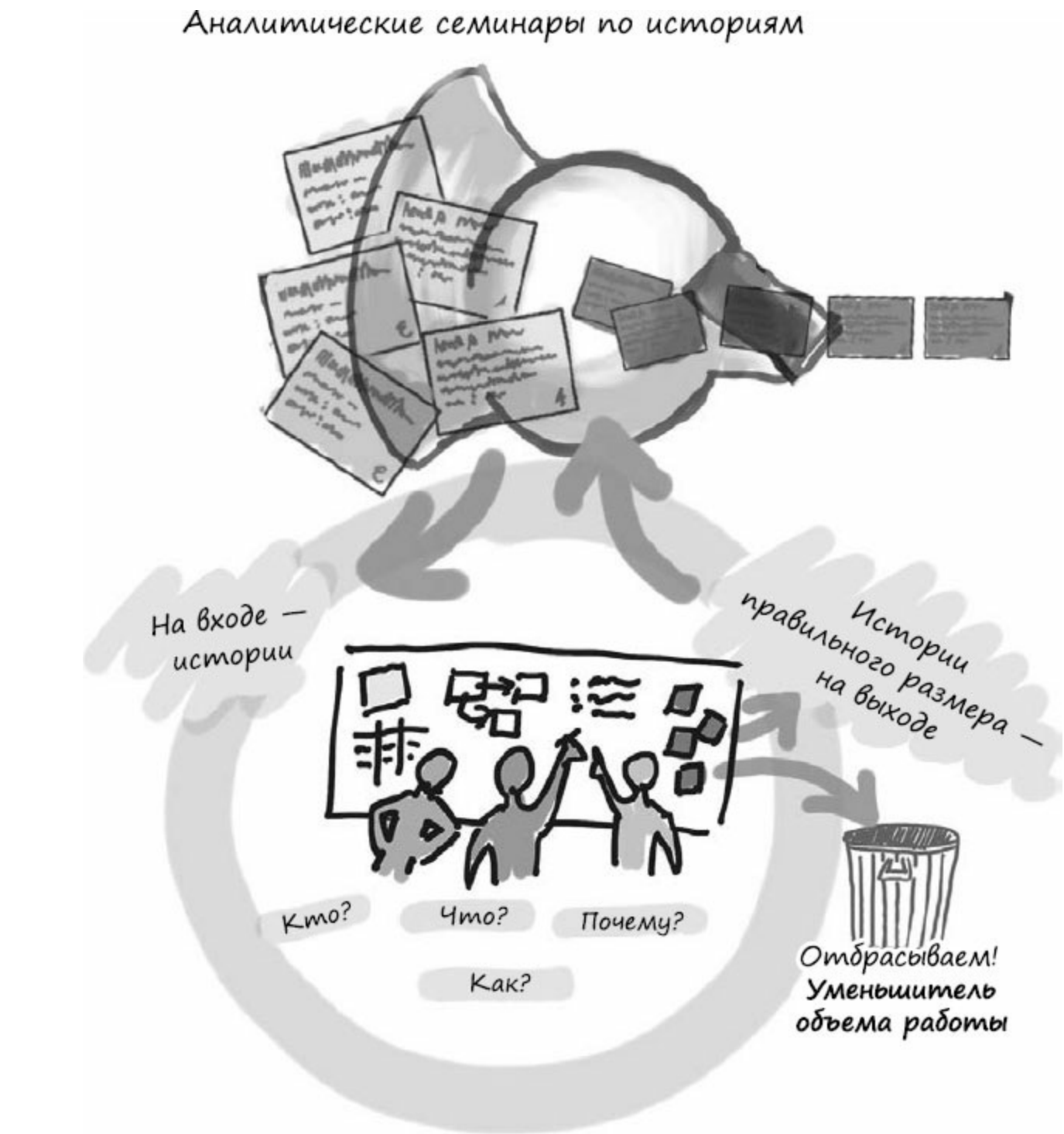
Во время исследования вы можете принять решение о создании небольших вещей, которые помогут вам в изучении, в частности прототипов архитектуры или графического дизайна.

*Спайк* – термин, используемый для небольших фрагментов программы или исследования, которые мы разрабатываем с целью изучить что-то конкретное. Своим происхождением он обязан сообществу экстремального программирования, где так называли работу, которая могла не войти непосредственно в создаваемый продукт. Используйте истории, чтобы описывать спайки, которые будет создавать ваша команда с целью что-либо изучить.

Когда вы уверены, что выделили небольшой набор историй, которые должны реализовать и предъявить пользователям и заказчикам, настало время подойти поближе к релизу. Список историй, реализация которых даст релиз полезного продукта, называется релизным *бэклогом*.

# Погрузитесь в детали каждой истории во время разработки

Наши возможности поначалу могли быть похожи на большие камни. Исследовательские обсуждения разбивают их на части и отделяют пустую породу от ценных металлов. Но разработка может идти быстрее и эффективнее, если нам удастся разбить эти части на как можно более мелкие фрагменты (помня о том, что каждый фрагмент должен быть чем-то, что мы можем создать и таким образом нечто изучить). Чтобы сделать это, понадобится еще больше обсуждений, в ходе которых мы углубимся в подробности и детали.




Я нарисовал здесь замечательную камнедробилку: с одной стороны я загружаю в нее эти большие необработанные камни, внутри которых есть ценные металлы, а с другой стороны получаю камни, идеально подходящие по размеру для следующего цикла разработки. Я

называл бы этот агрегат *машиной семинаров по историям* – мне кажется, это название отлично описывает то, что мы делаем.

Мы проводим углубленные обсуждения с программистами и тестировщиками, а также другими членами команды, вовлеченными в разработку продукта, чтобы очень, очень тщательно проанализировать детали. Это наши последние и самые продуктивные обсуждения, на них мы должны прийти к соглашению относительно способов подтверждения готовности или критериев приемки для небольших частей нашего программного продукта, ведь следующим шагом будет их разработка. Поскольку известно, что на этих обсуждениях мы делим проект на части, используйте их, чтобы привести истории к правильным размеру и форме, пригодным для отправки на следующий спринт разработки или итерацию.

> *Используйте аналитические семинары по историям, чтобы обсудить детали, разбить истории на части и прийти к реалистичным и конкретным соглашениям относительно того, что именно вы создаете.*

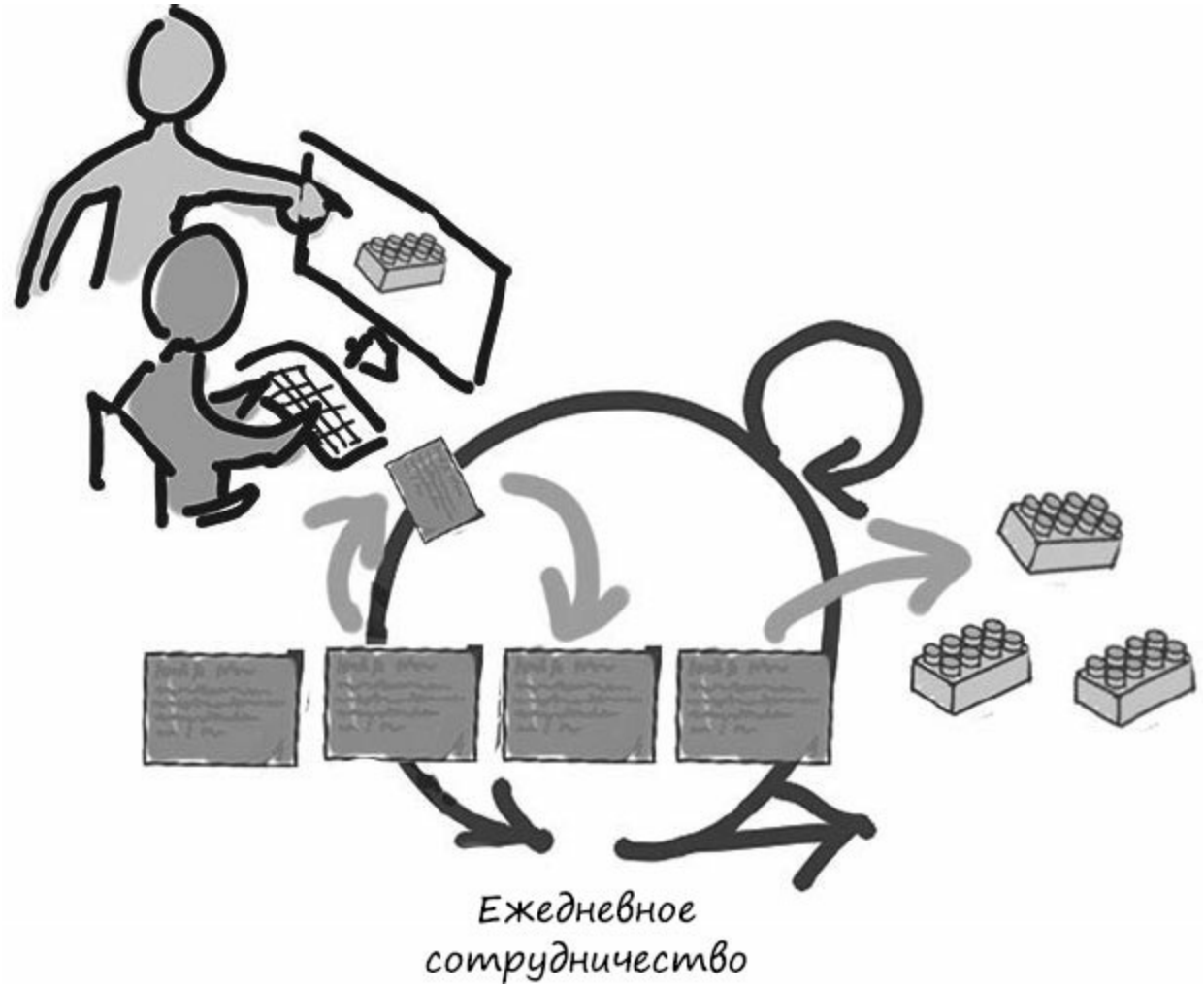
Я люблю называть эти последние обсуждения *семинарами по историям*, потому что каждый знает: совещания непродуктивны, а семинары придуманы для того, чтобы сделать определенную работу. Можно проводить их столько, сколько нужно, хоть каждый день. Иногда достаточно лишь одного в течение сессии планирования. В Scrum, процессе Agile, такие семинары можно проводить в ходе приведения бэклога в порядок. Но когда бы вы ни решили провести эти обсуждения, они непременно должны состояться.

# Продолжайте обсуждать в процессе разработки

Машина семинаров по историям производит продукт для следующего аппарата на конвейере – машины разработки Agile. Здесь я нарисовал реальный процесс работы этой машины: с одной стороны мы загружаем в нее маленькие истории правильного размера, а с другой стороны получаем идеально работающий маленький кусочек программного продукта – или того, что описывает ваша история.

Несмотря на то что вы работали очень усердно, даже последние продуктивные обсуждения не предскажут всего, что вы узнаете, начав разработку. Поэтому запланируйте ежедневные спонтанные обсуждения. Отведите время для них на ежедневных планерках.

Если вы разработчик и у вас в какой-то момент появились вопросы, а деталей, которые вы узнали из обсуждений истории, недостаточно для ответов, найдите кого-то, чтобы продолжить дискуссию. Не стоит винить в этом плохие требования. Вспомните, что раньше, перед началом разработки, вы работали вместе с коллегами, чтобы определить все необходимое для нее. Но мы люди, и никто не может всего предусмотреть.



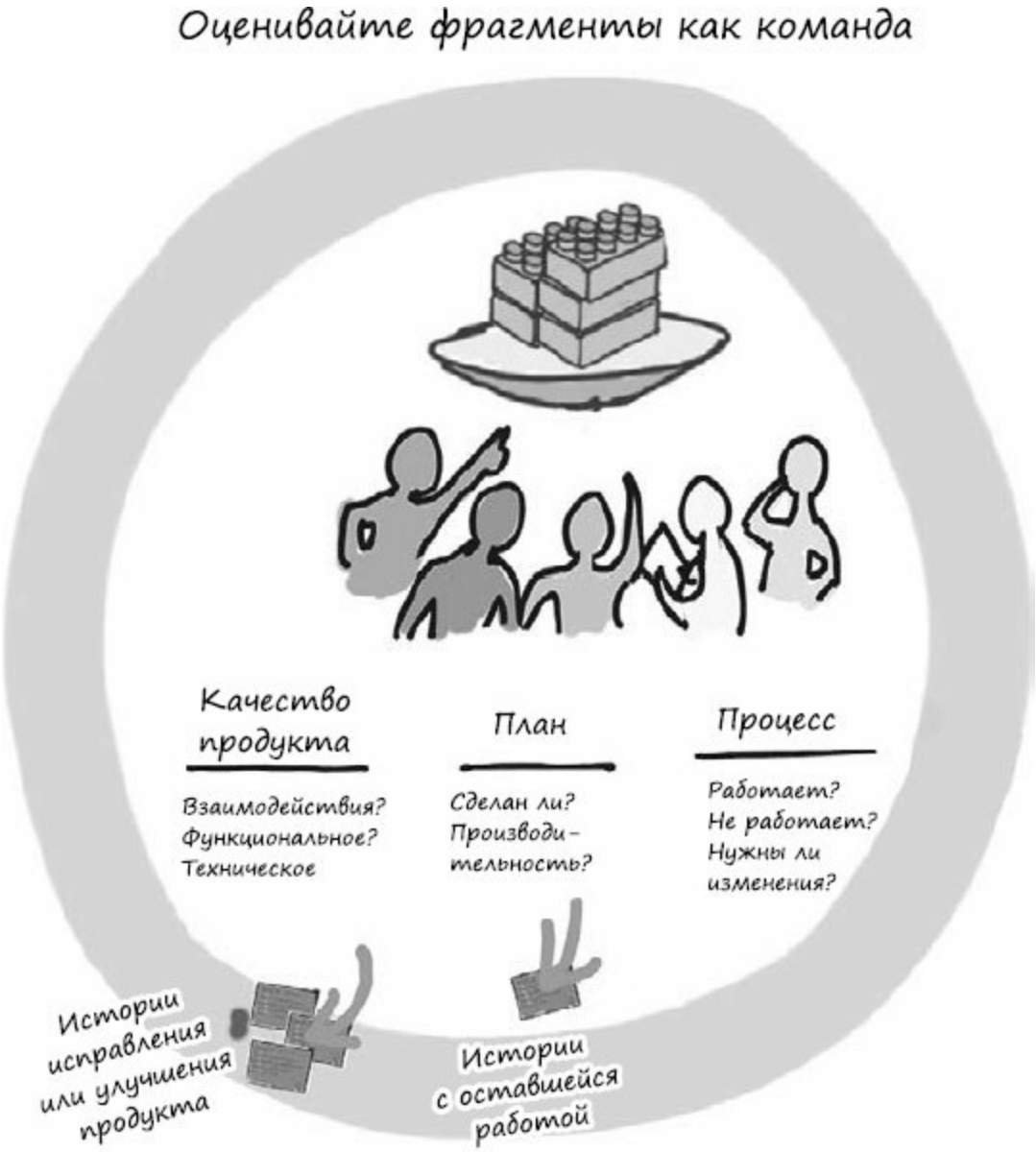

Если вы владелец продукта, дизайнер UX, бизнес-аналитик или еще кто-то из членов команды, которая решала, что нужно создавать, не стесняйтесь пойти в отдел разработки и посмотреть, как идут дела. Обещаю: чем раньше вы увидите что-то работающее, тем раньше поймете что-то важное. Кроме того, возможно, создателю этого работающего фрагмента не помешает отзыв о его детище.

*На протяжении процесса разработки не прекращайте обсуждения, чтобы уточнить детали и высказать свое мнение по поводу создаваемого продукта.*

# Оценивайте каждую часть

Как только готовые фрагменты работающего продукта выкатятся из машины Agile, всем, кто принимал решение, что создавать, и всем, кто создавал, следует взять паузу и внимательно оценить результат.

Помните, в реальности речь идет не о машинах. Ни вы, ни ваши коллеги не являетесь винтиками. А все фрагменты, которые вы выпускаете, – это не один и тот же виджет. Они абсолютно разные.




Остановитесь и реально оцените качество готового решения. Спросите себя, насколько эффективным было планирование. Создано ли то, чего вы ожидали? Заняло ли это намного больше времени, чем планировалось? Или, может, меньше? Или примерно столько, сколько вы наметили? Кроме того, откровенно обсудите, насколько хорошо работает «машина». Сейчас подходящий момент, чтобы внести корректировки или изменения в процесс работы, чтобы продукт стал более качественным.

*Почаще оценивайте качество продукта, планирования и процесса работы.*

Этот первый этап оценки в Scrum называется *ревизией спринта* или *ретроспективой*. Как бы вы ни называли моменты остановки, пересмотра и размышлений, главное, чтобы они у вас были.

# Оценивайте с участием пользователей и заказчиков

Никогда не забывайте: все, что вы разрабатываете, вы делаете не для себя – во всяком случае, не всегда. Вам нужно оказаться лицом к лицу с заказчиками и пользователями, чтобы узнать, как они оценивают вашу работу. Иногда можно показать им всего лишь прототип, или макет, или текстовое описание, но возможность увидеть и опробовать нечто реально работающее позволяет понять, верной ли дорогой вы движетесь.

Но отдельный маленький кусочек, выходящий из машины Agile, может быть маловат для демонстрации пользователям. Я представляю себе, как все эти маленькие фрагменты один за другим укладываются на чашу весов – старомодных весов с грузом на другой чаше. На ней лежит слово «достаточно», что означает: достаточно и для тестов с участием пользователей и заказчиков, и для нас, чтобы что-то узнать.


Обычно «достаточно» означает готовый экран или последовательность экранов, которые дают пользователю возможность выполнить задачу или достичь значимой цели. И я не хочу разговаривать с пользователями. Я наблюдаю за ними не для того, чтобы сказать: «Это прекрасно». Я наблюдаю для того, чтобы изучать, и результат изучения, как правило, примерно такой: «Получилось не совсем верно» или «Сейчас, когда я использую это в

реальности, то вижу, что будет лучше, если…»

*Обучайтесь, тестируя законченные фрагменты продукта с привлечением заказчиков и пользователей.*

# Оценивайте вместе с ключевыми партнерами

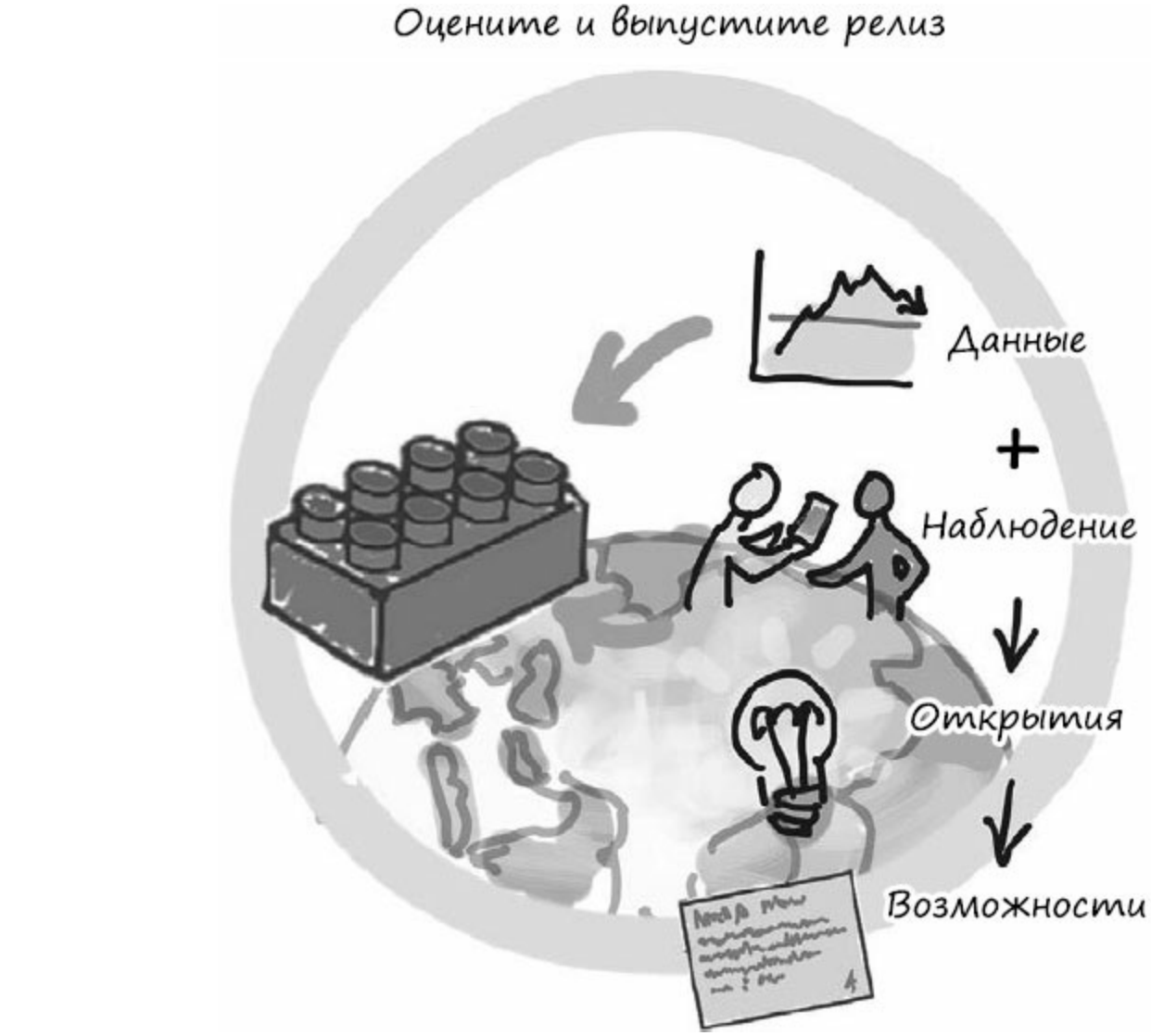
В вашей организации наверняка есть люди, очень заинтересованные в программном продукте, который вы создаете. Они не обязательно являются его пользователями, но им важно, чтобы этот продукт был представлен пользователям, и чем раньше, тем лучше, по крайней мере в заявленные вами сроки.


Познакомьте их с продуктом и продемонстрируйте его. В рамках демонстрации расскажите, на какой стадии своего большого плана вы сейчас находитесь. Помните: им, вероятно, не очень интересно знать, соответствует ли плану кучка разрозненных кусочков, которые вы создали. Что им интересно, так это прогресс в создании минимально жизнеспособного решения, потому что это самая малая часть, которую вы можете выпустить и получить какой-то полезный отклик от внешнего мира. Так что поговорите об этом. Поделитесь результатами тестов с пользователями и заказчиками, которые вы получили. Вашим партнерам тоже полезно узнать что-то новое.

*Обеспечивайте наглядность процесса работы и улучшения качества для других заинтересованных сторон в своей организации.*

# Выпустите релиз и продолжайте оценивать

Я нарисую еще одни, последние весы. На одну чашу мы кладем части, которые проанализировали, протестировали с участием заказчиков и пользователей и продемонстрировали заинтересованным лицам в своей компании. Эта чаша очень похожа на ту, что упоминалась несколько шагов назад, на стадии тестирования с пользователями и заказчиками. А на другой чаше снова лежит слово «достаточно», но в этот раз оно означает «достаточно для предъявления заказчикам и пользователям и получения нужных результатов». Как только чаши сравнялись, выпускайте релиз.




Но не останавливайтесь на этом этапе. Здесь все еще есть возможности для исследования. Если бы вы были похожи на Эрика из главы 3, обучение было бы вашей главной целью и вам были бы нужны параметры, показывающие, как людям работается с вашим продуктом и используется ли он вообще. Но вам нужно лично поговорить, чтобы выяснить, почему они работают или не работают с ним. Если вы предполагаете, что люди будут применять ваш продукт, а вы и ваша компания получите от этого выгоду, не ограничивайтесь простыми утверждениями – используйте измерения и обсуждение, чтобы узнать, что происходит в действительности.

*Используйте измерения, общайтесь с пользователями, чтобы узнать, получены ли в действительности ожидаемые результаты*.

Если рассматривать лишь проект, ваша работа уже закончена – вы его предъявили. Но пока вы просто сделали некий продукт. А жизнь продукта *начинается*, когда он представлен пользователям. Обещаю: начав обращать внимание на то, что люди делают с вашим продуктом, вы обязательно увидите возможности его немного улучшить. Запишите их и примените в начале описанной здесь модели.

Это и есть *реальный* жизненный цикл – во всяком случае, жизненный цикл истории.

Камней здесь разбивается много. Но кто конкретно ответственен за их дробление? Я рад, что вы это спросили, ведь именно об этом следующая глава.

# Глава 12. Берем в руки камнедробилку

В практике работы по модели Agile совершенно напрасно закрепилась традиция назначать кого-то одного ответственным за написание историй и проведение всех обсуждений. В Scrum, процессе Agile, этот человек называется владельцем продукта. Однако есть по меньшей мере две серьезные причины, по которым это не работает, а может, и целый ряд более мелких причин.

**Основная причина № 1.** Чтобы пройти весь путь от неясной идеи до маленьких конкретных фрагментов программы, поступающих в разработку, требуется очень много обсуждений. Один человек просто не может их все обеспечить. И если вы построите процесс так, что он зависит от одного человека, этот человек может стать «бутылочным горлышком» и, несомненно, очень скоро им станет.

**Основная причина № 2.** Один человек не обладает всеми нужными компетенциями и не сможет посмотреть со всех возможных точек зрения, чтобы прийти к наилучшему решению. Только сотрудничество людей с различными навыками позволяет создать по-настоящему полезный продукт.

> *Владелец продукта не может в одиночку написать все истории и присутствовать на всех обсуждениях. Это не работает.*

Не поймите меня неправильно. С моей точки зрения, при правильной разработке продукта лидерство владельца продукта очень важно. Он следит за тем, чтобы в ходе работы вся команда сосредоточилась на движении в верном направлении.

Альтернативой может быть проектирование комиссией – еще одна ужасная устоявшаяся практика, где каждый получает равные права при принятии решений о том, что мы делаем. Работая как комиссия, имея время и ресурсы на то, чтобы сделать какую-либо одну вещь, мы вынуждены идти на компромисс. Мы с моей бывшей женой часто прибегали к этому способу, выбирая ресторан. Она любила морепродукты, я – мексиканскую кухню, и в итоге мы отправлялись туда, где не нравилось нам обоим. Когда же комиссия не имеет ограничений ни в ресурсах, ни во времени, мы делаем все, что только можем. Вы, наверное, сталкивались с программными продуктами такого типа: в продукте бессчетное количество функций и вы вечно мучаетесь, пытаясь отыскать среди них нужное или вспомнить, как его надо использовать.

Эффективно действующие владельцы продукта окружают себя людьми, которые нужны им для принятия удачных решений. Они объединяют компетенции, мнение и опыт многих людей. Но в условиях ограниченных ресурсов, когда успех продукта находится под угрозой, решения приходится принимать им, и, конечно, всегда найдутся недовольные. Моя подруга Лиза Райшелт хорошо сказала: «…В дизайне нет ничего от демократии»[24].


Может быть опасно неочевидным тот факт, что для поиска решения в центральном секторе нужно сотрудничество между людьми, которые хорошо знают бизнес, заказчиков, пользователей и технологии, которые мы используем, – и не просто знают, а берут на себя ответственность за успех решения. Эти люди действительно общаются с партнерами по бизнесу, заказчиками и пользователями, они действительно проектируют и тестируют пользовательские интерфейсы, они реально пишут и тестируют код, который заставляет продукт работать.

Помните, что большим недоразумением в разработке Agile является ситуация, когда владелец продукта или менеджер в одиночку решают, что нужно создавать. Редко – если не никогда – один человек обладает навыками ведения бизнеса, разработки дизайна пользовательского интерфейса и инженерными навыками, необходимыми для обнаружения центральной точки «полезное – реализуемое – удобное». Вот почему в самых эффективных организациях работают небольшие кросс-функциональные команды, которые совместно разрабатывают верное решение. Как говорилось в предыдущей главе, исследование стоит воспринимать как дробление камней. Это и есть та работа, которую мы делаем, чтобы превратить историю из большой неясной идеи во что-то маленькое и конкретное, что мы способны воплотить в жизнь.

*Работой по исследованию продукта должна дирижировать маленькая кросс-функциональная команда под руководством владельца продукта.*

Идеальный размер такой команды – от двух до четырех человек, численность компании за обеденным столом, поэтому ее члены легко могут выработать одинаковое понимание проблемы.

Командой должен руководить менеджер продукта или владелец продукта, который глубоко понимает бизнес и бизнес-процессы, а также хорошо знает сегмент рынка, к которому относится продукт. В команду следует включить человека, хорошо знающего пользователей, умеющего общаться с ними, чтобы разобраться в процессах их работы, а также рисовать эскизы и создавать простые прототипы графического интерфейса. Кроме того, необходим ведущий программист из команды, которая будет разрабатывать продукт. Этот человек должен знать архитектуру системы и иметь представление о новейших инженерных подходах, которые могут помочь в решении сложных задач. Дело в том, что

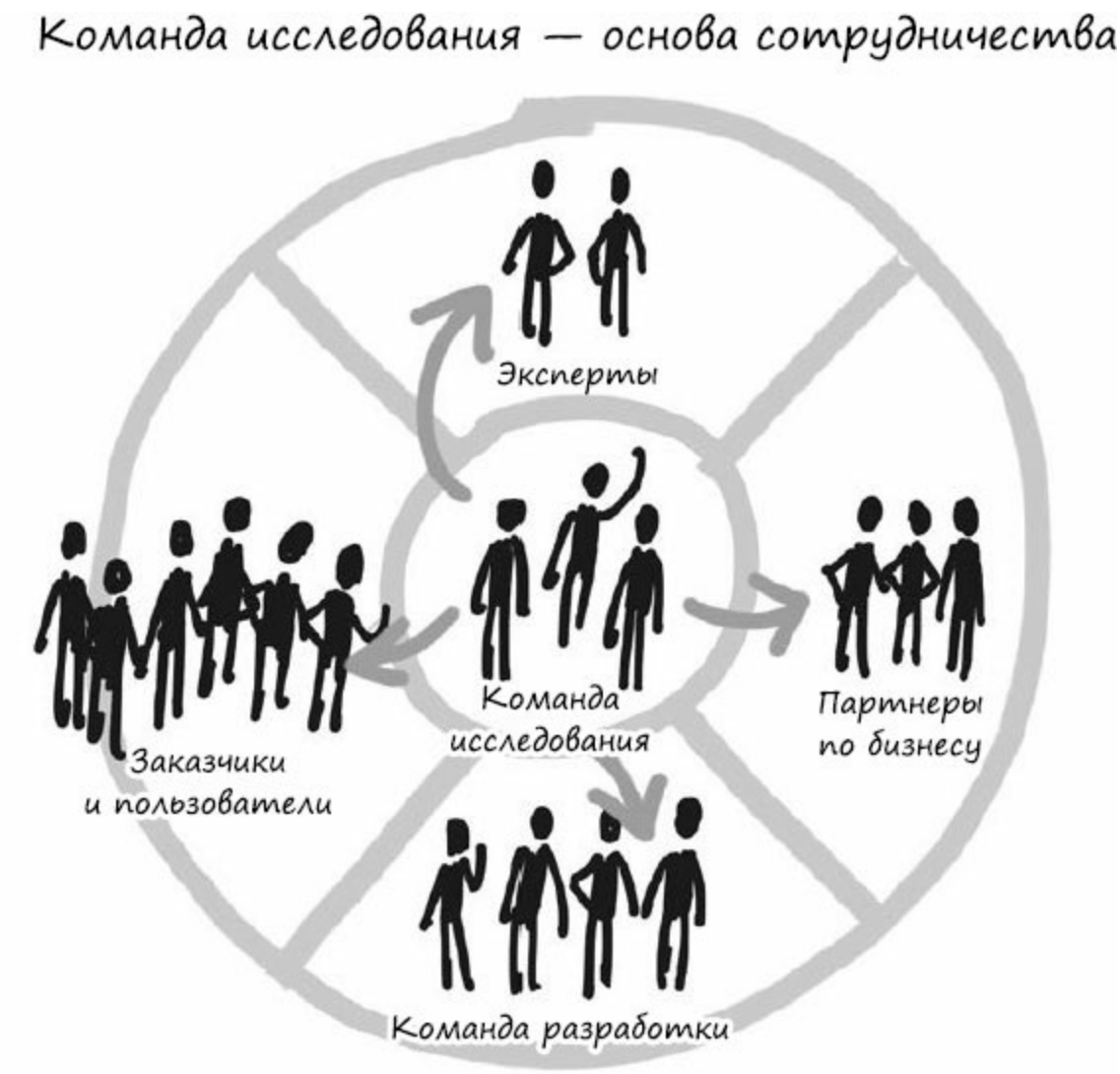
самые инновационные решения часто исходят от инженеров, вдохновленных сведениями о проблемах пользователей и задачах бизнеса.


Сплоченная команда исследования продукта представляет собой мощную, слаженно работающую группу экспертов, способную быстро обнаруживать проблемы и находить решения. Я часто слышу, что эту ключевую команду называют *триадой*. Когда я последний раз был в сиднейском офисе Atlassian, Шериф, с которым вы познакомились в главе 8, указал мне на три рабочих места, расположенных рядышком. «Здесь сидит триада», – объяснил он. Вокруг мест триады стояли рабочие столы с компьютерами, за которыми располагалась остальная команда. Мне также приходилось слышать, как *триадой* называли команду, где было два человека, четыре человека или даже больше, ибо три концепта, о которых мы говорим, – полезное, удобное, реалистичное – не соотносятся с тремя людьми.

> *Владельцу продукта должна помогать ключевая команда, куда входят люди, компетентные во взаимодействии с пользователями, дизайне и программировании.*

# Для успешной работы ключевой команде нужна помощь многих других

Эффективное исследование требует совместной работы не только с командой разработки, но и с заинтересованными людьми бизнеса, экспертами в предметной области, заказчиками и конечными пользователями. Это сложная работа, которая требует первоклассных навыков коммуникации и сотрудничества, а не только конкретных навыков в своей сфере, которыми обладают члены команды.




А сейчас будет настоящий секрет. Для создания продукта любой значимости нужна команда. Чтобы сохранить ясное видение продукта, убедиться, что запланированное решение реализуемо, и задавать общее движение в заданном направлении, наличие сильного лидера критически важно. Самые лучшие лидеры обеспечивают такие условия, чтобы каждый мог внести свою лепту в принятие решений. В дружественной среде, управляемой историями, вы увидите, что множество обсуждений историй идет не прекращаясь. И большинство из них не требует присутствия лидера.

# Three Amigos

¡Three Amigos! – название заурядного вестерна-комедии 1986 года со Стивом Мартином, Чеви Чейзом и Мартином Шортом в главных ролях. Что общего у этого фильма и разработки по методологии Agile, управляемой историями? Там есть тройка ловких и расчетливых героев, что очень здорово для семинаров по историям. Я не знаю, кто первым назвал их тремя амиго, но, кажется, это прозвище ушло в народ[25] (уверен, если бы фильм посмотрело больше людей, этого не произошло бы).

Как вы, должно быть, помните, под *семинарами по историям* я подразумеваю самые поздние и продуктивные обсуждения, в ходе которых принимаются конкретные решения о том, что нужно создавать. Именно тогда вступают в дело трое амиго.

В течение последних обсуждений нужно предусмотреть множество деталей и вариантов реализации, поэтому нужен разработчик из команды, которая будет писать код, – в идеале один из программистов, который будет непосредственно работать над задачей.

Чтобы маленький фрагмент программы мог считаться законченным, его нужно протестировать, поэтому необходимо, чтобы в беседе участвовал тестировщик. Тестировщик – *первый* амиго – часто привносит в дискуссию критический взгляд, выявляя то, что в дальнейшем может привести к ошибкам и сбоям. Как правило, тестировщик лучше всех играет в «Что, если».

И конечно, нужен кто-то, понимающий, что, для кого и почему мы делаем, то есть специалист из команды исследования продукта. Этот человек – *второй* амиго.

На этой стадии, как правило, не происходит представление новой функциональности, это должно было произойти раньше, на этапе исследования. Сейчас мы уже так или иначе решили что-то делать, поэтому очень важно понять, как должна выглядеть программа и как она будет себя вести. По этой причине очень часто к обсуждению привлекают дизайнера пользовательского интерфейса или бизнес-аналитика, который может работать со всеми этими деталями. Это *третий* амиго.


Эта группа проработает все детали и придет к соглашению относительно конкретных критериев приемки истории. Как правило, на этом обсуждении оценивается время, которое потребуется на разработку и тестирование программы. Часто здесь также принимают решение о разделении истории на небольшие истории для разработки верного размера, то есть такие, на написание кода и тестов которых потребуется от одного до трех дней.

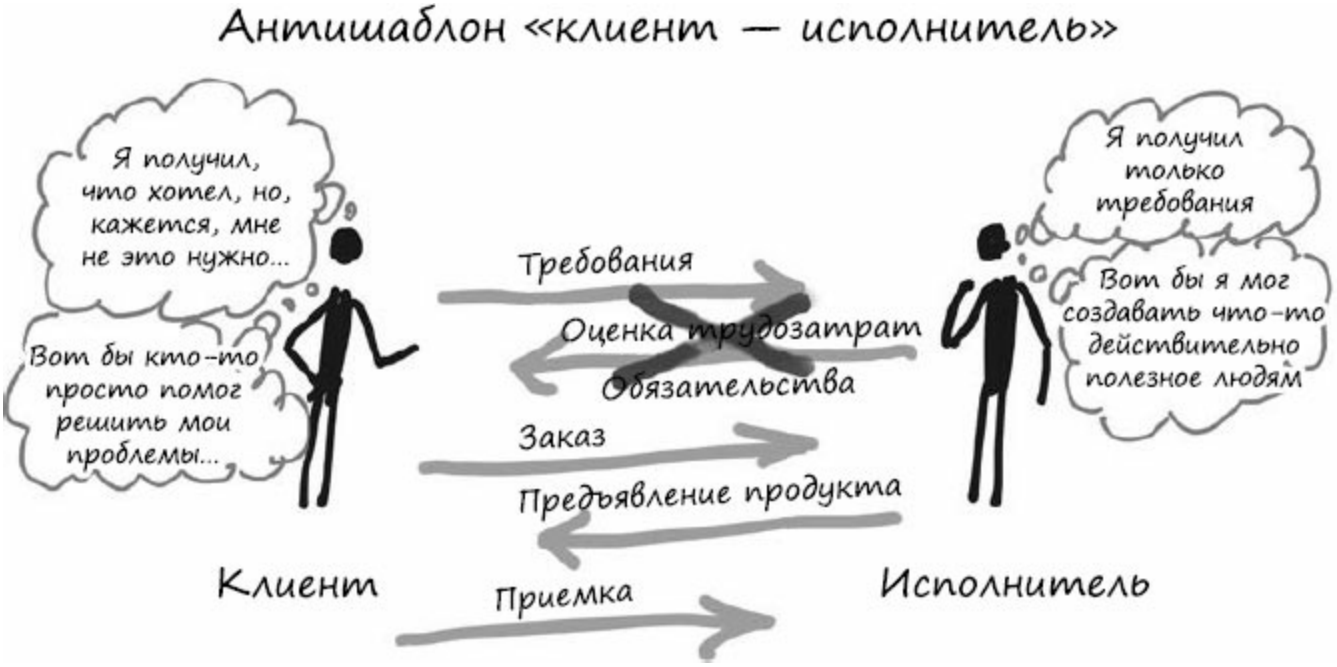
Истории постоянно обсуждают в процессе движения идеи от одной стадии разработки к другой. На каждом обсуждении не выходите за рамки «полезное», «удобное» и «реалистичное». Привлекайте людей, которые могут дать оценку этим факторам. Избегайте «дизайна в комиссии», возлагая на владельца продукта ответственность за удачный и цельный продукт.

### *Антишаблон «клиент – исполнитель»*

Существует вредный антишаблон, который часто мешает хорошей работе пользовательских историй. Фактически его применение может привести к тому, что в результате совместной работы не получится *вообще ничего* хорошего.

В этом антишаблоне один участник играет роль клиента, а другой – исполнителя. Задача клиента – знать, чего он хочет, и объяснить исполнителю все детали. Это мы называем требованиями. Работа исполнителя – слушать, вникать, а затем продумать технический подход к реализации того, что хочет клиент. После этого исполнитель оценивает трудозатраты, что в сфере разработки программ

часто является синонимом обязательства (кстати, этим часто объясняется страх программистов перед необходимостью давать оценки без тщательного анализа).




Конец истории печален и вполне предсказуем.

Изредка, конечно, оценка снайперски точна, клиент получает именно то, что просил, а просил он именно то, что ему было нужно.

Но в большинстве случаев реализация решения занимает больше времени, чем предсказывал исполнитель. Работник, являющийся исполнителем, может приводить любые оправдания, в том числе, что в требованиях, которые ему дали, плохо проработаны детали или просто «требования никуда не годятся». Клиент может жаловаться на неточную оценку, и, кажется, никому не приходит в голову, что это оксюморон. Когда же решение наконец предъявлено и клиент получает то, что просил, он может попробовать полученное и обнаруживает, что это совсем не то, что ему было нужно, намеченной им цели достичь нельзя.

Самое ужасное здесь то, что клиент понимает свою проблему лучше всех, а вот найти ее решение у него получается хуже, чем у других. А исполнитель отлично понимает технологии, и чаще всего только он обладает достаточной квалификацией, чтобы решить проблему, потому что он работает с этими технологиями и знает, как они могут помочь в данном случае. Более того, большинство технических специалистов искренне хотят помочь, им было бы приятно знать, что результаты их труда полезны людям.

Но в антишаблоне «клиент – исполнитель» обсуждение проблем и решений заменено дискуссиями и соглашениями относительно требований. Проигрывают в такой ситуации все.

Одна из целей метода историй – избавиться от этого антишаблона.

Хороший пример отношений, разрушающих этот антишаблон, – отношения многих из нас со своим врачом. Попробуйте явиться в кабинет врача и предъявить ему свои «требования». Перечислите рецепты, которые вам нужно выписать, и

процедуры, которые следует назначить. Если у вас хороший врач, он ответит: «Очень интересно, а сейчас расскажите мне, что вас беспокоит».

Я представляю себе диапазон, на одном конце которого слово «официант», а на другом – «врач». Вам нужно стремиться к тому, чтобы ваши рабочие отношения больше напоминали общение врача и пациента, а не официанта и гостя.

# Владелец продукта как продюсер

Если ваша организация работает в традиционной манере разработки программного обеспечения, в ней могут не совсем ясно понимать, чем должен заниматься владелец продукта. Например, вы участвуете в разработке критически важных систем для банков. Банк знает, что реальные продукты – это банковские услуги, которые он продает своим клиентам. Если у вас есть сотрудник на должности менеджера продукта, то его работа – отвечать за определенный тип банковского счета или кредитного продукта. Компьютерные системы, которые поддерживают этот тип услуги, всего лишь шестеренки в большом механизме. Зачастую бывает, что одна и так же IT-инфраструктура поддерживает множество различных банковских продуктов. Понятно, что банк не рассматривает эту инфраструктуру как продукт, и, как правило, нет никого, кто может выступать в роли ее владельца.

В таких типах организаций бизнес-аналитики часто исполняют обязанности по «сбору требований». Они являются передаточным звеном между программистами и представителями бизнеса, например менеджерами банковского или страхового продукта. Когда этим представителям бизнеса нужны изменения IT-инфраструктуры, которая поддерживает их продукт, они работают с бизнес-аналитиками, чтобы сформулировать эти изменения. Очевидно, что в данном случае они выступают в роли клиентов, а бизнес-аналитики – в роли исполнителей, в результате чего и запускается описанный ранее антишаблон.

Как-то в одном разговоре мой друг Дэвид Хассман предложил лучшую метафору для отношений, которые на самом деле должны существовать между бизнес-аналитиками и представителями бизнеса, – они должны быть такими же, как отношения между музыкальным продюсером и его группой. Дэвид знает в этом толк, потому что он не только гуру Agile, но и экс-гитарист хеви-метал-группы Slave Raider, существовавшей в 1980-х годах. Ему приходилось как работать с продюсерами, так и быть продюсером самому. Группа вступает в шоу-бизнес с энтузиазмом и, хочется надеяться, некоторым талантом, но участники не разбираются в подводных камнях шоу-бизнеса или в процессе записи альбома. В этом компетентен продюсер. Это его работа – помочь группе выпустить настолько успешный альбом, насколько это возможно. Хорошие продюсеры могут вырастить из талантливого новичка популярного, коммерчески успешного артиста.

Такова и работа бизнес-аналитика в IT. Используйте видение представителей бизнеса и помогите им воплотить его в успех. Вы не можете работать как приемщик заказов – вы должны выступать в роли доктора. Иногда это означает необходимость объяснять представителям бизнеса то, что они не хотят слышать. Но если вы искренне хотите помочь им добиться успеха, они заметят это и оценят вашу помощь.

> *Будучи владельцем продукта в отношении идей ваших партнеров, выступайте продюсером, который поможет им добиться успеха.*

Здесь кроется еще один потенциальный антишаблон, выражающийся в том, что представителя бизнеса пытаются заставить исполнять роль владельца продукта. Я сказал *«потенциальный»* потому, что это, в принципе, может и сработать, если представитель бизнеса получает помощь и поддержку других членов команды, готов учиться работе владельца продукта и имеет время, чтобы заниматься ею. Владение продуктом – обязанность

весьма нетривиальная, не стоит рассматривать ее как нечто необременительное, что можно делать в свободное время. Поэтому советую не возлагать на представителей бизнеса дополнительную работу, а найти продюсера, который поможет им добиться успеха.

# Это сложно

Для идеи, весьма простой в своей основе, вся эта история с историями становится слишком запутанной на практике. Если кто-нибудь когда-нибудь говорил вам, что разработка программного обеспечения – да и предъявление *любого* продукта – прошла легко и просто, это было вранье.

Истории объединяют в себе много разных вещей. Мы употребляем это слово, подразумевая карточку, фрагмент программного продукта, который разрабатываем, и особенно способ обсуждения, которое проводим, чтобы решить, что нужно создать. Истории могут описывать очень большие возможности или сложно формулируемые части предоставления продукта, которые сами по себе могут не значить ничего особенного для заказчиков или пользователей. Работа с историями – продолжительный процесс обсуждений и дискуссий, в результате чего большие истории должны быть разбиты на маленькие части. На протяжении всех этих обсуждений мы концентрируемся не только на том, что можем создать, но также для кого и почему это делаем. Карты историй – лишь один из способов облегчить разбиение больших частей на маленькие, при котором мы не упускаем из виду людей, использующих продукт с удовольствием и пользой для себя.

Если все это начинает обретать для вас смысл, значит, ваше мировоззрение изменилось, что очень важно и совершенно необходимо. Это шаг не в сторону использования историй для документирования требований, а к более эффективной работе с людьми, совместному фокусированию на решении реальных проблем с помощью продуктов, которые вы создаете.

Я надеюсь, вы тоже думаете, что это замечательно.

# Глава 13. Начните с возможностей

Разрешите мне вернуться к сходству историй с камнями. Части, на которые вы разобьете камень, все равно можно назвать камнями, просто меньшего размера. Но это все еще такие же камни, как тот, первоначальный, который нам нужно было внимательно рассмотреть, чтобы понять, достоин он разбиения или нет. Давайте условно назовем его «нулевой камень». В терминах метода историй я бы употребил слово «возможность».

Под возможностями я подразумеваю идеи, которые, по нашему мнению, могли бы решить проблему. Нет, я вовсе не тот человек, чей стакан всегда наполовину пуст, но все-таки не очень умно считать любую возникшую идею достойной реализации в нашем продукте, ведь, как вы знаете, времени и людей, чтобы реализовать все, никогда не бывает достаточно. Но даже если бы у вас было сколько угодно ресурсов, ваши заказчики были бы слегка шокированы таким продуктом.

# Обсуждайте свои возможности

Когда мы предлагаем для обсуждения какую-нибудь идею, обычно эта идея довольно велика, хоть и не всегда. На жаргоне историй мы можем назвать такую идею *эпиком*, но я предпочитаю слово *«возможность»*. Независимо от того, как вы их называете, это все еще истории, и главная цель их начальных обсуждений – решить, стоит ли продолжать работу над этими историями или просто отбросить их. Для каждой возможности мы можем обсудить:

• для кого она. На этом уровне чаще всего упоминаются разные группы пользователей, заказчиков или целевой сегмент рынка;

• какие проблемы она может решить. Нужно разобраться, какие проблемы всех типов пользователей мы можем решить. А также как они решают свои проблемы сейчас – вручную, или с помощью продуктов наших конкурентов, или, что еще хуже, с помощью продукта, который неудобен для них сейчас и неэффективен;

• что представляет собой идея. У нас могут быть соображения о том, какими должны быть продукт или функциональность. Об этом следует поговорить;

• почему именно эта возможность. Хороший момент для обсуждения причин, почему вашей организации выгодно создавать программные продукты для этих пользователей. Чаще всего решение проблем пользователей оказывается недостаточно весомым аргументом. Нужно предусмотреть отдачу от инвестиций в программное решение, а также соответствует ли эта инвестиция действующей бизнес-стратегии. Я не говорю, что нужно вычислять рентабельность инвестиций, так как на данной стадии, имея всего лишь истории, это сложно сделать. Просто обсудите в общем, какую пользу получит организация, если решение будет реализовано;

• каков ее размер. На этом уровне, даже если размах идей велик, мы можем сделать осторожные оценки времени на разработку, пусть даже они и не будут особенно точными. Лучше всего рассмотреть данную возможность в сравнении с чем-то, что мы уже делали раньше: «Это звучит похоже на ту, другую функцию, которую мы выпустили в прошлом релизе. На нее потратили несколько недель, значит, и эта займет примерно столько же». Для принятия решения о том, стоит ли продолжать работу над идеей, полезно знать, сколько времени займет ее разработка – дни, недели или месяцы.

Полный набор таких историй называется *бэклогом возможностей*. Мы пока не знаем точно, стоит ли реализовывать их или, по крайней мере, не нужно ли их выбросить. Помните: идей функций или продуктов, которые можно создать, всегда больше, чем времени. Поэтому ищите возможности, согласующиеся с бизнес-стратегией своей организации, а также решающие проблемы пользователей или заказчиков. Проведите достаточно обсуждений, чтобы принять решение «да» или «нет».

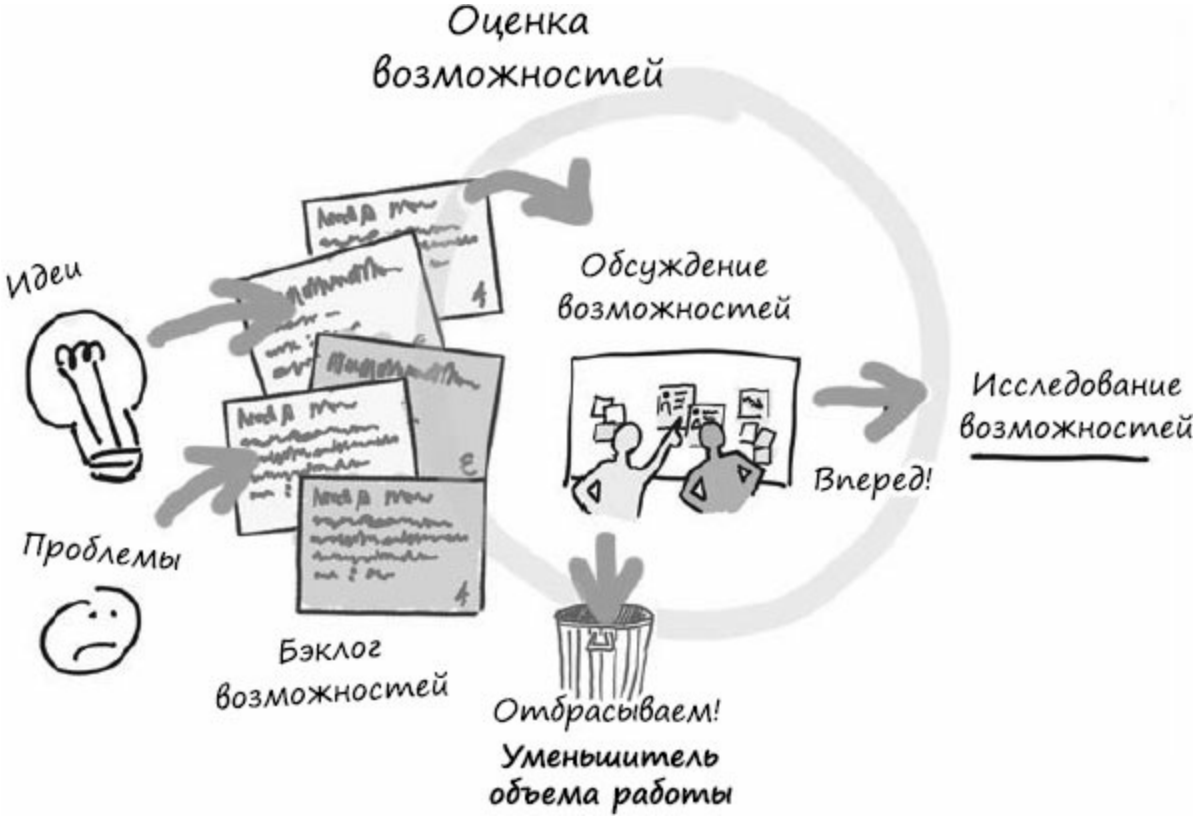
# После анализа: выбрасываем или продолжаем думать

«Да» не означает «мы собираемся реализовать это» – это означает, что мы переводим возможность на новый уровень и продолжаем дискуссии, углубляя анализ. Исследование потребует еще больше обсуждений – скорее всего, с другими людьми за пределами зала для совещаний. Если это новая функция или совершенно новый продукт, потребуется глубокий анализ поведения заказчиков и пользователей, способов решения их проблем на сегодняшний день. В идеале – пообщайтесь с ними напрямую. Потребуется также исследование и прототипирование разных решений. Тщательных обсуждений потребует вопрос о том, какие знания необходимы вам и вашей команде для принятия решения, что именно нужно разработать для эффективной помощи заказчикам, пользователям и своей организации. Но, даже выполнив всю эту работу, вы все еще можете отбросить данную идею.

«Нет» – это отличный результат обсуждения возможности. Помните: того, что нужно разработать, всегда больше, чем времени или ресурсов. Если в результате обсуждений выяснится, что возможность не так уж благоприятна, смело отправляйте ее в утиль. В этом случае полезно, чтобы в дискуссии принимали участие те, кто предложил эту идею, чтобы они сами могли прийти к такому же заключению.

У вашей группы может быть недостаточно информации для принятия решения «да» или «нет». Если это так, составьте список того, что нужно выяснить, и соберите нужную информацию.

Если вы все еще не можете решить, «да» или «нет», всегда можно перенести эту возможность обратно в бэклог. Это называется «отложенный выбор», и я прибегаю к нему довольно часто.

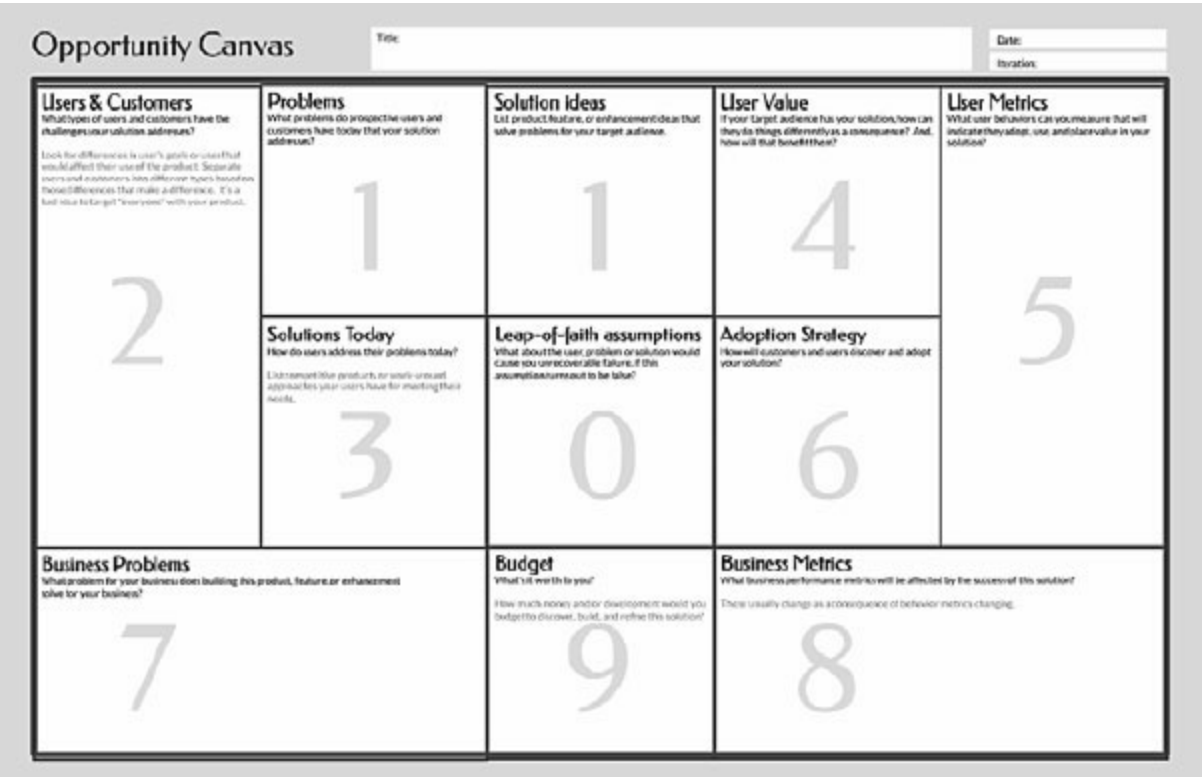


## *Сетка возможностей*

Рассматривая возможности продукта, я в качестве отправной точки использую шаблон экспертизы возможностей Марти Когана. В последнее время мне все больше нравятся подходы, основанные на использовании сетки. Такой подход к рассмотрению бизнес-моделей, описанный в книге «Выработка бизнес-модели»[26], – эффективный способ групповой работы для определения размера запускаемого бизнеса. Но мне самому, а также большинству людей, с которыми я работаю, редко приходится разворачивать новый бизнес или запускать принципиально новый продукт. Чаще всего мы должны добавить очередную важную функциональность к продукту, который уже существует, впрочем, это не должно помешать вам применить подход сетки к определению размера возможностей продукта.

Сетка организует информацию в пространстве. Большой объем информации удобно расположен в одном месте, в результате чего группа может охватить его взглядом и работать с ним. Делать то же самое со слайдами или печатным документом гораздо сложнее. Кроме того, такая организация позволяет вам увидеть зависимости: факты, зависящие друг от друга, расположены рядом.

Вот как может выглядеть сетка.



Совместная работа в попытке собрать информацию в виде сетки выглядит следующим образом.


Использование сетки дает следующие преимущества.

• Вы можете охватить самые важные проблемы одним взглядом.

• Вы можете видеть взаимосвязи между проблемами.

• Выработка одинакового понимания, участие в принятии решений и сплоченность, возникающие в результате совместной работы, позволяют создать сетку, куда свой вклад внес каждый.

Работая над сеткой, команда исследования продукта должна заполнить ячейки, основываясь на том, что вам известно сегодня. Привлекайте представителей бизнеса, экспертов в предметной области и любых других людей, которые, по вашему мнению, могут дополнить эту беседу полезной информацией.

Используйте стикеры, чтобы легче было редактировать сетку по мере развития дискуссии. Последовательно совершенствуйте сетку, узнавая что-то новое.

Вы можете заполнять сетку, начав с первой ячейки и последовательно двигаясь к девятой. Но если у вас пока нет ответов для какой-либо ячейки, запишите *только* то, что знаете или предполагаете, а затем двигайтесь дальше.

**Заполните сетку одним махом, а в следующий раз перечитайте**

Сетки в ячейке пронумерованы в логическом порядке, который можно использовать для обсуждения возможности. Но если вы покажете эту сетку кому-то другому, то можете прочитать ее слева направо или сверху вниз. Как можно видеть, порядок слева направо соответствует продвижению от «сейчас» к «будущему», как в модели объема работы и результата, представленной ранее. Кроме того, порядок сверху вниз соответствует продвижению от нужд пользователя к нуждам бизнеса.

Это не просто форма, которую нужно заполнить. Это набор тем, которые необходимо обсудить и в которых хорошо разобраться. Помните: дизайн через сотрудничество – не то что дизайн в комиссии. Привлечение множества разных людей позволяет всем узнавать новое быстрее, но все же решение «да» или «нет» в отношении этой возможности остается за владельцем продукта. Лучшие владельцы продукта привлекают к принятию решения свою команду, и, как правило, оказывается, что между ними и их командами нет противоречий.

Вот как протекает обсуждение в сетке возможностей.

**1. Проблемы и решения**

В идеале мы должны начать с ясно сформулированной проблемы, которую необходимо решить, но в реальном мире такое встречается редко. Чаще всего у нас есть просто функциональность или идея улучшения, а затем мы должны отступить назад, чтобы понять, в чем проблема. Поэтому начинайте с того, что у вас есть.

• *Идеи по поводу решения.* Перечислите продукты, функции или мысли по поводу улучшений, которые могут решить проблемы целевой аудитории.

• *Проблемы*. Какие проблемы есть сейчас у целевых пользователей и заказчиков? Если вы создаете развлекательный продукт, например игру или приложение для размещения в социальной сети каких-то забавных вещей, речь может идти не о реальной *проблеме*, а просто о желании развлечься.

**2. Пользователи и заказчики**

Какие типы пользователей и заказчиков испытывают трудности, которые устранит ваше решение? Рассмотрите различия целей или нужд разных людей, влияющие на способ, которым они используют продукт. Разделите пользователей и

заказчиков на разные типы, основанные на этих различиях. Адресовать продукт всем и каждому – очень плохая мысль.

**3. Существующие решения**

Как пользователи решают свои проблемы сегодня?

Перечислите продукты-конкуренты или способы, с помощью которых ваши пользователи решают свои задачи вручную.

**4. Выгоды пользователей**

Какие положительные изменения ощутит целевая аудитория, получив ваше решение? Какие выгоды это принесет?

**5. Пользовательские параметры**

Какие параметры поведения пользователя вы можете измерить, чтобы понять, как проходят адаптация, использование вашего решения и получение выгоды от этого?

**6. Стратегия адаптации**

Каков будет процесс перехода пользователей и заказчиков к использованию вашего решения?

**7. Проблемы бизнеса**

Какую проблему вашего бизнеса решает создание этих продуктов, функциональности или улучшения?

**8. Показатели бизнеса**

Какие показатели эффективности бизнеса будут затронуты в случае успеха вашего решения?

**9. Бюджет**

Во что вам это обойдется?

# Возможность не должна быть эвфемизмом

Я знаю, что в вашей компании, скорее всего, не используется понятие «возможности». Если она хоть немного похожа на те, с которыми мне случалось работать в прошлом, у вас есть стратегический план, куда включено то, что вы планируете разработать. Может быть, вы даже воспринимаете его как требования. Скорее всего, решение «нет» принимаете не вы. Правда заключается в том, что на самом деле мы не можем превратить все эти мудрые идеи в программные продукты независимо от должности человека, который их выдвинул.

Используйте первое большое обсуждение истории, чтобы примерно определить, какую работу придется выполнить вам и вашей команде. Несмотря на то что ответом на вопрос «Да или нет?» может быть «нет», перед уходом из комнаты совещаний удостоверьтесь, что у вас выработано одинаковое мнение о проблемах, которые вы решаете, о том, для кого вы их решаете, а также о выгоде для вашей организации от создания этого программного обеспечения.

Если не вы и не кто-то в вашей команде принимает решение, работать ли дальше с этой возможностью, постарайтесь выяснить это в беседе с теми, кто будет решать. Если эти люди недоступны для вас, все равно проведите обсуждение и сделайте предположения «Кто?», «Что?» и «Почему?». Затем передайте эти предположения лицам, ответственным за решения. Обещаю, они непременно поправят вас, если ваши предположения неверны. Рассматривая же корректировки, вы сможете начать правильное обсуждение.

# Карты историй и возможности

Как бы я ни любил карты историй, я редко использую их при управлении возможностями. Возможности – довольно крупномасштабные вещи, поэтому обсуждение этих «больших камней» чаще всего сводится к деталям, которые нужны нам, чтобы принять решение о более детальном исследовании.

Но вот где карты историй могут сработать – это действительно эффективная возможность сделать шаг назад и окинуть взглядом цельную картину продукта, как он есть сейчас. Используйте карту существующего продукта, чтобы найти возможности или рассмотреть те, которые вам предложены, в контексте всего, что у вас есть сейчас.

Попробуйте построить простую, высокоуровневую карту существующего продукта. Это карта «сейчас», похожая на представленую в главе 5, где отображается ваше утро (вы же построили ее, правда?). Эти типы карт много раз встречались нам в различных формах. Часто их называют *картами маршрутов*. Чтобы составить такую карту для текущего состояния вашего продукта и взаимодействия пользователя с ним, просто составьте карту основных действий пользователя в вашей программе, которые ведут к какой-либо цели. Используйте эту карту, чтобы рассмотреть свои возможности в контексте, – для этого нужно добавить на карту каждую возможность. Используйте стикеры или карточки разных цветов, чтобы выделить их. Скорее всего, вы обнаружите горячие точки возможностей – места на маршрутах взаимодействия пользователей с системой, где не помешают улучшения и где, скорее всего, пользователи испытывают затруднения.

Рассмотрите пользователей, вовлеченных в каждое такое действие, и частоту этих действий. Возможности, которые затрагивают часто встречающиеся действия и критически важных пользователей, следует взять в работу в первую очередь, оставив прочие на потом.

Вы можете также использовать эту карту, чтобы добавлять на нее то, на что ваши пользователи сейчас жалуются. Чтобы добиться баланса, рассмотрите части продукта, которые они любят, и добавьте на карту то, что им нравится. Если же вы обнаружили место, где много жалоб пользователей, но пока нет ни одной возможности, очевидно, стоит о нем подумать.


## *Карта маршрутов и генерация концептов*

*Бен Кротерс, Atlassian*

Поскольку мы предлагаем больше десяти различных продуктов, нам необходимо быть уверенными в том, что мы проектируем, разрабатываем и улучшаем эти продукты так, чтобы наши клиенты могли использовать их вместе, а не только по отдельности. Поэтому в рамках проекта по исследованию и улучшению совместной работы этих проектов мы сформировали мультидисциплинарную команду, которая должна была составить карту полного, от начала до конца, маршрута пользователя в системе: знакомство, оценка, покупка и использование, а кроме того, получение помощи и добавление других продуктов и сервисов.

Объем работы оказался огромным. Чтобы разделить его на части, мы сначала построили очень высокоуровневую карту маршрутов, а затем разбили ее на подгруппы, чтобы впоследствии развивать каждую секцию получившегося каркаса. Сначала мы заполнили стену ключевыми моментами, действиями и вопросами, с которыми работали клиенты, присвоили им цветовые коды, а затем повторно прошлись по этой карте, добавляя точки раздражения, возможности и предположения.


Мы сделали множество открытий, когда появилась возможность проследить всю историю от начала до конца, а не просто рассматривать каждую функциональность в отдельности. Так, мы быстро поняли, что некоторые части пользовательского взаимодействия, например настройка продукта или получение помощи, не были выделены в особую ветку маршрута и нуждаются в большей детализации и проработке.

Много других команд и партнеров были привлечены к этой работе для добавления деталей, относящихся к определенным точкам высокоуровневого маршрута. В результате мы зафиксировали и подтвердили максимально возможное на тот момент количество сведений.

Затем мы с головой ушли в поиск как можно большего количества идей улучшения и реорганизации различных этапов уже детализированных маршрутов. Все эти идеи были как следует выверены и записаны на карточки, которые в свободном порядке приклеили на стену вдоль маршрута.

Каждый член команды объяснил каждый свой концепт, после чего все мы проголосовали за эффективность всех концептов методом точек, учитывая их реализуемость, жизнеспособность и полезность для пользователей.


После этого мы уже были способны сформировать полноценное видение идеального взаимодействия клиента и всех продуктов, основываясь на картах маршрутов и подтвержденных концептах. За время работы над картой родился 20-страничный комикс, который трансформировал маршрут, а также всех персонажей, истории и концепты, которые были как-то с ним связаны, в один набор историй, в результате чего значительно упростилась коммуникация во всей организации. В комиксе описываются возможные улучшения в продукте, и он продолжает развиваться.

Для нас был очень показателен тот факт, что многие из вовлеченных в разработку и корректировку этих улучшений затем включились в разработку концептов, с которых мы начали работу. И хотя это упражнение было проделано восемь месяцев назад, одинаковое понимание и концептуализация значительно повысили эффективность работы.

## *Сложные обязательства и прогон карты*

*Эрин Бейерволтес и Аарон Уайт*

**1. Одна карта, чтобы управлять всем**

После долгого общения с клиентами, интервью и исследований владелец продукта получил маршрут, который, как он был уверен, отражал наилучший итог работы для заказчика. Все происходило как обычно. Настало время поделиться информацией с командой и взяться за работу.

**2. Все под сомнением…**

Но по прошествии нескольких обсуждений мы уже не были так уверены, что ничего не пропустили. Возможно, мы сделали какие-то простые предположения, эффект от которых окажется больше ожидаемого? Нужен был быстрый способ проверить наши предположения, обнаружить пробелы, а также выработать одинаковое понимание у всех, кто примет участие в проекте.

**3. Подготовка сцены**

Мы собрали команду, назначили, кто каким персонажем будет (их несколько), и описали желаемые результаты, которые должны были естественным образом обусловливать действия персонажей во время использования части нашего продукта. Таким образом мы должны были подтвердить или опровергнуть свои предположения. Мы *не* писали пошаговые инструкции. Задача была поставлена такая: «Просто, как сделал бы этот персонаж, вы должны достичь такой-то цели». Прогон!


**4. Прогон**

Пока актеры играли свои роли и пытались достичь поставленных целей, несколько наблюдателей смотрели на них, словно из зрительного зала. Владелец продукта следил, какие способы находили актеры, ответив на пару вопросов, но не указав никому конкретного пути. Дизайнер пользовательского взаимодействия

наблюдал за их поведением, комментариями и реакциями.

**5. Перестроение карты**

Мы не показывали участникам изначальную карту, вместо этого попросили актеров под нашим руководством построить новую, где бы они описали, как достигли своих целей, по ходу дела поделившись друг с другом найденными способами.

**6. Удовольствие и раздражение**

После перестроения карты мы попросили каждого актера рассказать, из-за чего он или она испытывали раздражение (в частности, что вызывало сложности, злило или ставило в тупик), а из-за чего – удовольствие (что было легко и приятно сделать или было интуитивно понятно). Во время беседы все почувствовали усиление сопереживания и понимания.

**7. Выводы**

Под конец настала очередь зрителей задавать вопросы об интересных моментах поведения, которые они наблюдали. Мы были поражены тем, как много актеры проделали, сами того не зная, но это навело нас на мысль, что кто-то может попытаться сделать то же самое намеренно, имея какую-то цель.

**8. Профит!!!**

Всего через 2,5 часа у нас сложилось одинаковое понимание того, как воспринимается новое решение, что могло произойти только в обсуждении. Актеры действовали, реально сочувствуя клиентам, роль которых исполняли, а исследовательская команда сделала массу открытий о том, как что работает и где может потребоваться больше экспериментов.

# Не бойтесь риска

Если вы просто согласитесь со всем, что у вас есть, ничего хорошего не выйдет. Активно отбрасывайте возможности, не обещающие достижения тех результатов, на которые вы надеетесь. Сотрудничайте для этого с представителями бизнеса, возможно, они помогут вам принять эти решения.

Если же вы приняли решение «да», настало время засучить рукава и взяться за работу. Именно об этом следующая глава.

# Глава 14. Вырабатывайте единое понимание с помощью исследований

Видя простые модели описания Agile Development, я часто замечаю, что с левой стороны, где они начинаются, расположен большой список – *бэклог продукта*. Я считал бы это смешным, если бы многие люди не думали, что это очень просто. Получение хорошего, выверенного бэклога из накопившихся возможностей неизбежно займет очень много времени и сил – он не появится сам по себе. И конечно, этот бэклог не может просто состоять из того, что люди хотели бы создать. Это отдельный исследовательский процесс, который начинается с концентрации на углубленном изучении «Для кого?», «Что?» и «Почему?».

# Суть исследований не в создании программных продуктов

Исследовательская работа обычно не направлена на создание пригодного к предъявлению программного продукта. Ее основная цель – выработка углубленного понимания того, что мы можем создать. Речь идет о постановке и поиске ответов на следующие вопросы.

• Какие *проблемы* мы решаем в действительности?

• Какое решение было бы наиболее *полезным* для нашей организации и клиентов, покупающих или использующих наш продукт?

• Как должно выглядеть решение, наиболее *удобное* и *эффективное* в использовании?

• Что мы имеем возможность *реализовать*, учитывая имеющиеся время и средства?

Таким образом, постановка и поиск ответов на все эти вопросы, заданные по поводу какой-либо возможности, запускает первый раунд разбиения камней. Все детали относительно продукта или функциональности, которые вы обсуждаете, становятся названиями более мелких историй. А каждая из этих более мелких историй может привести к еще более углубленным дискуссиям и разбиению на еще меньшие истории.


Но исследовательские дискуссии не должны привести к одному только увеличению

количества историй. Помните, что в результате обсуждения историй должны появиться другие простые модели, отражающие то, что нам удалось постичь. Они необходимы для выработки одинакового понимания.

*Если единственное, что у вас получилось в результате обработки большой многообещающей возможности, – это набор более мелких историй, вероятно, вы что-то сделали неправильно.*

# Четыре необходимых шага исследования

Если у меня есть большая идея или даже не очень большая, но требующая некоторой доработки, я последовательно провожу определенные обсуждения. От большой идеи мы переходим к деталям, которые нужны, чтобы лучше понять, достойно ли это решение реализации.

1. *Сформулируйте* идею с точки зрения бизнеса.
2. Постарайтесь *понять,* каким образом вы можете помочь *заказчикам и пользователям*.
3. *Сформируйте видение* вашего решения.
4. *Минимизируйте его, а также составьте план*, как выявить и затем создать минимально жизнеспособное решение.

## 1. Сформулируйте идею

Если вы и в самом деле используете бэклог возможностей и проводите настоящие обсуждения возможностей, чтобы принять решение о начале исследований, то вы на правильном пути. Используйте дискуссии для обкатки идеи, чтобы запустить направленное исследование. Привлекайте людей, с которыми будете вместе работать, чтобы они лучше поняли эту возможность.

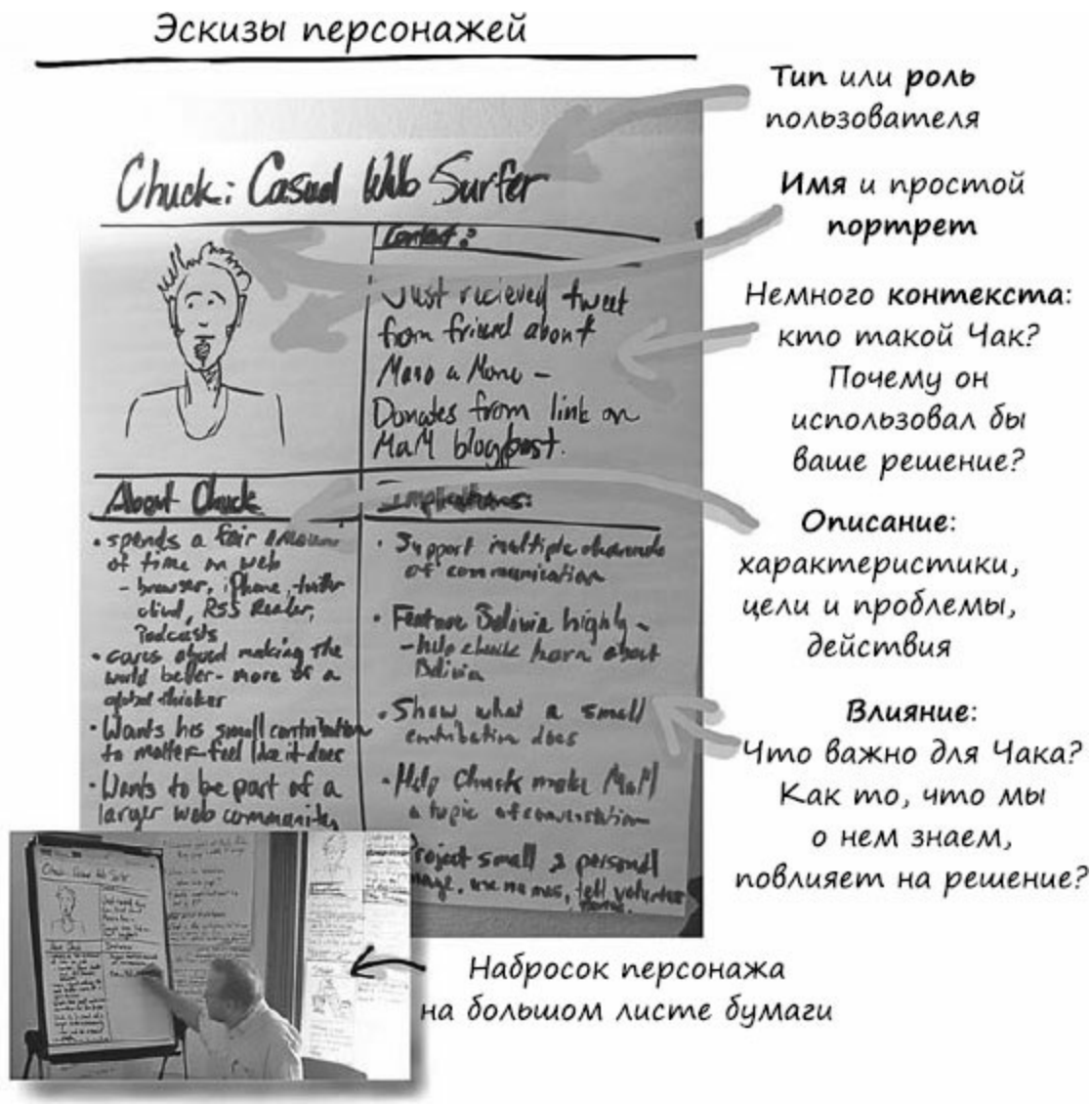
Применяйте формирующие дискуссии, чтобы очертить границы намеченной работы. Если вы хорошо понимаете, зачем и для кого что-то делаете, то вам и вашей команде легче будет прекратить обсуждать решения, которые никак не повлияют на рассматриваемую проблему и никак не помогут выбранным пользователям.

## 2. Поймите заказчиков и пользователей

Применяйте обсуждения заказчиков и пользователей, чтобы лучше понять людей, для которых предназначены ваши продукт или функциональности, а также пользу, которую вы собираетесь им принести. Привлекайте к дискуссии тех, кто хорошо понимает проблемы пользователей, а также всех, кому нужно их понять.

### Рисуйте эскизы упрощенных персонажей

Мне нравится быстро набрасывать эскизы упрощенных *персонажей* вместе с небольшой исследовательской командой, чтобы выработать одинаковое понимание пользователей. Персонаж – это пример целевых пользователей, в описании которого использованы факты, а порой и предположения о ваших пользователях. Создание персонажей помогает нам взглянуть на программный продукт глазами пользователей, поэтому создание персонажей – очень полезная техника.




Я создал этот простой персонаж вместе с группой из Mano a Mano, некоммерческой организации, которая помогает жителям Боливии всем, чем может: от постройки дорог до обеспечения образования и здравоохранения. Наша дискуссия в тот день касалась скромных благотворителей, вносящих небольшие пожертвования через Интернет, – пусть у них нет больших денег, но они могут внести свою скромную лепту, если уверены, что она пойдет на доброе дело. Мы ожидали, что люди наподобие Чака могут найти информацию о Mano a Mano в Интернете или наткнуться на упоминание о ней в Facebook или Twitter.

Мы создали этот персонаж все вместе на большом листе бумаги. Это было веселое занятие с участием многих людей, каждый из которых вносил свой вклад, называя известные ему факты. И времени оно заняло немного.

Если вы опытный дизайнер UX, которому прежде приходилось создавать персонажей, то, наверное, уже закипели от возмущения. Я поясню для остальных: обычно хорошие персонажи создаются по результатам тщательного сбора и анализа данных, полученных путем научного исследования. Специалисту UX члены команды, наперебой выкрикивающие характеристики пользователя и пишущие их на доске, кажутся совершенно нелепыми. В этом нет ничего научного. Поэтому и в самом деле не стоит орать что попало – просто обсудите то, что вы знаете и сами наблюдали. Изложите истории. Вовлекайте в дискуссию людей, которые непосредственно взаимодействовали с пользователями. Если вы проделали большие исследования, захватите их результаты на обсуждение. Отметьте детали, наиболее важные с точки зрения возможности, которую вы обсуждаете, и занесите их в карточку персонажа.

Отфильтруйте все лишнее. Когда закончите, честно обсудите, какая часть сделанного основана на догадках и предположениях.

«У нас уже есть персонажи. Вон там, на стене, висят красиво оформленные документы». Сколько раз я слышал эти слова! Но будьте честны сами с собой: большинство из вас этого не читали, правда ведь? А половина тех, кто прочитал, сделали это просто ради смеха. Может быть, я циник, но я наблюдал такую картину множество раз.

Создавайте персонажей вместе, как команда. Сделайте это, чтобы у всей команды было общее мнение по поводу людей, которые будут использовать ваш продукт. Сделайте это, чтобы учесть самые релевантные аспекты персонажа.

> *Создавайте упрощенных персонажей вместе, чтобы выработать одинаковое понимание и сопереживание внутри команды.*

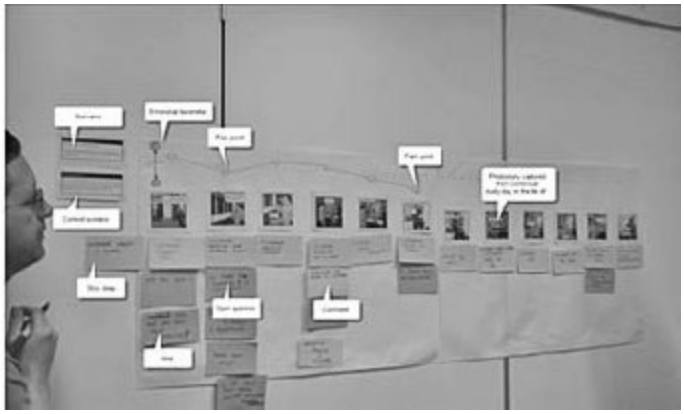
Я создаю упрощенных персонажей для каждого типа пользователя, который может использовать обсуждаемую нами функциональность. Если времени мало, то просто перечисляю разные типы или роли пользователей, применяющих продукт, и кратко упоминаю некоторые относящиеся к ним детали. Помните Гэри из главы 1? Одна из групп карточек рядом с ним была именно такой – с названием пользователей и парой предложений о каждом из них.

### Создавайте профили организаций, или «оргсонажи»

Если вы работаете над продуктом, который могут купить организации, скажем над программой для бухгалтерского учета, выделите время, чтобы перечислить разные типы организаций и записать несколько деталей о каждой из них. Это ваши заказчики – люди с деньгами, которые собираются извлечь какую-то пользу из вашего продукта. Информацию о типе организации с несколькими дополнительными деталями принято называть *профилем организации*. Моя подруга Лейн Хелли первой подала мне идею создавать примерные профили организаций аналогично тому, как делаются персонажи. В шутку она называет эти профили *оргсонажами*.

### Составьте карту работы пользователя на сегодняшний день

Вы можете продвинуться на один шаг дальше и составить карту работы ваших пользователей сегодня, еще до реализации продукта. Если вы сделали упражнение из главы 5, то легко построите такую карту аналогично той, где отображается ваше утро. Создание карты, где представлены существующие способы решения проблем пользователей, поможет вашей исследовательской команде по-настоящему разобраться в проблеме, которую они решают.



На этих фотографиях Дункан Браун из Caplin Group показывает то, что у них называется *описательной картой маршрута*. Видна «теперешняя» часть модели «до и после», с которой я начал эту книгу. На ней никак не отражается наше великолепное решение, а изображено лишь то, каким образом люди достигают своих целей сейчас – неудобства, неудачи и т. п.

Основная часть карты содержит факты, наблюдения, моменты раздражения и удовольствия. Составляя карту того, что вы понимаете сейчас, вы непременно найдете горячие точки – области процесса, где сосредоточено много проблем. Возможно, вы также найдете моменты удачи, когда, выполнив последовательность из нескольких шагов, пользователь доволен результатом, который принесли его усилия. Вы можете создать великолепный продукт, минимизируя неприятные вещи и увеличивая приятные. Используйте карту как трамплин для решений путем мозгового штурма или для того, чтобы убедиться, что уже намеченное решение поможет справиться с реальными проблемами.

## 3. Визуализируйте свое решение

К настоящему моменту вы должны были ясно сформулировать, почему что-то создаете, с точки зрения бизнеса; вы исследовали пользователей и заказчиков, так что знаете, как у них обстоят дела сейчас. Настало время вообразить будущее – сформировать видение своего решения, а также взаимодействия с ним заказчиков и пользователей.

### Составьте карту решения

Вот где карты историй проявят себя – по крайней мере, я так думаю. Я использую карты историй, чтобы четко представить себе жизнь пользователей после того, как им будет предъявлено мое решение. И Гэри, и команда Globo.com из первых двух глав строили именно такие карты. И как мы говорили во вступительных главах, шаги, которые делают люди в вашей истории, формируют последовательное повествование слева направо. Эти шаги называются пользовательскими задачами и представляют собой короткие глагольные фразы, которые, если читать их слева направо, образуют историю. Более точно сформулированные задачи и различные детали перечислены вертикально под каждым шагом. Если история длинная, выделите группы действий и постройте трехуровневую карту.

### Слова и картинки

Случалось ли вам, описав программисту идею продукта, с приятным удивлением слышать в ответ: «О, это просто. На разработку не потребуется много времени»? Правда, позднее, когда вы приступали к разработке, оказывалось, что программист понял вас неправильно и думает о чем-то значительно более простом, чем представлялось вам. Например, вы говорили о сайте, с помощью которого можно продавать вашу продукцию в Интернете. Вы воображали что-то вроде Amazon Marketplace или eBay, а программист подумал о чем-то наподобие Craigslist, поэтому и дал такую оптимистичную оценку. В общем, за последние десять лет я твердо усвоил, что одних слов недостаточно.

*Визуализируйте пользовательский интерфейс, чтобы выработалось одинаковое понимание вашего решения.*

Если в вашей команде есть дизайнеры пользовательского взаимодействия, настало время им нарисовать эскизы. Наброски отдельных экранов можно помещать прямо наверху карты в порядке их появления. В конце концов у вас получится что-то вроде раскадровки.

### *Полная визуализация взаимодействия*

*Джош Сайден (иллюстрации Демиана Репуччи)*

В один прекрасный день мне позвонил Роберт, менеджер по дизайну, недавно получивший работу в большом, хорошо финансируемом образовательном стартапе. Компания недавно начала крупный проект и активно набирала людей, которые должны были в сжатые сроки разработать обширную систему. Одна проблема: никто не знал, как подступиться к дизайну такого огромного и сложного проекта. Не могу ли я помочь?

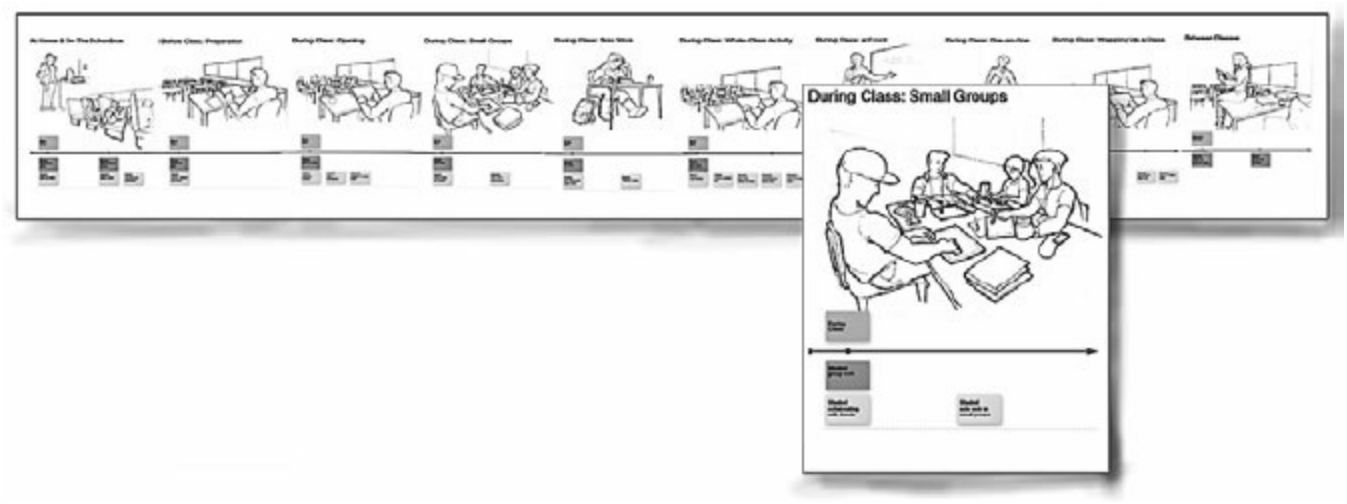
Придя в офис через несколько дней, я встретился с Робертом. Он, одновременно гордый и предельно замотанный, все мне показал. Компания наняла большую консалтинговую фирму для помощи в разработке требований к проекту, и фирма проделала впечатляющий объем работы. Все стены в просторном офисе-лофте были завешаны листами тонкой коричневой бумаги, а каждый лист, в свою очередь, покрыт карточками и стикерами. Это были требования в форме пользовательских историй. Тысячи историй. Пока Роберт вел меня вдоль стены с историями, я обратил внимание на то, что истории были организованы согласно функциональным модулям: стена текстового редактора, стена английского языка. Но мне не удавалось представить себе работу системы целиком.

Роберт работал над созданием команды дизайна, а также пытался разделить задачу на составные части. Обсудив возможные потребности его команды, мы пришли к выводу, что можно использовать карту историй, с помощью которой будет удобно организовать тысячи пользовательских историй так, чтобы команды дизайна и разработки пользовались ею совместно.

Случилось так, что через несколько недель я присутствовал на семинаре, который проводила команда художников-раскадровщиков. Целью семинара было обучение заказчиков формулировать видение их бизнес-идей. Эти художники сидели рядом с заказчиками и, быстро работая, рисовали по их описанию истории

и набрасывали идеи в виде раскадровок – мини-комиксов, которые ясно и четко излагали каждую историю. Мне тут же захотелось скомбинировать этот подход с составлением карт историй, поэтому я пригласил присоединиться ко мне художника Демиана Репуччи, чья работа впечатлила меня наиболее сильно.

Через пару недель Демиан и я встретились с Робертом и его командой, а также менеджерами продуктов из разных частей системы. Мы сфокусировались на высокоуровневых процессах – основных вариантах действий пользователя в системе. В течение встречи Демиан рисовал в блокноте, а я выкладывал на стенах конференц-зала пользовательские варианты с помощью карточек и стикеров. После окончания встречи Демиан должен был вернуться к себе в студию, чтобы проиллюстрировать ключевые моменты, а я использовал Omnigraffle, чтобы выполнить окончательные версии карт историй, черновики которых мы набросали на встрече.




Мы с Робертом решили: самое лучшее, что мы можем сделать для команды, – это организованная структура, поэтому выпустили серию постеров, которые можно было напечатать на листах 11×17 дюймов и повесить на стены, что сформировало бы каркас карты историй. После этого команды могли бы пользоваться ими независимо друг от друга, чтобы реорганизовать свои истории. В результате из вида, организованного по модулям, который никак не помогал итеративной разработке, мы получили вид, основанный на логике пользования, который легко было трансформировать в несколько многомодульных релизов.

Один из способов визуализации пользовательского взаимодействия, вовлекающий в работу всю команду, – подход проектной студии. Проектная студия – это быстрый и простой способ, позволяющий организовать сотрудничество большой группы людей для работы над осознанным формированием идеи, или, проще говоря, выдвижения большого количества всевозможных идей. Вы быстро поймете, что самые лучшие идеи не может выдвинуть какой-то отдельный человек – лучшие идеи возникают из комбинации предложений нескольких авторов во время обсуждения. Проектная студия (и осознанное выдвижение идей) – противоположность общепринятому подходу, когда кто-то приносит идею, на первый взгляд

кажущуюся подходящей, и все обсуждают ее. Впервые услышав о методе проектной студии от его авторов Джеффа Уайта и Джима Унгера, я удивился, почему раньше не пользовался таким прекрасным подходом. С тех пор я много раз запускал сессию проектной студии с участием команд разработки, ключевых партнеров и даже пользователей или заказчиков.

Какой бы подход вы ни использовали, комбинируйте идеи, совершенствуйте их и приходите к единому пониманию программного продукта, который хотите создать.

Может случиться, что в попытке визуализировать свое решение вы обнаружите, что некоторые моменты не были внесены вами в карту. Придется что-то добавлять, менять или переставлять карточки, чтобы отразить на карте вашу новую идею. Не волнуйтесь, на самом деле это положительный момент.

### *Рецепт проектной студии*

Проектная студия – эффективный и быстрый способ работы в группе для генерации идей. Этой техникой можно пользоваться по-разному, вот мой простой рецепт.

1. *Пригласите группу людей*, чьи мнение и идеи вам близки, а одобрение и понимание нужны для создания продукта. Оптимальный размер группы – от восьми до двенадцати человек.

2. *Опишите задачу, которую решаете*. Рассмотрите уже проделанную работу, чтобы выявить возможности. Просмотрите еще раз персонажей и карты, отражающие то, как пользователи решают проблемы сейчас. Если вы уже начали составлять какие-то карты для решений, просмотрите и их, но будьте осторожны, не говорите слишком много, иначе все неизбежно будут отталкиваться от ваших мыслей и вы не услышите самые лучшие идеи других людей.

3. *Можно продемонстрировать примеры для вдохновения*. Посмотрите вместе и обсудите удачные похожие продукты. Рассмотрите также продукты, не совсем схожие с вашим, но воплощающие удачные идеи, которые могут натолкнуть на хорошую мысль.

4. *Рисуют все!* Раздайте всем бумагу, ручки, карандаши, при желании – какие-то шаблоны экранов и установите таймер. В моей практике встречались отрезки времени как 5, так и 60 минут. Лучше всего, по-моему, примерно 15 минут.

5. *Делитесь идеями, если действуете в маленьких группах*. Этот прием работает в группах до четырех человек, поэтому, если всего вас, например, 12, разбейтесь на три меньшие группы. Поделитесь друг с другом своими идеями и дайте им оценку. Следите, чтобы оценки высказывались с точки зрения эффективности решения изначальной проблемы, а не просто из-за того что нравится или не нравится. Следите, чтобы каждый развивал не только свои, но и чужие идеи. Продолжайте личную работу в маленьких группах в течение фиксированного времени, скажем 30 минут.

6. *Попросите каждую группу объединить свои лучшие идеи* в одно решение и проиллюстрировать его эскизами. Это самая сложная часть. На нее стоит отвести 15–30 минут.

7. *Попросите каждую группу представить свои лучшие идеи, сведенные*

*воедино*, всей группе. Обсудите их.

8. *Поблагодарите всех и объедините все идеи и наброски*. Вы, UX-специалист и исследовательская команда должны рассмотреть и задействовать их, чтобы создать финальные, наилучшие эскизы UI. Помните выражение моей подруги Лизы Райшелт: «Дизайн через сотрудничество – не то, что дизайн в комиссии»? У вас будет много отличных, но не согласующихся друг с другом идей, и кому-то придется принять нелегкое решение.

Фотографии любезно предоставлены Edmunds.com

Фотография любезно предоставлена Лейн Хелли (LaneHalley)

## Убедитесь в законченности

Что мы умеем делать хорошо, так это додумывать детали. Например, если мы видим два кадра какого-нибудь комикса, то легко догадаемся, что произошло между ними. Этот прием широко используется в комиксах, романах и кинофильмах. Но, раздумывая о том, что делает программный продукт, мы часто склонны воображать себе интересные функциональности, пренебрегая тем необходимым, что их связывает. Продолжая аналогию с фильмом, можно представить, что на экране сменяли бы друг друга погони на автомобилях и перестрелки, но ни слова не было бы сказано о том, почему все это происходит и какая история их связывает.

Изложение во всех подробностях пользовательских историй с помощью карты поможет вам проговорить все критически важные детали, которые их связывают. В большинстве случаев люди понимают, что крутая функциональность, которую они рассматривают, потребует изменения некоторых пользовательских настроек в самом начале истории, а также отчетов и уведомлений в конце. Могут быть затронуты не только функциональности, но и люди. Например, администраторы должны обработать новые настройки безопасности для системы, а менеджеры – контролировать, как функциональность используется в их командах.

## Проверьте технические особенности

Возвращаясь к аналогии с киносъемками: если вы собираетесь продолжать работу над своим фильмом, стоит подумать о выборе места съемок, спецэффектах и других деталях. В какой-то момент вам придется с головой погрузиться в сценарий и продумать все технические детали создания фильма.

Карта историй вашего программного обеспечения очень полезна в дискуссиях того же

типа, что вы проводили бы, будь вы режиссером кинофильма. Обсудите карту своих решений с инженерами и архитекторами, прежде чем двигаться дальше. Видение картины в целом поможет им заранее предусмотреть большинство технических ограничений, которые могли бы затруднить реализацию по-настоящему удачных решений. Вас могут заранее предупредить о том, что хоть ваше решение и выглядит прекрасно, но реализовать его при заданном бюджете времени и имеющейся архитектуре не представляется возможным. Они могут также предложить альтернативные пути выполнения задач, которые обеспечат вашим пользователям те же возможности взаимодействия, но будут более экономичны с точки зрения реализации.

Эти инженеры большой страховой компании потратили много времени на обсуждение находящейся на стене карты. По ходу обсуждения они обнаружили в огромной карте своего продукта небольшую нестыковку: придется изменить обработчик бизнес-правил их продукта. Возможность увидеть картину в целом помогла им визуализировать свою работу несмотря на сложность продукта. Обладая этим знанием, ребята смогут обсудить, что им необходимо запланировать на самых ранних этапах и как минимизировать риски.

## Поиграйте в «Что, если»

Вы представляете свое решение с точки зрения пользователя, и у вас есть видение пользовательского взаимодействия. Выделите некоторое время для обсуждения того, что происходит внутри интерфейса. Обсудите строгие бизнес-правила, сложную валидацию данных, а также разные серверные сервисы и системы, с которыми придется

взаимодействовать. Добавляйте на карту истории, обозначающие эти части, или делайте пометки в уже имеющихся историях.

Это хороший момент, чтобы рассмотреть то, что вами уже сделано, с большим количеством других людей. Расскажите им обо всем, к чему удалось прийти вам и вашей группе. Обещаю: у вас будет большая группа людей, наперебой задающих вопросы, начинающиеся с «А что, если…». Я обожаю таких людей (только не в тот момент, когда они стоят прямо передо мной). И я благодарен им за то, что они помогают мне задуматься о множестве проблем, прежде чем я натолкнусь на них сам позднее, когда полученный урок обойдется гораздо дороже.

Аналогия с киностудией, как мне кажется, отлично работает. Если бы я собирался снимать кино, мне нужны были бы сценарий и раскадровка – эскизы ключевых сцен, которые помогли бы мне лучше представить будущий фильм. Если бы я собирался вложить деньги в съемку фильма, то хотел бы увидеть как минимум две эти вещи, чтобы иметь представление о картинке, сложившейся в головах сценаристов и режиссеров. Если бы в тот момент задумка мне понравилась, я продолжил бы исследования, чтобы узнать, сколько денег потребуется на съемку фильма и насколько реально воплотить в жизнь эти замыслы.

Я думаю, в кино эти эскизы используются для того, чтобы более тщательно обдумать будущий фильм. Как инвестор я хотел бы, чтобы можно было заранее подсчитать, сколько площадок и павильонов нужно для съемки и как они должны выглядеть. Необходимо также знать, какие потребуются декорации, реквизит и спецэффекты. Словом, мне нужны были бы сценарий, раскадровка и множество других деталей, на которых базируются построение плана и оценка затрат. Имея эти данные, можно запланировать бюджет и временной график съемок фильма.

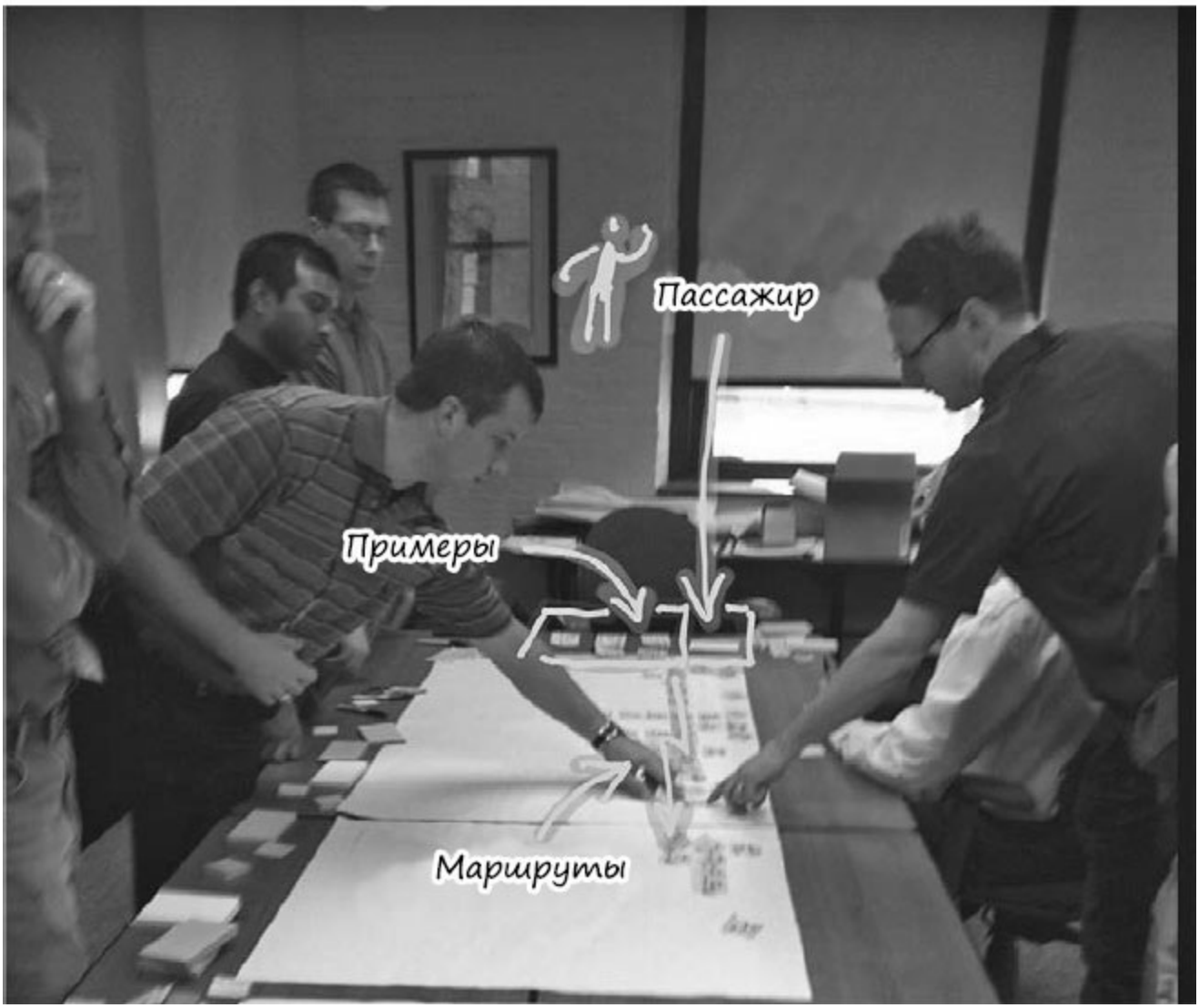
Вот какие сведения вы должны получить, прежде чем приступите к реализации своего решения.

### *Идеи, примеры и маршруты*

*Дэвид Хассман, DevJam*

Многие люди излишне усложняют процесс исследования, а он может быть очень простым и при этом вполне эффективным. Оставьте в стороне воображаемую жесткость требований и выделите время на исследование идей с помощью примеров и маршрутов пользователей, как говорится в приведенном далее руководстве.

Карта маршрутов и примеров




**Следуйте простому плану.**

1. Предложите идею для исследования.
2. Выберите людей, которым, по вашему мнению, будет выгодна реализация этой идеи.
3. Сделайте подборку примеров того, как эти люди используют данную идею.
4. С помощью этих примеров составьте карту и маршруты, которые, по вашему мнению, понадобятся.

Помните, что, как автор, вы несете ответственность за создание адекватного пользовательского взаимодействия, а не просто новых функциональностей.

**Идеи не обязаны быть гениальными.** Да, вы хотите создать великолепный продукт, но зачастую самые блестящие идеи не срабатывают как надо, в то время как полезность других, не вызывающих такого восхищения поначалу, раскрывается, когда их рассматривают в контексте применения для выполнения определенной задачи.

**Выбор «пассажиров» не отбор космонавтов.** Не стоит чересчур усложнять отбор целевой аудитории. Если вы не уверены, с чего начать, просто составьте список пользователей, которым, по вашим предположениям, будут полезны ваши

идеи. Дополните их образы, чтобы они долго оставались живыми и реальными в сознании членов команды разработки. Поработав с ними некоторое время, выберите какой-то конкретный тип пользователя и не слишком волнуйтесь о правильности решения. Будьте готовы извлекать уроки из исследований и не беспокойтесь о том, верен ваш выбор или нет, – скорее всего, нет.

**Составьте обширный перечень примеров.** Именно здесь многие люди теряют контроль над собой и лезут в дебри. Начните с самого простого, очевидного примера, чем конкретней он будет, тем лучше. Затем переходите к примерам посложнее и не бойтесь поднимать планку высоко. Создавая набор ограничений, вы не даете никаких гарантий. Но чем сложнее пример, тем конкретнее он должен быть. Если вы не уверены в необходимом уровне конкретики, трансформируйте рабочий пример в тест – это на шаг приблизит вас к автоматизированной валидации.

**Добавьте еще несколько примеров среднего уровня сложности, а затем начните работу с простыми примерами.** Расскажите историю «пассажира», используя самый простой пример в своем руководстве. Поделитесь ею с кем-нибудь и попросите этого человека помочь вам обработать остальные истории в маршруте. С чего начинают ваши пользователи? Какова их мотивация? Что конкретно они проделывают? Чем заканчивается история? Используйте разные примеры для обработки разных маршрутов ваших «пассажиров».

**Выберите маршруты, которые помогут вам в исследовании.** Еще одна сложность, которая вас ждет, – выбор стартовой точки. Пройдитесь по карте, выбирая маршруты, которые, как вы думаете, помогут вам узнать больше о пользователях и их нуждах. Как и прежде, беспокойство о правильности выбора может быть оправданным. Но лучший способ проверить это, как и лучшая инвестиция в исследование, – просто выбрать несколько маршрутов, создать их, а затем лично или средствами аналитики в реальном времени посмотреть, как люди их используют.

**Остерегайтесь ложных представлений о продукте.** Разница между тем, что, *как вы думаете*, нужно пользователям, и тем, что им нужно *на самом деле*, вызвана вашими ложными представлениями о продукте. Реализуя изложенный здесь процесс, вы можете быстрее провести исследования прямо в контексте, создавая и проверяя один маршрут за раз.

## Праздновать победу рано

На этом этапе процесса исследования вы уже должны были начать описывать части своего решения в виде отдельных историй. Каждый фрагмент является частью большой возможности. Если мы с вами хоть немного похожи, вам будет удобно организовать все эти части с помощью карты историй. Если вы умнее меня, может быть, изобрели лучший способ и, если это так, уже погружены в работу. Если же вы не склонны к порядку, у вас получится просто большая куча историй. Или, еще хуже, кто-то может написать длиннющий документ с требованиями, который немедленно запутает все, чего вам удалось достичь. Пожалуйста, не допустите этого.

Но с чем бы вы ни подошли к этому рубежу, именно сейчас вы, как и многие другие, испытываете искушение отпраздновать «окончание работы над требованиями». Не поддавайтесь ему. Перед вами последний и самый важный этап работы.

## 4. Минимизируйте объем работы и составьте план

У вас есть представление решения в словах и картинках, и сейчас ваша команда вполне может гордиться собой. Но самые большие проблемы с исследованиями возникают, когда вся команда работает сообща, чтобы разработать блестящее решение с множеством разных фишек и примочек, которые мы все так любим.

Я знаю, вы думаете: *«А что тут страшного-то?»*

Как правило, проблема заключается в том, что мы хотим сделать довольными всех и каждого и в то же время упускаем из виду четкие и конкретные ожидаемые результаты. В итоге полученное решение оказывается куда более громоздким, чем должно было быть.

Помните: наша цель в том, чтобы сделать объем работы, необходимый для реализации, как можно меньшим, а пользу, которую мы получаем (результаты и долгосрочные эффекты), – как можно большей. Ваша возможность разбита на множество маленьких индивидуальных историй. Вряд ли вы собираетесь реализовать их все до одного, ведь в таком случае о минимизации объема работы и речи не идет.

> *Если вы не отбросили больше историй, чем оставили для реализации, то, скорее всего, исследовательская работа проделана неверно.*

### Ресурсов всегда будет не хватать

Я очень сожалею, но это все еще так. Если вы имеете какой-то опыт в разработке программного обеспечения, то, скорее всего, и без меня это знаете. Я много лет – почти десять – пробовал притворяться, что это неправда, но в конце концов пришлось взглянуть фактам в лицо. Вам тоже придется с этим смириться. Но не волнуйтесь: есть способы выделить какую-то часть, которую реально построить при имеющихся у вас временных и человеческих ресурсах.

В главе 3 вы прочитали о том, как в Globo.com использовали карты, чтобы уложиться в жесткие временные рамки. Компания сконцентрировалась на дедлайне, чтобы выделить достаточный для успеха объем работы. Затем бэклог был разбит на части: все, что не было необходимо для достижения нужного результата, отбросили. Таким образом появилось предположительное минимально жизнеспособное решение. И это не была грубая, слабо очерченная версия большой идеи – это было в самом деле блестящее решение, точно нацеленное на достижение необходимого результата к выборам в Бразилии. Ребята из Globo.com были уверены, что эта работа принесет пользу их бизнесу, рекламодателям, телевизионным компаниям и клиентам. Они подумали обо всех своих пользователях и были уверены, что делают для них что-то нужное. А разделив карту на части, они сформулировали решение, которое можно было реализовать силами своих команд за имеющееся время. Сочетание ценности, полезности и реалистичности для конкретных пользователей, заказчиков и задач и является жизнеспособным решением.

*Жизнеспособность означает успех решения для конкретной бизнес-стратегии, конкретных пользователей и заказчиков.*

Мы можем сразу же применить этот подход не только к первому, но и ко второму и к третьему релизам, чтобы обеспечить и их жизнеспособность. Но мы знаем: как только первый релиз выйдет в свет, мир изменится (и это хорошо). Это автоматически означает, что нам нужно будет пересмотреть свои будущие релизы с точки зрения новых условий, которые мы создали.

## Секрет расстановки приоритетов

Подойдите поближе, я хочу поведать вам один секрет.

Он известен немногим – во всяком случае, люди ведут себя так, словно ничего о нем не знают. Впрочем, быть может, они прикидываются тупицами с каким-то умыслом?..

Если вы какое-то время вращаетесь в мире разработки Agile, то наверняка много раз слышали выражение «расставить приоритеты историй по ценности для бизнеса». Само по себе это утверждение верно, но, видя слова «ценность для бизнеса», хотелось бы понимать, что конкретно под ними подразумевается. Вы и ваша команда должны определить, что в данном случае будет ценным для бизнеса.

Давайте снова вернемся к MadMimi. Гэри нужно было придумать продукт, который быстро завоевал бы определенный рынок, прежде чем закончатся деньги. Жизнеспособность, с точки зрения Гэри, означала, что у него были бы клиенты, полюбившие продукт и готовые за него платить. Затем он начал бы увеличивать аудиторию продукта и, как следствие, свою прибыль.

Данная бизнес-цель в сочетании с финансовыми ограничениями заставила Гэри сконцентрироваться на конкретных пользователях и их задачах, которых он решил поддерживать. Гэри не оставлял надежду создать «маркетинговый интерфейс музыкальной индустрии», давший Mimi название, но для начала решил сфокусироваться на задачах менеджера, рекламирующего свою музыкальную группу с помощью прямой электронной рассылки фанатам. Как только Гэри принял это решение, конкретные функциональности, над которыми следовало работать, стали яснее ясного.

Если вы сейчас читали внимательно, то, конечно, уже поняли, в чем секрет расстановки приоритетов.

Конкретные результаты, необходимые бизнесу, устанавливают фокус на конкретных пользователях, их целях, а также действиях, которые они проделывают в нашем продукте. Фокус же на действиях, в свою очередь, привлекает наше внимание к каким-то конкретным функциям и инструментам, которые нужны пользователям для достижения своих целей.

Работая над MadMimi, Гэри принял взвешенное решение сконцентрироваться на достижении целей менеджеров, рекламирующих свои ансамбли. *Это и была конкретная ценность, которую он поставил своей основной целью*. Он не использовал абстрактное понятие «ценность для бизнеса», а совершенно конкретно определил то, что будет иметь ценность для него.

Ошибка большинства людей в том, что они пытаются сперва расставить приоритеты функциональностей.

*Начните с расстановки приоритетов бизнес-целей, заказчиков и пользователей, а также их задач и лишь потом расставляйте приоритеты функциональностей.*


В следующий раз, услышав, как кто-то интересуется, у какой функциональности самый высокий приоритет, без обсуждения бизнес-целей, целевых пользователей и ценности для них, сочтите это хорошей возможностью начать задавать вопросы. Постарайтесь не быть слишком высокомерным: все-таки не все знают этот секрет.

# Действия, обсуждения и артефакты в исследовании

Существует очень много действий, которые вы и ваша команда можете проделать, и артефактов, которые вы можете создать, проводя исследование. Эта простая таблица даст вам некоторое представление о том, что вы можете использовать. Не стоит пытаться выполнить сразу все – это излишне, но и не ограничивайтесь лишь тем, что перечислено здесь, ведь наверняка существуют и другие методы, которые больше подойдут в вашей ситуации с учетом имеющихся у вас возможностей и навыков. Главное – не просто сидеть в комнате и писать бесконечное количество историй. Это глупо.

| **Сформулируйте идею** | |
|---|---|
| Используйте эти обсуждения, чтобы вспомнить, для чего ваша организация создает программные продукты, для кого и какие у вас есть показатели успеха | Перечисление бизнес-проблем, которые вы решаете.<br>Конкретные параметры бизнеса, на которые вы влияете.<br>Краткие списки конкретных заказчиков и пользователей.<br>Показатели, которые помогут оценить, как люди используют новую функциональность и насколько они ею довольны.<br>Крупные риски и допущения.<br>Обсуждения с владельцами бизнеса и экспертами предметной области |
| **Узнайте своих пользователей и заказчиков** | |
| Используйте дискуссии и исследования, чтобы понять своих заказчиков и пользователей, их нужды и способы работы на сегодняшний день | Список ролей и описаний пользователей.<br>Простые персонажи или профили пользователей.<br>Простые профили организаций, или «оргсонажи».<br>Карты историй, отражающие способы выполнения пользователями их задач на сегодняшний день, также известные как карты маршрутов.<br>Исследования пользователей и наблюдения за ними, заполняющие пробелы в сведениях о них |
| **Визуализируйте решения** | |
| Сконцентрируйтесь на конкретных пользователях и заказчиках, а затем представьте себе решения, которые могли бы им помочь.<br>Визуализируйте решения с помощью слов и картинок.<br>Проверьте свои решения с помощью заказчиков и пользователей | Карты историй.<br>Пользовательские истории и пользовательские случаи.<br>Эскизы UI и раскадровки.<br>Прототипы UI.<br>Архитектурно-технические и дизайнерские наброски.<br>Архитектурно-технические прототипы.<br>Большая совместная работа членов команды, пользователей, заказчиков, представителей бизнеса и экспертов в предметной области |

| **Минимизируйте объем работы и составьте план** | |
|---|---|
| Сформулируйте то, что представляется вам минимально жизнеспособным решением. Оцените объем работы как можно более точно, чтобы установить бюджет, необходимый для предъявления продукта. Составьте план разработки с учетом максимального снижения рисков | Используйте карты историй для разделения бэклога на части.<br>Применяйте оценки для определения бюджета на разработку |

# Цель исследования – выработка одинакового понимания

Приходилось ли вам когда-нибудь работать над проектом программного продукта, где никто как следует не представлял себе картину в целом? Приходилось ли вам сталкиваться с ситуацией, когда в разгар разработки команда обнаруживала, что необходимо сделать большой незапланированный кусок работы? Когда такое случалось со мной, нередко оказывалось, что среди членов нашей команды и людей со стороны, которые сотрудничали с нами, хоть кто-нибудь да имел необходимые сведения. Чтобы избежать проблем, нам нужно было всего лишь иметь полное взаимопонимание.

Отчасти история Гэри из главы 1 как раз об отсутствии общей картины у него самого и его команды разработки. Даже сам Гэри, в голове которого родился продукт, не осознавал до конца сложности и размера этого проекта. Визуализация продукта в виде ряда простых моделей помогла ему и всем, кто ему помогал, нарисовать одну и ту же цельную картину в своем сознании.

Для некоторых создаваемых вами вещей может быть вполне достаточно достичь взаимопонимания относительно заказчиков и пользователей, а также формулировки найденного решения. Но хочу предостеречь: ваши гипотезы о том, что вы создаете, вполне могут быть неверными. Не беспокойтесь, как раз в следующей главе я собираюсь рассказать о нескольких стратегиях, которые помогут вашим исследованиям стать максимально эффективными.

# Глава 15. Эмпирическое обучение через исследование

Вообще-то частично я ввел вас в заблуждение.

Многие из вас, наверное, читали предыдущую главу и другие главы перед ней, медленно закипая, так как я, очевидно, упускаю самое интересное. Сожалею об этом.

На самом деле истории, которые я рассказывал о MadMimi и Globo.com, еще не завершены. Это правда, что в обоих проектах использовались исследовательские обсуждения для выявления минимально жизнеспособного решения, но действительно ли именно это решение является жизнеспособным или нет, пока только догадки. По сути, все в этих историях – лишь догадки, пока продукт не представлен на рынок и мы не знаем, как на него реагируют пользователи и заказчики. Начальные исследовательские обсуждения, как и карты историй, помогли сделать хорошие стартовые предположения. Но для обоих проектов это было лишь начало куда более длинного пути к созданию действительно жизнеспособного продукта.

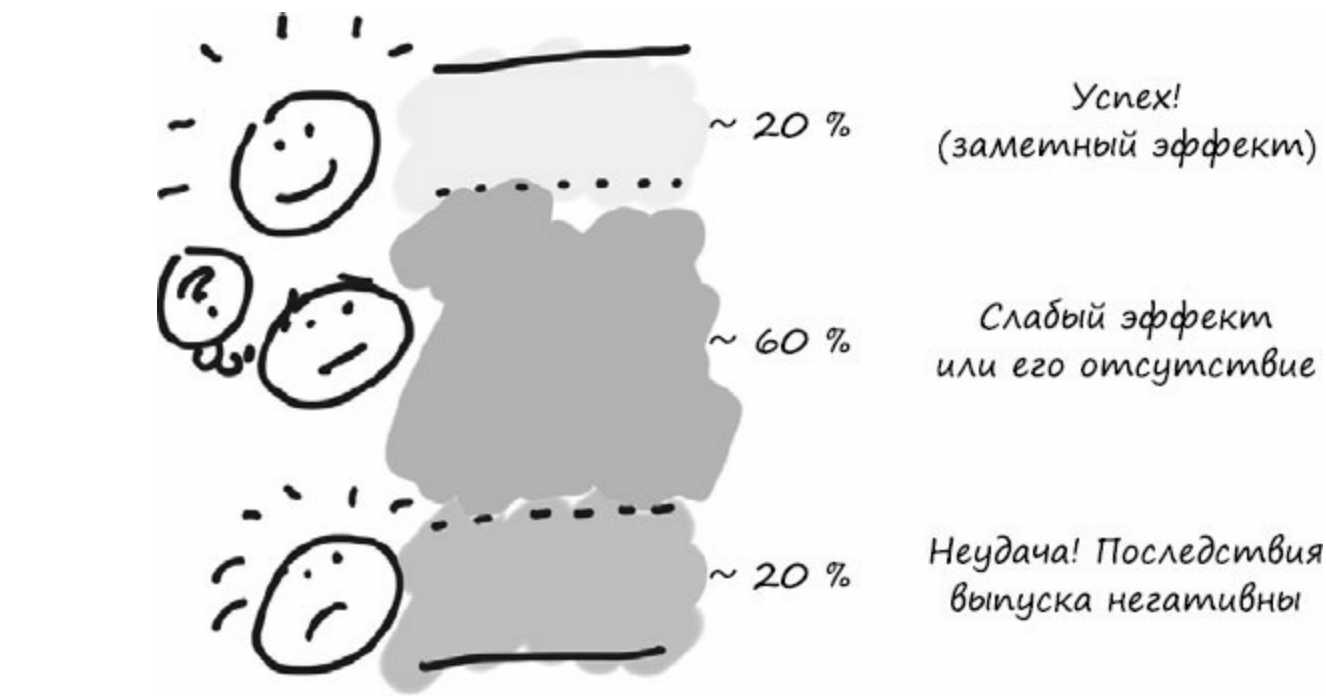
Таким образом, мы подошли к одной из крупнейших ошибок, которые обычно делают люди, – твердой уверенности, что минимально жизнеспособное решение непременно будет успешным.

# Большую часть времени мы ошибаемся

Я не исключение: как и многие другие, я склонен верить, что все мои блестящие идеи будут успешными. В прошлом я много раз реализовывал решения, которые, по моему мнению, непременно должны были «взлететь», но этого почему-то не происходило. Нельзя сказать, чтобы это были какие-то значительные провалы, они просто не давали особого эффекта. В конце концов я и моя компания научились думать по-другому. Это были не только мои идеи – мы *все* хотели бы, чтобы функциональности, которые мы внедряем, приносили пользу. Но в конце концов оказывалось, что функциональностью, которую мы добавили, пользуются лишь пара человек, остальным она неинтересна, и все понимали, что придется прекратить ее поддержку, чтобы сохранить весь остальной продукт.

В чем я убежден – не на основании каких-то конкретных исследований или наблюдений, просто из собственного опыта неудач и сходного положения дел в других компаниях, – очень небольшая часть того, что мы создаем, будет иметь успех или реальный эффект, на который мы надеемся. Я бы сказал, не более 20 %. Более того, около 20 % созданного нами будет иметь негативные последствия, то есть, попросту говоря, навредит нашему бизнесу. Я много раз наблюдал, как организации выпускали новую, лучшую версию своего сайта, а продажи после этого падали, или предлагали заказчикам новую версию своего продукта, которую те пробовали, но затем возвращались к старой версии. Вот о каких неудачах я говорю.

Но между ними находятся около 60 % – чуть больше или чуть меньше, неважно, – которые не являются ни успехом, ни неудачей. По сути, это большая проблема: на создание этих функциональностей или продуктов мы потратили значительное количество денег, но в результате не получили ровным счетом ничего.




Исследования, опубликованные организацией Standish Croup в отчетах Chaos прошлых лет, доказывают, что от 64 до 75 % функциональностей используются редко или не используются никогда[27]. А 75–90 % всех IT-стартапов (в зависимости от того, что считать

неудачей) не достигают успеха[28].

Такие данные очень демотивируют, если о них задуматься. Неудивительно: большинство организаций придерживаются стратегии «сделаем вид, что у нас все работает».

# Старые недобрые времена

В старые недобрые времена я, как правило, работал примерно так. Я приносил очередную блестящую идею, или, честно признаюсь, то, что было мне предложено генеральным директором или важным заказчиком как *их* блестящая идея. Я дорабатывал ее, конкретизировал, корректировал. Затем я и моя команда приступали к реализации. Разработка всегда занимала вдвое больше времени, чем мы ожидали, но об этой проблеме мы поговорим в следующих главах. Мы заканчивали. Мы представляли сделанное клиентам. Мы устраивали вечеринку. Иногда вечеринка предшествовала представлению продукта. Но в любом случае в конце концов все было сделано.

Вот что происходило затем. Как правило, люди начинали жаловаться, что предъявленные нами функциональности не работают так, как бы им хотелось. Иногда ни одной жалобы не поступало (это, как мы понимали позже, объяснялось тем, что никто на самом деле не использовал функциональность). Тем не менее мы делали вид, что это и есть успех, что все хорошо. Вероятно, именно так дело обстоит в компаниях, где работают некоторые из вас. И, признаюсь честно, я сам иногда скатываюсь к такому пути в своей работе. Не говорите никому. Все-таки предполагается, что я эксперт.

Но, к счастью, есть способы работы получше.

# Эмпатия, фокус, формулировка, прототип, тестирование

Несколько лет назад со мной связался один клиент и спросил, могу ли я помочь адаптировать процесс, который он назвал *мышлением дизайнера*. В организации этого клиента был внедрен процесс, типичный для Agile, и дела у них шли неплохо – неплохо в том смысле, что все предъявлялось вовремя и с хорошим качеством. Но, как известно, чем быстрее предъявляешь барахло, тем больше барахла получаешь. Это, конечно, звучит грубовато. Другими словами, клиент убедился, что корреляция между количеством программного обеспечения, которое они разрабатывали, и эффектом, который получали в результате, была невелика.

В то время я был известен как специалист по дизайну пользовательского взаимодействия и разработке Agile. Я подумал: «Раз я дизайнер и способен мыслить, значит, у меня есть мышление дизайнера». Но я был не прав. Это было не то, что имел в виду клиент. К счастью, я не высказал свою мысль вслух.

Мышление дизайнера – это способ работы, изначально внедренный компанией IDEO. Затем в школе дизайна Стэнфордского университета этот способ изучили и начали преподавать. В наши дни ему обучают во множестве университетов и используют во множестве компаний по всему миру.


В процессе дизайнерского мышления есть несколько этапов, которые в моем изложении

выглядят многообещающе. Но на практике и я, и большинство других людей склоняемся к тому, чтобы делать прямо противоположное. Неудивительно, что нас преследуют неудачи.

Первый шаг дизайнерского мышления заключается в обеспечении *эмпатии*. Я бы не назвал это исследованием, что, как правило, ожидается в процессе дизайна. Название *«эмпатия»* выбрано потому, что в результате исследовательской работы необходимо по-настоящему осознать и прочувствовать, каково это – быть пользователем вашего продукта. Для этого нужно отправиться туда, где находятся пользователи, познакомиться с ними, понаблюдать за их работой и в идеале поработать вместе. Разумеется, если вы создаете программное обеспечение для хирургов, никто не ждет, что вы займетесь любительской хирургией, но сделайте все, что от вас зависит, чтобы ощутить себя в их шкуре. Не забывайте, что при традиционных исследованиях, особенно если это автоматические инструменты и опросы, собирающие количественные данные, мы получаем только данные, но не эмпатию.

> *Общайтесь с пользователями и заказчиками напрямую. Испытайте на себе те проблемы, от которых вы хотите их избавить своим решением.*

Следующий шаг называется *определением (фокусом)*. На предыдущем шаге, обеспечения эмпатии, мы многому научились, теперь нужно извлечь из него суть для того, чтобы выработать одинаковое понимание. Для этого понадобится большая совместная работа: изложение историй, передача информации и выделение главного из того, что мы узнали. После этого можно выбрать конкретных людей, имеющих конкретные проблемы, на которых мы сконцентрируемся.

На этом этапе используйте карты историй для того, чтобы отразить на них актуальное положение вещей. Включите в них детали, отражающие то, что вы видели или узнали. Сфокусируйтесь на трудностях пользователей, на том, что их раздражает или, наоборот, радует. Используйте простых персонажей, чтобы отобразить образ пользователя, который объединяет все, что вы изучили. Выберите специфические проблемы, на которых будете фокусироваться.

> *Концентрируйтесь на небольшом количестве проблем. Точно их сформулируйте.*

Следующий шаг – *формулировка*. Если вы внимательно прочитали предыдущую главу, то помните упомянутую там простую методику проектной студии. Это и есть пример хорошего подхода к формулировке идеи. Как правило, в реальном бизнесе, где первый выдвинувший идею получает все пинки и все почести, кажется бессмысленным тратить время на выявление всех возможных идей. Может быть, вы помните, что ваши первые очевидные решения были именно очевидными. Поэтому, если важно найти действительно инновационное решение, подумайте об этом.

Мне нравится использовать карты историй в качестве контекста для формулировок. Применяйте карту, где показаны приятные и неприятные стороны работы, а также имеется другая информация о пользователях, а затем набрасывайте идеи прямо на карте. Записывайте идеи на карточках или стикерах и вставляйте их в карту в тех местах, где они наиболее уместны.

> *Сознательно придумайте несколько возможных решений проблем заказчиков и*

*пользователей.*

Следующий шаг – составление *прототипа*. Надеюсь, нам всем известно, что такое прототипы, но зачастую мы пренебрегаем их созданием, чтобы поскорей взяться за создание работающего продукта. В самом деле очень жаль! Затраты на создание простейшего бумажного прототипа невелики, но он помогает нам лучше обдумать свое решение. С его помощью мы можем сами попробовать взаимодействовать с продуктами. Создание простых прототипов из бумаги или средствами специальных программ занимает немного времени, но позволяет отфильтровать много идей, которые попросту не будут работать. Эмуляция реального использования продукта может помочь вам при формулировке идей и стимулирует к высказыванию дополнительных идей, которые улучшат решение.

> *Создавайте простые прототипы, чтобы проработать наилучшие решения. Доведите их до такой степени детальности, которая позволит пользователям и заказчикам оценить, действительно ли это решение поможет им справиться со своей проблемой.*

Последний шаг заключается в *тестировании*. Под этим я не подразумеваю проверки, выполняемые для поиска багов. Я говорю о проверке того, действительно ли ваше решение может помочь кому-то справиться со своими проблемами. Быть может, вы удивитесь, но иногда это возможно, даже если в продукте *есть* баги. Когда у вас наконец появляется прототип, который, по вашему мнению, решает проблему, на которой вы решили сфокусироваться, покажите его людям, которые будут использовать продукт. Вы должны не просто показать и рассказать, да и о продажах говорить пока рано. Потенциальные пользователи должны оценить этот прототип как нечто, что может решить одну из имеющихся у них проблем. Они должны использовать его для выполнения реальной задачи. Вы можете обеспечить это, смоделировав необходимое количество нужных данных.

> *Представьте свое решение людям, которые будут покупать или использовать ваш продукт. Не ожидайте успеха с первого раза, итеративно совершенствуйте решение.*

Важная особенность мышления дизайнера – такой способ работы, который поощряет небольшие мультидисциплинарные сплоченные команды к тому, чтобы быстро работать вместе, используя простые модели, эскизы, не слишком детальное документирование и общение. Это описание должно напомнить вам исследовательскую команду и ее партнеров, которых я описал в главе 12. А еще метод работы, в котором важное место отводится выработке одинакового понимания.

Использование элементов дизайнерского мышления помогает хорошо осмыслить проблемы, которые мы решаем, в результате чего мы всегда работаем над реальными, а не воображаемыми сложностями. Прототипирование и тестирование решений перед вложением значительных средств в создание полноценных масштабных продуктов поможет нам подтвердить, что мы создаем нечто действительно полезное людям.

Но само по себе дизайнерское мышление может вызвать некоторые проблемы.

# Хороший инструмент в неумелых руках

Процессы дизайна используются уже довольно долго. Мышление дизайнера как общий подход к разработке – тоже. С хорошо поставленным процессом дизайна дела идут намного лучше, чем в старые недобрые времена. Но не стоит путать процесс и умение работать в соответствии с ним: у вас не будет недостатка в возможностях сесть в лужу. К примеру, если вы видите, что хороший процесс дизайна занимает слишком много времени и все равно дает плохие результаты, то можете сделать вывод, что процессы такого рода не работают. Но дело вовсе не в процессе.


Вот несколько замечательных способов безнадежно испортить процесс дизайна.

• Не беспокойтесь о формулировании нужд бизнеса и целевой аудитории. Таким образом будет очень сложно выбрать, на ком концентрироваться в первую очередь, и нелегко оценить правильность своего решения.

• Потратьте уйму времени на доскональные исследования и формирование сути того, что вы изучили. Вопросы без ответов никогда не закончатся, но не сдавайтесь! (В этом случае очень полезно будет строгое ограничение времени на исследования.)

• Не тратьте время на разговоры с людьми и получение от них сведений. В конце концов, у вас есть множество данных! А идеи решений гениальны! Надо просто поскорей приступить к дизайну.

• Пропустите выбор нескольких проблем, на которых будете концентрироваться. Вместо этого поставьте цель решить все проблемы, которые обнаружите. Ведь чем больше проблем вы решите, тем лучше, не правда ли? (Но большие проблемы часто требуют больших решений! А попытка одним махом решить проблемы людей с взаимоисключающими нуждами, скорее всего, даст решение, которое не устроит никого.)

• Рассмотрите несколько решений, но для их разработки привлекайте только профессиональных дизайнеров, ведь лишь они обладают достаточной квалификацией и

только их идеи будут качественными.

• Не тратьте времени на рассмотрение нескольких идей, ведь одна замечательная у вас уже есть.

• Тщательно разработайте прототип, который выглядит совсем как настоящее приложение, и неважно, что он не выполняет того, чего хотят заказчики и пользователи. Зато, лишь увидев этот прототип, они тут же вскричали: «Как классно он выглядит!»

• Убедите себя, а затем и всех остальных, что тщательного исследования и профессионального дизайна достаточно для того, чтобы решение работало. В конце концов, вы строго придерживались дизайнерского процесса. Что может быть не так?

• Не беспокойтесь о стоимости реализации решения. Решение безоговорочно верное, и любая стоимость разработки оправданна.

• Когда вы предъявите свое решение заказчикам и пользователям, а затем не увидите результатов, которых ожидали, найдите в процессе какое-то упущение, которым можно объяснить ошибку. А лучше найдите лицо или группу, которых можно обвинить в своей неудаче.

Это, конечно, сарказм. Дело в том, что, хотя я обычно всецело за использование дизайнерского процесса, я также обычно больше всех жалуюсь на его применение. Могу также признаться, что чаще всего за все неудачи в таких процессах был ответственен именно я. Но из опыта последних лет я сделал выводы, как внести в эти процессы ряд улучшений.

# Короткие циклы с эмпирическим обучением

Эрик Райс – автор книги под названием «Стартап по методологии Lean»[29]. В этой книге Эрик рассказывает, как он угодил в ловушку, описанную мной ранее под названием «старые недобрые времена». Он участвовал в создании продукта, который собиралась выпустить его компания, в качестве технического директора. Успешность продукта не вызывала сомнений ни у кого, кроме заказчиков и конечных пользователей. Встречались положительные отзывы, отрицательные отзывы, явное безразличие. Компания, конечно, ожидала совсем не такого результата.

Одним из консультантов компании Эрика был Стив Бланк. Стив написал книгу «Четыре шага к прозрению»[30], где говорит, что первая вещь, о которой вы должны позаботиться, – не продукт, а клиенты. Он описывает процесс прогрессивного подтверждения достоверности используемых сведений: сперва вы убеждаетесь, что клиенты, которых вы нашли, заинтересованы в вашем решении, а затем – что придуманное решение заинтересует их настолько, что они согласятся его купить, использовать и порекомендовать другим. Бланк назвал все это процессом *эмпирического обучения*.

Ценнейший вклад Эрика Райса в разработку программных продуктов – это упрощение и продуцирование такого способа мышления в короткую мантру *«создание – обратная связь – извлечение опыта»*. Эрик подчеркнул, что времени на прохождение этого короткого цикла изучения требуется относительно немного. В традиционном процессе проектирования, как правило, затрачивается очень много времени на фазу исследования и дизайна – так много, что в конце концов вы привыкаете к решениям и не можете проверить, приводят ли эти решения к тем результатам, на которые вы рассчитывали. Если в традиционном дизайнерском процессе, как правило, требуются недели или месяцы на подтверждение верности идеи решения, то в процессе Lean Startup это, как правило, занимает всего несколько дней.

### *Что означает Lean?*

Хочу признаться, что почти все в подходе Lean Startup мне очень нравится. Единственное, что я не люблю, – это название. Нельзя сказать, что все в этой концепции так уж lean (экономно), да и сама концепция слишком значительна, чтобы использовать ее только в стартапах.

Название Lean (экономный) происходит, с одной стороны, от термина «экономное мышление», описанного в системе процессов Toyota несколько десятилетий назад, а с другой – от того же термина, популярного в настоящее время и широко используемого в различных контекстах, включая разработку программного обеспечения. В идеологии мышления Lean можно найти тонны отличных идей, и Lean Startup лишь слегка касается некоторых из них.

Эрик пытается рассмотреть случай, когда стартап конкурирует с крупным бизнесом, в результате чего работать приходится в ситуации высокого риска и неопределенности, которые и требуют особого способа мышления. Но я твердо уверен, что слова «В этом проекте особого риска нет и неопределенности тоже»

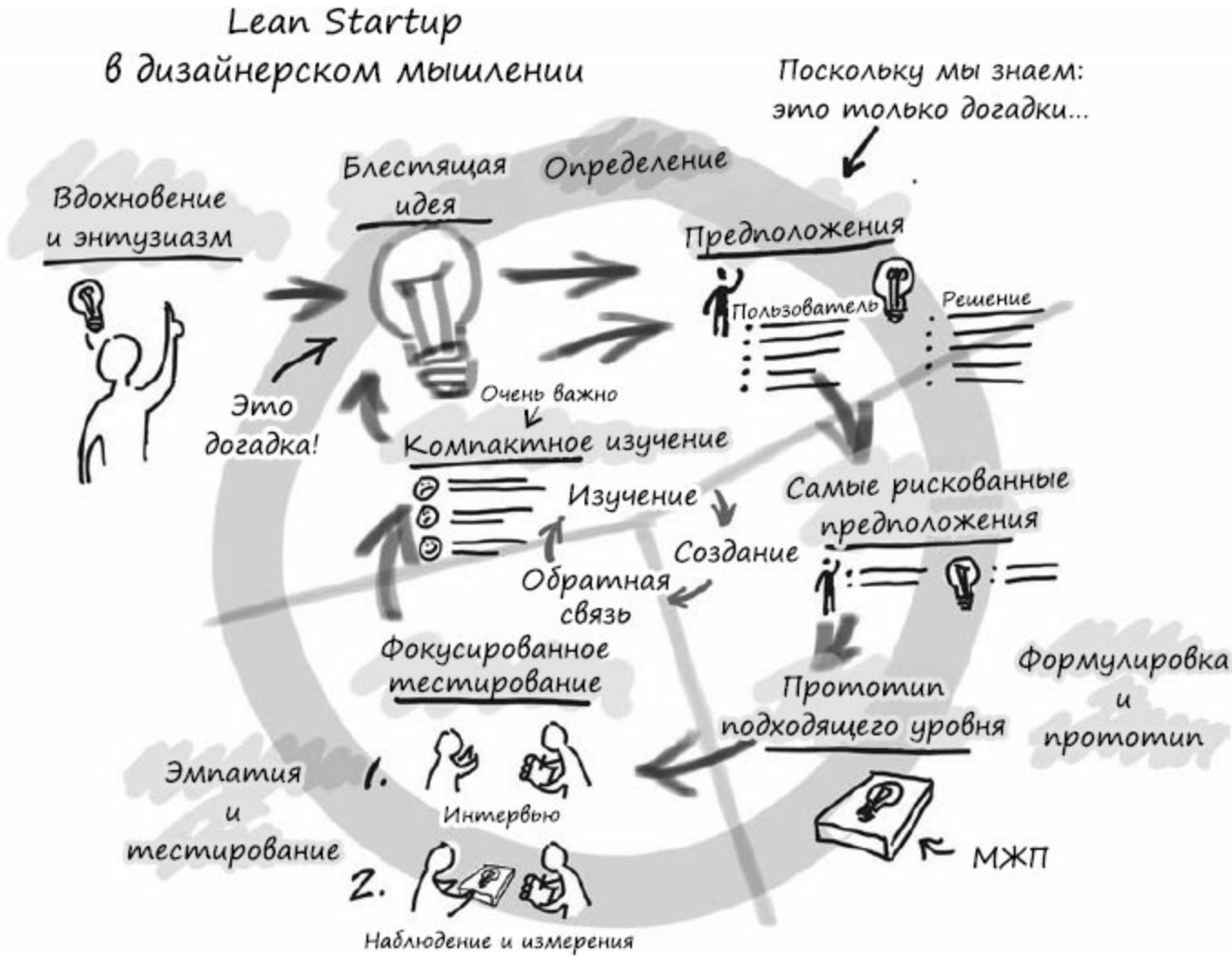
печально известны как финальные для многих проектов. Какая-то доля риска есть всегда, и стратегии изучения, описанные в процессах Lean Startup, очень полезны во многих контекстах. Поэтому не стоит оправдываться перед собой или другими людьми из-за применения методики, которая предназначена лишь для запуска нового бизнеса.

# Как метод мышления Lean Startup меняет дизайн продукта

В старые недобрые времена у нас не было иного пути, кроме как придумать блестящую идею, воплотить ее в жизнь и надеяться на хороший исход.

В попытке выбраться из этой ловушки мы могли бы использовать строгий процесс дизайна, временно отложив в сторонку наши блестящие идеи, а вместо этого глубоко погрузившись в исследования, чтобы разобраться в проблемах, которые мы решаем.

Вот как я рекомендую применять стратегию Lean Startup, работая сегодня.




## Начните с догадок

Да, с догадок.

В старые недобрые времена вы так или иначе оперировали догадками, просто притворяясь, что это не так. В процессе проектирования вы не должны допускать догадок, но другого пути не было, поэтому вы их допускали, но притворялись, что ничего такого не было. Поэтому просто перестаньте притворяться.

На самом деле это ведь не просто догадки. Отчасти что-то берется из опыта, отчасти из интереса к теме, понимания смысла того, что вы делаете, и, конечно, отчасти из предположений, но таким образом и запускается процесс. Я суммирую свои предположения

и догадки о том, кто мои пользователи, обычно составляя простые прототипы. Я описываю, как, по моему мнению, они работают сейчас, составляя карты историй по состоянию на сегодняшний день. Все это я делаю совместно с другими людьми, у которых есть непосредственный опыт работы с пользователями и заказчиками. А в некоторых ситуациях на этом этапе я привлекаю к работе пользователей и заказчиков напрямую. Поэтому фактически многие догадки моей команды не являются догадками по сути. Но и исследованием в точном смысле слова их назвать нельзя. На такую работу уходит от нескольких часов до нескольких дней – ни в коем случае не недели или месяцы.

После того как у всех нас появилось одинаковое представление о пользователях программного продукта и о проблемах, на которых мы концентрируемся, можно предположить, каковы будут возможные решения. Мы снова используем принципы дизайнерского мышления – выдвигаем и рассматриваем несколько независимых решений, но стараемся как можно быстрее прийти к соглашению о том, какое решение будет самым лучшим. Иногда одно решение выбрать не получается и приходится брать в работу несколько. Не следует переоценивать важность этого этапа, так как, очевидно, мы в любом случае делаем какие-то ошибки.

## Определите рискованные предположения

Поскольку мы выдвинули немало догадок о наших пользователях и их сегодняшних проблемах, давайте явно назовем эти догадки. В частности, потребуется совместная работа по составлению списка того, что мы считаем соответствующим истине, но если окажется, что это не так, придется переделывать все заново.

То же самое нужно сделать в отношении нашего решения. Следует подумать, какой реакции на него мы ожидаем от своих пользователей и как, по нашему мнению, они будут использовать это решение. Мы формируем в уме какие-то гипотезы о возможных путях взаимодействия пользователей и нашего продукта. Мы также обсуждаем технические риски – то, что может поставить под угрозу реализацию нашего решения.

Имея списки догадок и предположений о заказчиках, пользователях и нашем решении, мы можем выделить самые крупные, по нашему мнению, риски.

## Дизайн, реализация и небольшие проверки

Именно здесь заметны наибольшие *отличия*.

В старые недобрые времена мы планировали и создавали весь продукт целиком. В процессе дизайна прототипировали весь продукт целиком или большую его часть. Следуя подходу Lean Startup, где наша цель – учиться как можно скорее, мы делаем все от нас зависящее, чтобы сделать самый простой и быстрый прототип. Во многих случаях получившееся даже нельзя, строго говоря, назвать прототипом.

Вот пример из жизни моих друзей из некоммерческой организации под названием ITHAKA. Они работали над продуктом под названием JSTOR. Если в последние десять лет вы учились в каком-нибудь американском колледже, то, скорее всего, пользовались JSTOR в библиотеке, чтобы найти статьи или книги для заданного вам реферата.

Студенты, пользующиеся продуктом, хотели бы с легкостью делать это в любом месте –

в кофейне, дома, во время путешествия. Но подключиться к JSTOR, находясь вне колледжа, было очень сложно. Нужно было ввести имя пользователя и пароль сети колледжа, так что, сидя где-то в кофейне, студенты должны были авторизоваться и получить доступ ко всем ресурсам, лицензированным университетом. Команда JSTOR уже нашла решение, но использовать его было трудновато. Они хотели проверить новый способ взаимодействия.

Вот какие предположения о студентах они сделали.

• Работают из кофеен и общежитий.

• Не знают, что могут подключиться к JSTOR из этих мест.

• Или знают, но думают, что это очень трудно.

Вот какие предположения были сделаны о решении.

• Интуитивно понятное.

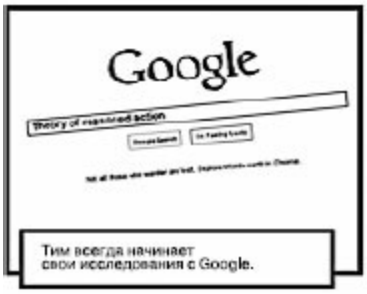
• Студентам должно быть очевидно, что у них есть полный доступ к JSTOR без необходимости присутствовать в библиотеке.

Чтобы проверить эти предположения, команде JSTOR не нужно было создавать работающее программное обеспечение, для этого пока было рановато. Все, что нужно было сделать, – спросить студентов о том, где конкретно они находятся, когда занимаются исследованиями, особенно когда возникает необходимость в JSTOR. Команде нужно было убедиться, что у студентов и в самом деле возникают предполагаемые сложности и, следовательно, они сочтут, что решение команды JSTOR их устранит.

Команда планировала пообщаться с множеством студентов. А чтобы упростить работу и описать проблему и ее решение более точно, они нарисовали простой дизайнерский комикс. Если вы не сталкивались с подобным раньше, то сейчас убедитесь, что его суть полностью соответствует названию. Он выглядит в точности как страница из книжки комиксов, но вместо супергероев, сражающихся со злодеями, там показаны обычные люди, решающие реальные проблемы с помощью вашего дизайнерского решения.

Вот несколько страниц из дизайнерского комикса JSTOR[31].


Тим находится в Common Grounds, кофейне рядом с общежитием, и хочет сделать небольшое исследование. К завтрашнему дню он должен написать работу к семинару по маркетингу по материалам трех научных статей.


Google
Тим всегда начинает свои исследования с Google.

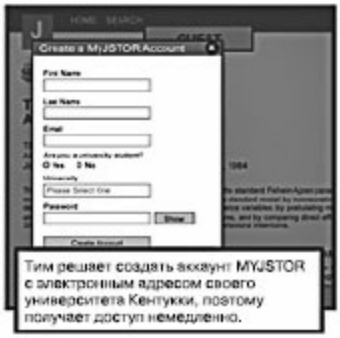

В результатах поиска он находит статьи на JSTOR.

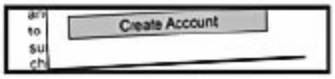

GUEST
CHICAGO JOURNALS
The Theory of Reasoned Action
Applied to Coupon Usage
Щелкнув на одной ссылке, он попадает на страницу со статьей.


GUEST
Страница идентифицирует его как гостя.

1


Next Page
Read the whole Article
Когда Тим нажимает кнопку, открывающую следующую страницу, система предлагает ему выбрать способ доступа к статье.

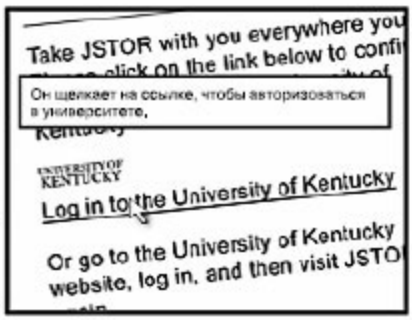

Create a MyJSTOR Account
Тим решает создать аккаунт MYJSTOR с электронным адресом своего университета Кентукки, поэтому получает доступ немедленно.


Create Account

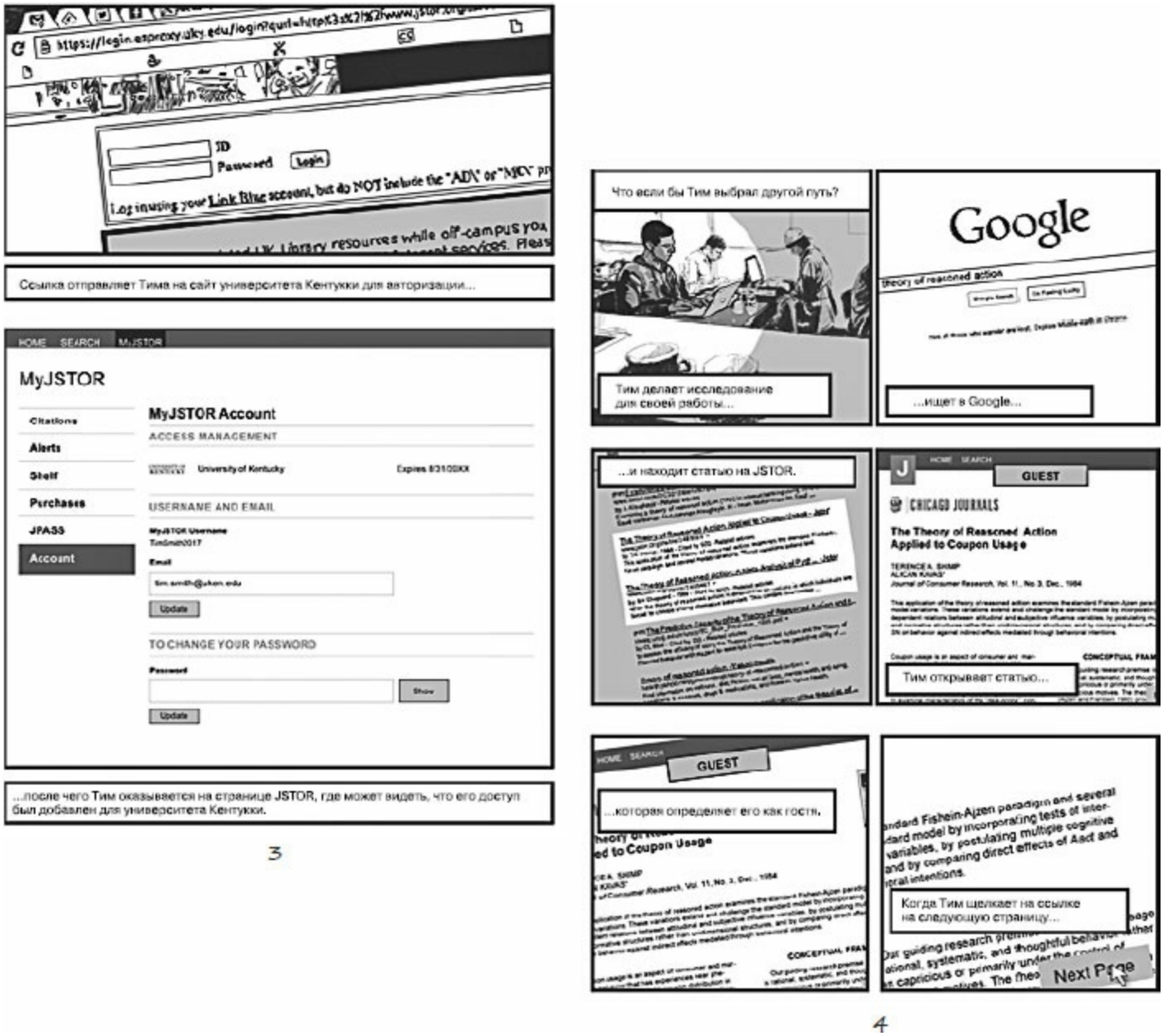

Registration Complete!
Тим завершает регистрацию и теперь может читать статью дальше.


Он продолжает свои исследования на JSTOR и заканчивает работу.


Later...
Welcome to your MyJSTOR account!
Take JSTOR with you everywhere you go. Please click on the link below to confirm that you are part of the University of Kentucky
Log in to the University of Kentucky
Тим проверяет почту и находит сообщение от JSTOR, предлагающее ему завершить регистрацию.


Take JSTOR with you everywhere you
Он щелкает на ссылке, чтобы авторизоваться в университете,
UNIVERSITY OF KENTUCKY
Log in to the University of Kentucky
Or go to the University of Kentucky website, log in, and then visit JSTO

2




Тест, разработанный командой, требовал, чтобы ее члены потратили немного времени на интервью со студентами и спросили их о трудностях, которые они сейчас испытывают. После этого они вместе со студентами рассмотрели историю, чтобы проверить, может ли их разработка решить проблемы студентов. Они не создавали полноценный прототип. У них *имелись* сомнения в том, что их решение будет полезным, и этого нельзя было проверить с помощью книги комиксов. Существовали также определенные технические проблемы, которые требовали написания некоторого количества кода для проверок и тестирования. Но все это не имело бы никакого значения, если бы оказалось, что никаких проблем у студентов на самом деле нет, а идея никому не интересна.

Минимально возможное для тестирования решение – то, что Lean Startup называет *минимально жизнеспособным продуктом*. Да, Эрик Райс знает, что это не полноценный продукт. Но если ваша цель – обучение, это *самый маленький* продукт, который вы можете создать, чтобы научиться чему-то.

## Получайте обратную связь из тестирования с пользователями и заказчиками

Попросите пользователей и заказчиков протестировать продукт. На ранних стадиях работы это чаще всего означает планирование интервью и время, проводимое в беседах с людьми. Если вы создаете решение для определенного круга потребителей, то можете провести коридорные интервью, то есть отправиться туда, где бывают ваши заказчики или пользователи, остановить их и поговорить с ними. Я часто составлял компанию моим коллегам, когда они ходили с этой целью в торговые центры, кофейни или привлекательные для туристов места.

JSTOR просил студентов и аспирантов уделить немного времени исследованию, после чего в течение 30–60 минут их спрашивали, как они справляются со своими задачами сегодня, так что команда могла подтвердить свои предположения относительно проблем, которые решали ее члены. А после показывали дизайнерский комикс и просили высказать свое мнение об идее решения.

## Пересмотрите свое решение и предположения

Запустив тест несколько раз, вы начнете получать все более схожие результаты. Если вы шли в абсолютно неверном направлении, то поймете это очень скоро. Обдумайте то, что узнали. Скорректируйте свои представления о пользователях и способах их работы на сегодняшний день с учетом полученной информации. Используя эти данные, переработайте принятое решение. Затем обдумайте допущения относительно пользователей и решений. После этого разработайте следующий тест.

Запустив свои тесты, ребята из JSTOR обнаружили, что у многих студентов не было тех проблем, которые они предполагали. Обычно такого рода новости расстраивают, ведь никто не любит оказываться неправым. Но с точки зрения подхода Lean Startup это прекрасные новости. Прекрасные потому, что ошибочное направление обнаружилось всего за несколько дней, а не после многих недель командной работы над созданием программного обеспечения.

Если вы придерживаетесь такого подхода, труднее всего будет научиться радоваться тому, что узнали что-то новое, а не беспокоиться о том, что вы можете быть не правы.

*В подходе Lean Startup самая большая ошибка – завалить этап обучения.*

В подходе Lean Startup *разработка* означает проведение настолько маленького эксперимента, насколько возможно. Обратной связью может быть *аналитика*, собранная от работающего программного продукта, а также прямые наблюдения на интервью и пользовательском тестировании прототипов. *Обучение* – это то, что мы делаем с информацией. Это может быть пересмотр предположений и изменение того, что мы считаем лучшим решением, на всем протяжении работы.

# Истории и карты историй

Вы можете спросить: «А какое отношение к этому имеют истории и карты историй?» Это хороший вопрос.

На всем протяжении процесса эмпирического обучения вы будете постоянно составлять истории о ваших пользователях, о том, что они делают и почему. Вы станете использовать карты историй, чтобы рассказывать длинные истории о том, как люди работают сейчас, и о том, как, по вашему мнению, они могут применять ваше решение. Когда придет время создавать прототипы, вы используете истории и материалы обсуждений, чтобы прийти к соглашению о том, как должен выглядеть прототип, а также что использовать в качестве критериев его готовности. Один раз поняв, как работают истории, очень трудно рассказать что-то без их использования.

Впрочем, тут существует значительное отличие от использования историй на стадии исследования. Обычно, применяя истории, мы разговариваем с разработчиками, тестировщиками и множеством других людей о программном продукте, который мы собираемся создать и выпустить в мир. Мы много работаем над тем, чтобы добиться одинакового понимания. Мы с головой погружаемся в детали реализации нашего продукта и можем раздобыть достаточно данных, чтобы довольно точно оценить необходимые временные затраты. Обычно мы говорим сразу о нескольких историях, поэтому можно прийти к соглашению, сколько из них можно успеть сделать за двухнедельный спринт или итерацию. Но во время исследования мы работаем гораздо быстрее. На создание нескольких простых прототипов нужна пара часов, а не несколько дней. Даже на прототипы, которые требуют написания кода и использования реальных данных, нужны дни, а не недели. Мы создаем, чтобы обучаться, мы ожидаем, что большинство наших идей будут неудачными или как минимум потребуют усовершенствования для того, чтобы быть успешными. Поэтому особое внимание уделяем тому, чтобы быстро работать вместе, быстро приходить к соглашениям и как можно меньше времени тратить на формальности.

Во время исследований и эмпирического обучения вы можете постоянно составлять истории, фрагментировать идеи, работать над маленькими частями, которые удобно создавать, и приходить к соглашению, что именно нужно создать. Все это обычно происходит так быстро, что само по себе использование историй может быть незаметно. Но оно есть.

# Глава 16. Огранка, полировка, разработка

Что теперь? Если истории нужны для планирования и управления дискуссиями, с помощью которых разрабатывается программное обеспечение, похоже, всем нам придется очень много разговаривать.

# Карточки, обсуждения, много карточек, много обсуждений…

Первая серия обсуждений помогла вам проникнуть в смысл открывшейся возможности. Вы обсудили, кто мог бы использовать ваш продукт, и представили, каким образом эти люди могли бы извлечь для себя пользу. Ваши дискуссии были достаточно глубоки, чтобы разбить большую возможность на компактные части и затем решить, какие из них должны войти в ближайший релиз, а какие можно отложить. Вы собрали истории, которые описывают следующий жизнеспособный релиз, в бэклоге.

Если вы разумный человек, а я уверен, что это так, следующие обсуждения затронут то, как должно выглядеть приложение, как оно будет себя вести, как изменения встроятся в уже существующий продукт и программную архитектуру. Наибольшее внимание в этих дискуссиях уделяется рискам. Вы разделили идею на части, которые можно быстро разработать и в результате как можно больше изучить за короткое время. А поскольку вы разумный человек, истории в вашем бэклоге разделены по следующему принципу: на ранних этапах – изучение, в середине – активная разработка, в конце – совершенствование и корректировка.

Сейчас наконец настало время для проведения *лучших*, завершающих обсуждений.

# Огранка и полировка

Мы приступили к работе с огромным энтузиазмом. Мы уверены: разработка программного обеспечения, которое описано в наших историях, будет идти гладко и без сюрпризов, если мы точно сформулируем, что именно хотим разработать. Но даже после всех этих обсуждений и составления историй обычно обнаруживается некоторое количество острых углов, на которые все постоянно натыкаются. Скорее всего, во время обсуждений не удалось внимательно рассмотреть эти моменты, как следует разобраться, какими они должны быть, и точно предсказать, сколько времени потребуется на их разработку. Но не стоит расстраиваться – у нас есть волшебная машина, которая быстро все исправит.

Представьте себе компактную и красивую волшебную машину. В большую воронку с левой стороны мы бросаем грубоватые шероховатые истории из нашего релизного бэклога. Изнутри машины доносятся свист, звон и клацанье. После этого из тонкой трубочки с правой стороны выходят маленькие, аккуратно отполированные изделия. Каждое изделие – это элемент, который команда может использовать, чтобы разработать идеальное, изящное, высококачественное программное обеспечение.

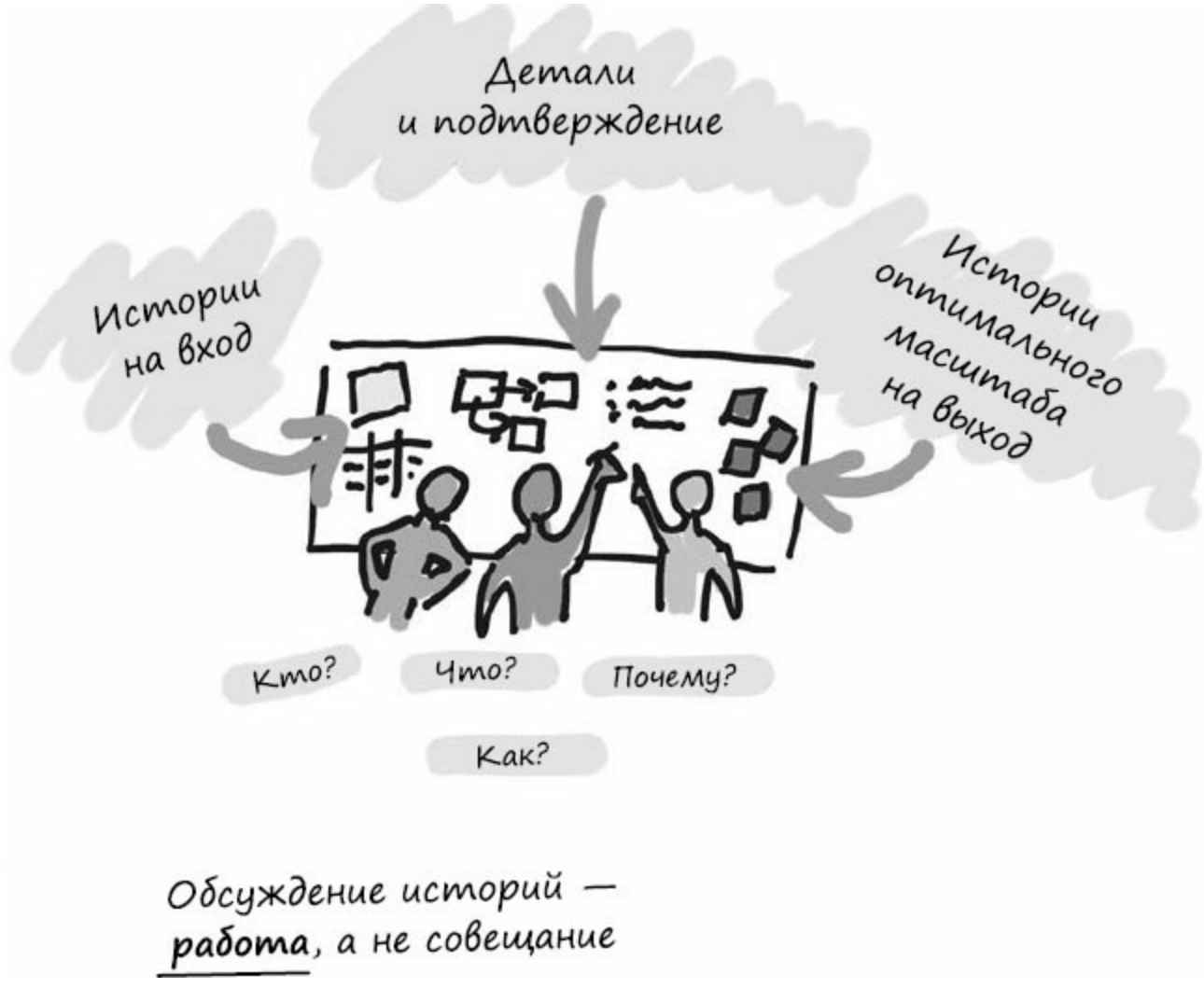
Со стороны такая машина и в самом деле кажется волшебной. Но внутри нее вы и ваша команда обсуждаете, дорабатываете и совершенствуете попавший в машину элемент. Этот особый секретный механизм, скрытый внутри машины, называется *семинаром по историям*.

Как вы, вероятно, помните из главы 11, семинары по историям – это небольшие, но продуктивные обсуждения, где все работают вместе, чтобы проговорить истории еще один, последний раз и в процессе твердо решить, что именно они собираются создать. Это очень глубокие обсуждения историй, результатом которых становится подтверждение принятых решений. Это значит, что мы подошли к третьему «П» в триаде «пишем – проговариваем – подтверждаем». И именно это «П» поможет нам окончательно огранить и до блеска отполировать наши камни.

# Проведение семинаров по историям

Вам понадобится небольшая группа людей, куда должны войти программист, тестировщик, специалисты по изучению пользователей и интерфейса – дизайнеры UI или аналитики в зависимости от того, как принято в вашей организации. Чтобы работа у доски была эффективной, группа не должна быть очень большой. Обычно – от трех до пяти человек.

На семинаре нужно работать, а не просто разговаривать. *Совещание* давно уже стало эвфемизмом непродуктивного совместного времяпрепровождения. В течение семинара должно состояться много продуктивных обсуждений, детальных объяснений. Нужно чертить схемы на доске и рисовать эскизы в блокноте. Необходимо сотрудничество, чтобы как можно точнее решить, что именно мы разрабатываем. Нужно стремиться к тому, чтобы все участники покинули комнату совещаний с прочным ощущением одинакового понимания, поэтому должно обеспечиваться пространство для продуктивных обсуждений на словах и в картинках.




Предметом всех обсуждений до этого момента были детали, но мы погружались в них ровно настолько, чтобы принять необходимые на тот момент решения. Теперь же нужно принимать решения, чтобы точно ответить на вопрос: «Что конкретно мы разрабатываем?»

Возможно, именно при этом обсуждении выяснится, что ваша история слишком велика. Я подразумеваю под этим – слишком велика для предмета, который можно немедленно

отдать в разработку и на его реализацию понадобится примерно пара дней. Да, размер не *всегда* оказывается слишком большим, но если вы подозреваете что-то в этом роде, просто проверьте и порадуйтесь, если убедитесь в обратном. В любом случае, в комнате вместе с вами находятся как раз те люди, которые должны помочь разбить эту историю на более мелкие. Размер историй должен быть таким, чтобы было удобно их реализовать, тестировать и выпустить в рамках продукта, над развитием которого работает вся команда.

### *Рецепт семинара по историям*

Используйте семинары по историям, чтобы довести понимание до совершенства и точно определить, что конкретно должна создать команда разработки. Семинар – это обсуждение продукта (поддерживаемое множеством данных и картинок), которое поможет команде принять необходимые решения и определить признаки готовности, то есть критерии приемки того, что было решено разработать.

*Перед началом семинара* нужно уведомить команду о историях, которые вы планируете обсуждать. Развесьте их на стенах или разошлите по электронной почте. Попросите команду проголосовать, учтите их мнение при окончательном выборе.

*Небольшое количество людей – условие продуктивности семинара*. От трех до пяти человек – наилучший вариант.

*Тщательно подбирайте участников*. Чтобы обсуждение было эффективным, включите:

• кого-то, кто хорошо понимает пользователей и пользовательские интерфейсы, их требования и возможности, – чаще всего это владелец продукта, специалист по пользовательскому взаимодействию или бизнес-аналитик;

• одного или двух программистов, имеющих представление о коде, который вы хотите добавить в продукт, и поэтому хорошо понимающих степень реалистичности разработки;

• тестировщика, отвечающего за качество продукта, потому что именно он задаст каверзные вопросы, известные как «Что, если», о которых не подумают остальные, обычно настроенные оптимистично.

Другие люди и другие роли тоже могут быть полезны, но не забывайте об оптимальном количестве участников дискуссии – не больше, чем в компании за обеденным столом.

Вы убедитесь, что один человек вполне может выступать в двух ролях, или, как говорят, носить сразу две шляпы. Например, во многих IT-организациях я встречал тестировщиков, одновременно исполняющих обязанности бизнес-аналитиков. Но если силами присутствующих не удалось преодолеть все камни преткновения, приостановите семинар и найдите кого-то еще, кто может помочь в решении трудного вопроса.

*Тщательно анализируйте и рассматривайте варианты*. Используйте обсуждения, чтобы детально проанализировать:

• кто конкретно ваши пользователи;

• как конкретно, по мнению команды, они будут использовать продукт;

• как конкретно будет выглядеть продукт, то есть его интерфейс;

• как конкретно ведет себя программный продукт внутри интерфейса – какие бизнес-правила действуют, как нужно валидировать данные;

• сколько времени потребует разработка продукта с учетом всех технических деталей (на этом этапе мы обсуждаем все настолько подробно, что оценки могут быть довольно точными).

Помните: пока еще ничего не решено окончательно. Если дискуссия приводит к решениям, которые слишком сложны или дороги, вернитесь на шаг назад и обсудите проблемы, которые вы решаете, а также альтернативные способы решения.

*Придите к соглашению о том, что нужно разработать*. После того как по результатам обсуждения вам удалось выработать одинаковое понимание, перейдите к поиску ответов на вопросы.

• По каким критериям можно будет определить, что работа над программным обеспечением закончена?

• Что будет происходить во время демонстрации программного обеспечения, когда мы будем оценивать его все вместе?

*Говорите и записывайте*. Используйте доски или большие листы бумаги, чтобы рисовать картинки, записывать примеры и рассматривать варианты. Не позволяйте своим решениям испариться. Записывайте их на досках или на бумаге, чтобы каждый мог их видеть. Сфотографируйте записи и картинки, чтобы позднее задокументировать информацию.

*Оперируйте примерами*. Где только возможно, используйте специфические примеры поведения пользователей: какие конкретно данные могут быть введены, что именно пользователи увидят в результате, – словом, любые примеры, которые лучше всего иллюстрируют вашу историю.

*Разделяйте и уменьшайте*. Обсуждая детали и думая о затратах на разработку, вы, возможно, обнаружите, что истории слишком велики, чтобы уместиться в стандартный цикл. Поработайте вместе с группой, чтобы разделить крупные истории или уменьшить их, удалив все лишнее.

Но все это не будет работать, если:

• нет совместной работы – кто-то один описывает, что нужно сделать, а все остальные слушают;

• разговор идет только о критериях приемки, но никто не вспоминает об истории и о «Кто?», «Что?», «Почему?»;

• не рассматриваются варианты реализации как с функциональной, так и с технической стороны.

# Планирование спринта или итерации

Некоторые практики Agile воспринимают эти важнейшие обсуждения историй во время сессий планирования как планирование итерации или спринта. Это не так страшно, если вы имеете дело с командами, эффективно работающими вместе и начинающими дискуссию, хорошо понимая своей продукт. У команд, с которыми я работал на протяжении многих лет, это был устоявшийся способ работы.

Тем не менее самая частая жалоба, которую я слышу от команд, работающих по Agile, заключается в том, что эти совещания по планированию – зачастую долгие изматывающие мероприятия. В какой-то момент все соглашаются разработать то-то и то-то, даже если на самом деле никакого общего понимания нет, просто чтобы закончить это мучительное совещание.

## *Быть среди своих*

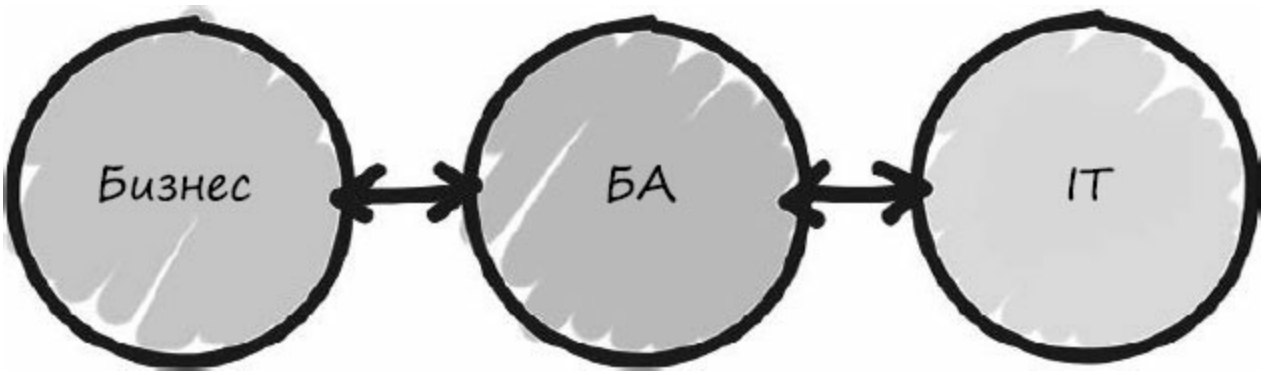
*Никола Адамс и Стив Барретт, RAC Insurance (Perth, Australia)*

*Мой первый опыт работы в мире проектов Agile в роли бизнес-аналитика был тяжелым и горьким уроком о превосходстве командной работы над печатным словом.*

*Никола Адамс*

**Контекст**

Историю трансформации от методологии водопада к Agile в компании RAC Incurance (Западная Австралия) рассказывает Никола, квалифицированный бизнес-аналитик с большим опытом в традиционном подходе к разработке программного обеспечения. В ее обязанности входило сотрудничество с бизнесом, понимание предметной области и составление функциональных спецификаций, в соответствии с которыми команда IT создавала продукт. Коммуникация происходила по следующей схеме.




Наибольшее внимание уделялось детальным спецификациям, где старались не пропустить ни одной детали. Поскольку было очевидно, что разработчики не читают документацию, были внедрены специальные стратегии смягчения проблемы (например, пошаговые руководства по спецификациям), но они не

решали ее полностью. Как правило, между окончанием работы над спецификацией и моментом, когда ее нужно было использовать для поддержки разработки и тестирования, проходило довольно много времени.

**С чего все началось?**

Устоявшийся подход к письменным спецификациям не стали отменять в одночасье. Концепцию написания требований на обратной стороне карточки внедрить было нелегко. Как Никола могла ожидать, что разработчики и тестировщики внедрят необходимую функциональность, за которую она несла ответственность, если у них нет необходимой информации? Фокус просто переместился на создание последовательностей действий пользователя, почти ничем не отличающихся от традиционных функциональных требований, за исключением размера; схема коммуникации не изменилась.

Чтобы передать требования команде на сессиях разработки, Никола:

• собирала требования партнеров по бизнесу;

• глубоко анализировала требования и данные;

• создавала последовательности действий пользователя (каждая занимала от одной до пяти страниц), где описывались требования, дизайн решения и критерии приемки;

• излагала эти требования команде с использованием проектора и отвечала на вопросы присутствующих.

К несчастью, результаты не впечатляли. Подготовительные сессии были скучными и утомительными, большая часть команды никак не включалась в обсуждение. Кроме того, Никола чувствовала, что зря тратит время на подготовку историй, так как команда чаще всего игнорировала их во время разработки.

После одной из сессий Сэм, эксперт в предметной области, одновременно играющий роль владельца продукта, заметил: «Если это и есть проект Agile, я не желаю в этом участвовать!»

Нужно было что-то делать.

**Что поменялось?**

Менеджер проекта Стив собрал команду, чтобы обсудить проблему. В результате команда приняла несколько важных решений: исключить из процесса документы с историями, привлечь к подготовке историй как команду бизнеса, так и разработчиков, а также явно включить в процесс окончательную обработку бэклога и работу над историями.

Теперь Никола не просто выполняла требуемые от нее действия, а активно участвовала в процессе. Следующая сессия разработки разительно отличалась от всех предшествующих. В конференц-зале не сидела скучающая команда, пассивно наблюдающая за экраном, где демонстрировались истории действий пользователя. Члены команды, включая владельца продукта, экспертов в предметной области и разработчиков, активно работали с визуальными моделями и другими артефактами и живо обсуждали истории.


Схема коммуникации изменилась. Никола больше не работала «испорченным телефоном» между бизнесом и IT – теперь она выступала координатором обсуждений между теми, кто понимал цели бизнеса, теми, кто хорошо знал нужды пользователей, и командой разработки, которая знала, что можно реализовать.


И представителям бизнеса, и команде разработки очень понравился новый формат, все они активно включились в процесс. Наконец удалось достичь одинакового понимания проблем, требующих решения, различные точки зрения, существовавшие в группе, позволили команде найти оптимальные решения в условиях ограничений, а Никола уже не чувствовала такого сильного давления и больше времени посвящала работе. Никола наконец ощутила себя своей!

# В толпе не может быть сотрудничества

Утомительные совещания по планированию – такая распространенная проблема, что многие команды интуитивно предпочитают проводить обсуждения историй в дни, предшествующие таким совещаниям. Как правило, такие обсуждения занесены в их ежедневники как «совещания по предпланированию», «окончательная обработка бэклога» или «лакировка бэклога». Но что происходит чаще всего? Та самая тягомотина, которую они так ненавидят на планировании, просто переносится на другой день. Словно пытаясь усугубить проблему, членов команды просят прервать их текущую продуктивную работу, чтобы посидеть на скучном совещании. Неудивительно, что никто не испытывает восторга по этому поводу.

Проблема не в том, что обсуждать истории очень сложно. Да, иногда это может вызывать трудности, но абсолютно любые обсуждения станут гораздо тяжелее, если вы попытаетесь привлечь к ним слишком много людей. Если к тому же большая часть этих людей не заинтересована в участии или не мотивирована, считайте, что дело провалено. Вы знаете, о ком я говорю, – эти люди надеются, что никто не замечает их смартфоны, на которых они играют под столом.

*Позвольте членам команды самим решать*, хотят ли они участвовать в обсуждении. Если позднее они будут жаловаться на принятые решения, просто не забудьте пригласить их в следующий раз.

*Если хотят участвовать все, попробуйте шаблон сотрудничества «Аквариум»*, описанный в следующем примере. В этом случае заинтересованные могут прийти, принять участие, если хотят, или уйти, если обнаружат, что не происходит ничего интересного.

### *Шаблон сотрудничества «Аквариум»*

Если среди ваших сотрудников найдется несколько человек, которые очень хотят, чтобы их пригласили на обсуждение, но количество участников и без того так велико, что под угрозу поставлена продуктивность, попробуйте шаблон сотрудничества «Аквариум». Он заключается в том, чтобы дать им возможность присутствовать, не участвуя активно или минимально влияя на окончательный результат. Чаще всего люди обнаруживают, что происходящее не настолько важно, как они ожидали. Спустя какое-то время вы и сами заметите, что они охотно предоставляют другим возможность обсуждать детали, а затем узнают результаты из последующих обсуждений.


Процесс происходит так: от трех до пяти человек работают вместе у доски. Это значит, что они в аквариуме.

Другие находящиеся в комнате могут наблюдать за ними и слушать, но не говорить. Они – вне аквариума.

Если кто-то находящийся вне аквариума хочет принять участие, он может «нырнуть». Но когда один человек заходит извне, кто-то изнутри должен «вынырнуть».

Таким образом обсуждение остается компактным и продуктивным, все информированы и включены в процесс. Кроме того, это хороший способ обучения: новички включаются в процесс, не замедляя его скорости.

## Важность компактности

Помните обсуждение торта и капкейков из главы 10? Сейчас как раз настало время разбить наши торты на капкейки наименьшего размера из возможных. Именно сейчас, когда присутствуют разработчики, тестировщики и другие специалисты, непосредственно участвующие в создании программного продукта, можно по-настоящему взяться за разделение истории на части.

Надеюсь, вы знаете, что корень слова software (программное обеспечение), soft, означает «мягкий». Это значение не ограничивается тем, что мы подразумеваем, говоря о губках или булочках. Лучше подойдет ассоциация с большим документом или книгой. Если бы вы писали книгу, как стараюсь делать сейчас я, вы бы не пытались сделать все за один раз. Вы бы написали, допустим, главу в один присест. Я, например, пишу за раз одну главу, а Питер, опытный и доброжелательный редактор, с которым я сотрудничаю, просматривает ее, делая необходимые предложения и поправки.

Но это не значит, что глава готова. До этого еще далеко.

Мне нужно вернуться к ней снова и подумать, где в тексте должны быть иллюстрации. Я должен решить, стоит ли добавить сноски, ссылки, объяснения терминов из глоссария или элементы указателя. Затем другие редакторы издательства проходятся по всем главам, внося финальные правки. Получается, что я разделяю работу, делая ее итерационно, и таким образом вижу, как книга постепенно обретает форму.

Сейчас вы читаете главу «Огранка, полировка, разработка». Если так, то, видимо, это значит, что мой тортик успешно испекся. Если бы я решил позаботиться о финальных критериях приемки, то, наверное, сформулировал бы такие.

- Материал понятен мне и лично проверен мной.
- Материал понятен моим редакторам и отредактирован ими.
- Книга снабжена иллюстрациями, которые облегчат читателям понимание материала и визуализируют ключевые моменты.
- Книга снабжена предметным указателем, с помощью которого читатели легко могут найти объяснение термина в соответствующей главе.
- Книга снабжена глоссарием, который читатели могут использовать, чтобы уточнить определения терминов, встречающихся в данной главе.

Это немаленькое количество работы. Даже сейчас, печатая первый, черновой вариант критериев, я понимаю, что предстоит сделать еще очень много. Но я не собираюсь закончить все это до перехода к следующей главе, потому что мне хотелось бы сначала увидеть книгу целиком. Поэтому я разбиваю работу на капкейки – маленькие законченные фрагменты, не готовые к публикациям, но поддерживающие мою уверенность в том, что я двигаюсь в правильном направлении, работая над книгой.

Я разделил бы работу над книгой на следующие истории.

- Огранка, полировка и создание первого чернового текста.
- Огранка, полировка и создание второго чернового текста.
- Огранка, полировка и создание текста с иллюстрациями.
- Огранка, полировка и создание текста с учтенными замечаниями и поправками.
- Огранка, полировка и создание текста с предметным указателем.
- Огранка, полировка и создание текста с глоссарием.

• Огранка, полировка и создание окончательной версии текста.

Для каждого из этих этапов я могу создать полноценную историю, в которой описано, что представляет собой этап, а также перечислены шаги, которые я (с помощью Питера в части редактирования) должен проделать, чтобы получить в конце этапа текст со всеми запланированными улучшениями и поправками. Как вы можете видеть, по мере завершения каждого этапа глава становится все более проработанной и приближается к состоянию готовности. Теоретически вы вполне могли бы прочитать и даже найти небесполезной первую версию, которая появилась после завершения первого этапа. Но я не хочу ее никому показывать, и вряд ли она бы вам очень понравилась.

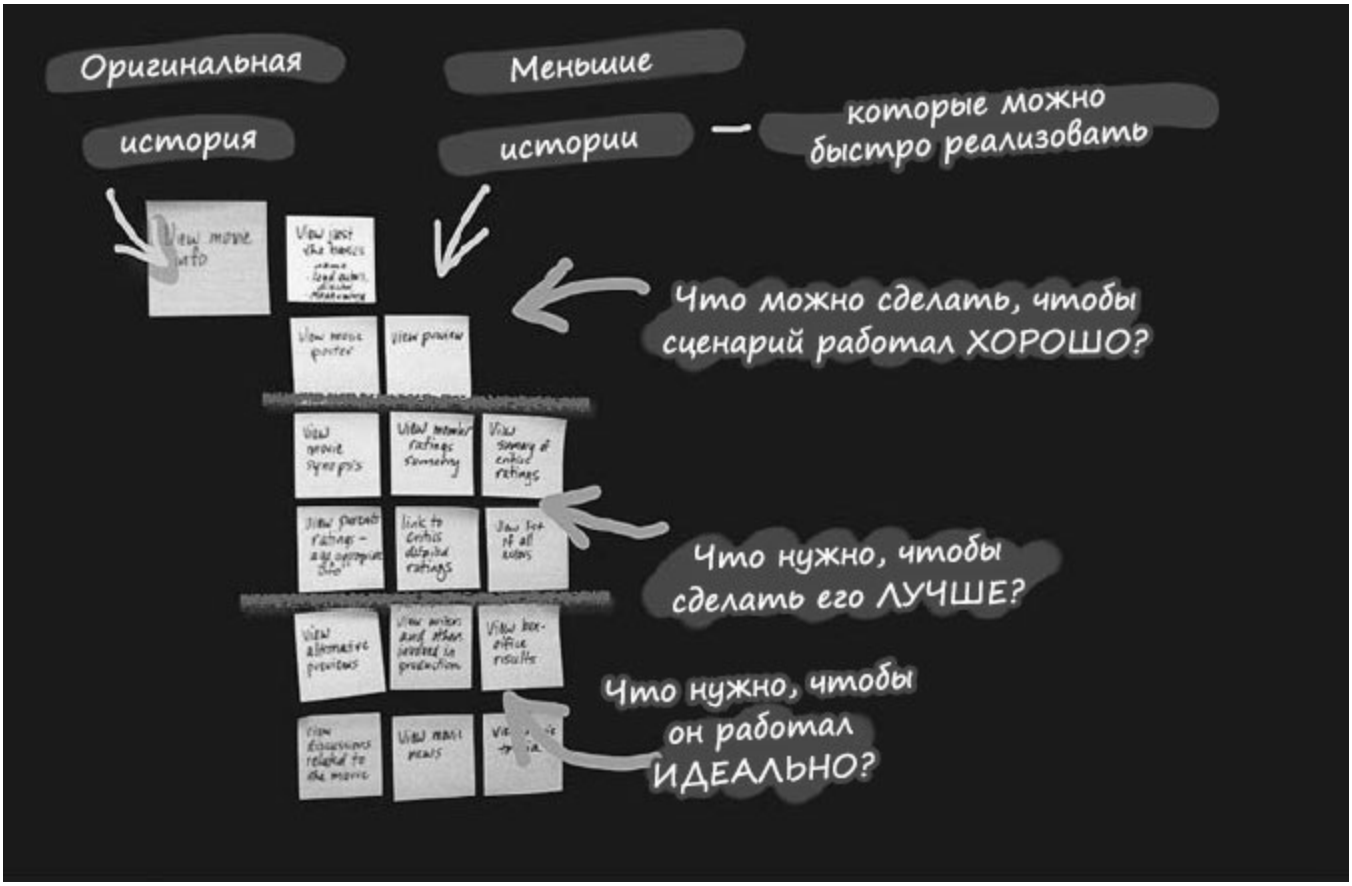
Поскольку я уверен, что вы хорошо знаете свое дело, думаю, вы заметили, что этот список маленьких компактных историй во многом очень похож на критерии приемки для этой главы. В этом и заключается суть. Именно обсуждение критериев приемки должно подсказать нам, на какие меньшие части лучше всего разбить работу, чтобы их удобно было создавать и контролировать по ходу дела.

Очень важно контролировать работу постоянно, чтобы можно было оценить ее и при необходимости скорректировать курс. Вот здесь, например, вы могли бы прочитать очень неудачный пример, который я изначально вставил в это место. Но этого не произойдет, потому что, написав текст, я позже перечитал его и удалил этот пример.

При традиционном подходе к разработке программного обеспечения работу по контролю и удалению неудачных фрагментов непременно объяснили бы плохими требованиями. Но после того, как вы надели плащ и шляпу рыцаря Agile, эта работа становится обучением и итеративным улучшением.

### *Игра «Хорошо – лучше – идеально»*

Одна из моих любимых и очень простых техник окончательного разбиения историй – игра «Хорошо – лучше – идеально». Чтобы играть в нее, нужны большая история и пачка стикеров. Результат игры выглядит примерно так.




**Довольно хорошо… пока**

Как только у вас появится история, обсудите для начала, что достаточно сделать, чтобы она просто хорошо работала. Пока не стоит думать о том, как вызвать у клиентов восхищение, хватит минимально рабочего состояния. Запишите характеристики, которые обеспечат состояние «достаточно хорошо», и в дальнейшем относитесь к ним как к отдельным небольшим историям.

Если рассмотреть в качестве примера, скажем, IMDb.com (the Internet movie database – интернет-база данных кинофильмов), мы можем обсудить историю «Просмотреть информацию о фильме». Представим себе экран, где можно увидеть разные подробности о фильме и, таким образом, принять решение, стоит ли его смотреть. Обсуждая этап «довольно хорошо», можно ограничиться следующими простыми вещами.

• Просмотреть базовую информацию: название, рейтинг, режиссер, жанр и т. д.

• Просмотреть постер фильма.

• Просмотреть трейлер.

**Лучше**

Затем спросим себя, что может сделать историю лучше. Продолжая рассматривать пример с кинофильмами, можно добавить такие пункты.

• Прочитать аннотацию к фильму.

• Прочитать рейтинги участников.

• Прочитать рейтинги в обзорах.

• Просмотреть список всех актеров фильма.

**Идеально**

Наконец, подумайте, что может сделать историю потрясающей. На этом этапе не бойтесь самых безумных идей. Помните: вы не пишете требования. Это просто варианты, которые будут рассмотрены вами совместно с командой. Зачастую в таких дискуссиях зарождаются самые блестящие идеи – то, что может сделать ваш продукт выделяющимся, но оказывается удивительно недорогим в реализации. Продолжая работу с примером, можно добавить следующее.

- Посмотреть альтернативные обзоры или видео об этом фильме.
- Почитать любопытные факты о фильме.
- Почитать новости о фильме.
- Почитать дискуссии об этом фильме и принять в них участие.

Как видите, прогрессия небольших историй может помочь развить историю «Просмотреть информацию о фильме» от простого состояния, когда мы можем лишь убедиться в его работоспособности, до возможностей, которые сделают эту историю поистине впечатляющей. Если бы я занимался этой функциональностью, то создал бы базовые вещи всего приложения, прежде чем двинуться к стадиям улучшения и доведения до идеала. Работая таким образом, я куда безопаснее чувствую себя, приближаясь к дедлайнам.

Когда обсуждения историй станут по-настоящему хорошими, а я знаю, что скоро так и будет, в конце *семинара по историям* вы должны получить набор историй верного размера, снабженный множеством дополнительной документации и различных моделей, а также критериями приемки, где будет описано, как вы собираетесь убедиться в том, что работа над историей успешно завершена. Иногда требуется несколько семинаров, дополненных несколькими внешними исследованиями, анализом данных и дизайнерской работой, прежде чем удастся прийти к соглашению, но это нормально. Огранка и полировка всегда требуют времени и терпения.

### *Рецепт планирования цикла разработки*

В процессах Agile, таких как Scrum или экстремальное программирование, используется принцип *ограничения времени разработки*. Это значит, что каждый цикл разработки начинается с сессии планирования, а заканчивается оценкой сделанного. Во многих компаниях эти два вида совещаний – самые ненавистные из всех. Чаще всего они длинные и очень утомительные, и к тому времени, когда члены команды могут покинуть конференц-зал, они готовы согласиться с чем угодно, лишь бы выбраться на свободу. Очевидно, что качество составленных планов при таком подходе невысоко.

Но этот подход – не единственный из возможных.

Вот простой рецепт, который поможет вам избежать самых распространенных проблем.

**Подготовка**

*Выберите истории, над которыми будете работать следующие один-два цикла*. Если вы владелец продукта, периодически уточняйте у членов ключевой команды, работающей над продуктом, состояние функциональностей, находящихся

в данное время в разработке. Выберите для добавления те истории, которые помогут приблизить эту работу к релизу.

*Проведите предварительный семинар*. Выделите время, чтобы члены ключевой команды продукта могли посовещаться между собой перед сессией планирования. Погрузитесь в детали, разделите на части большие истории, предусмотрите несколько вариантов. Перечитайте историю Мэта Кроппера из главы 7. Когда я общался с Мэтом, он возлагал самые большие надежды на серии коротких получасовых спонтанных семинаров по историям, на которых программисты и тестировщики готовились к планированию.

*Пригласите всю команду*, а также других людей, чья помощь может вам потребоваться в предстоящем цикле разработки.

**Планирование**

*Начните с обсуждения стратегической цели* предстоящего цикла. Вы выбрали несколько историй, над которыми собираетесь работать. Как именно эта группа историй будет способствовать прогрессу решения, которое вы планируете выпустить?

*Просмотрите истории, которые планируете обсудить*. Не следует слишком углубляться в детали – просто хорошо передайте общую картину. Вернитесь к истории Николы и Стива, рассказанной в этой главе. Посмотрите на фотографию, где Никола стоит на фоне стены: на ней очень много слов и картинок, с помощью которых члены команды легко могут представить себе общую картину. Очень разумный подход.

*Дайте командам время на самостоятельное планирование*. Помните: толпа не может сотрудничать. А специалистам, которые будут разрабатывать и тестировать программный продукт, необходимо продумать собственные рецепты создания историй – точно так же, как делала Сидни из главы 10. Дайте командам примерно час на то, чтобы они разбились на небольшие группы и хорошенько обдумали свои истории. Если вы владелец продукта, UI-дизайнер или бизнес-аналитик, оставайтесь где-то поблизости. Наблюдайте, если хотите. Будьте готовы ответить на вопросы, которые помогут им продвигаться быстрее.

*Работая в маленькой группе, создайте план для каждой истории*. Помните трех амиго, о которых мы говорили в главе 12? Возьмите их за образец при формировании групп. Затем как команда разработки решите, сколько историй могут быть успешно завершены в данном цикле разработки. Не забудьте принять во внимание праздники и отпуска. Однажды мне рассказывали историю, как замечательный план разработки провалился из-за Дня благодарения – как будто никто не знал о том, что приближается этот праздник!

*План должны утвердить все*. В конце выделенного промежутка времени, после того как команда составит свой план для каждой истории, нужно снова собраться вместе и поделиться своим планом с остальными. Не стоит подробно рассказывать обо всех деталях, это не слишком интересно даже вашим коллегам. Что им интересно, так это какие функциональности будут созданы вашей командой в конце цикла. Команда должна отнестись к плану и соглашениям очень

серьезно, если хочет иметь репутацию надежной и ответственной.

*Чтобы достичь соглашения, может потребоваться некоторое время*, особенно если вся работа, которую нужно сделать, не умещается в заданное время разработки. К счастью, вы знаете несколько приемов разбиения историй, описанных ранее. Попробуйте также рассмотреть реализацию истории в состоянии «хорошо», а не «лучше». Это должно помочь уместить в итерацию или спринт все необходимое.

*Отдыхаем!* Все готово. В прошлом мы любили выполнять планирование днем, чтобы закончить этот процесс за какое-то время до конца рабочего дня, а затем отмечали это событие, расходясь по домам пораньше. На следующий день все приходили отдохнувшими и готовыми начать работу по плану, составленному накануне.

# Используйте карту историй во время разработки

Используйте карту, чтобы у членов команды разработки выработалось одинаковое понимание. Я часто слышу от участников различных команд, работающих по процессу Agile, как сильно они любят помогать друг другу и какими продуктивными чувствуют себя, потому что каждую неделю или две видят и демонстрируют работающие программные продукты. Но потом они всегда говорят нечто вроде: «Я, кажется, утратил понимание общей картины. Все, что я вижу, – это маленькие части продукта, который мы создаем». Используйте карту, чтобы облегчить команде понимание продукта в целом или функциональности, над которой вы работаете. Нахождение тактических проектных решений и выполнение задач разработки будет куда эффективнее, если люди хорошо понимают контекст, в который должна встроиться их работа.

# Используйте карту для визуализации прогресса

Начав разрабатывать свой продукт к предстоящему релизу, активно пользуйтесь картой: она представляет собой отличную визуальную панель, на которой хорошо видно, что вы уже построили, а что – еще нет.

Одни команды предпочитают убирать детализированные истории с карты после того, как работа над ними завершена. Таким образом, лишь взглянув на карту, можно узнать, что еще осталось сделать.

Другие, не изменяя структуру карты, используют фломастеры или цветные стикеры, чтобы помечать законченные истории. Отойдя назад и взглянув на карту, они легко отделяют то, что уже сделано, от того, что осталось сделать.


Используйте карту, чтобы определить следующие истории для разработки. Каждую неделю владельцы продукта должны оценить, насколько продвинулась работа, а затем принять решение, за что нужно взяться в следующую очередь. Когда карта визуализирует прогресс, ее намного проще просканировать взглядом и выявить области, которые требуют больше внимания. Примерно так работают художники. Отступив на шаг назад и окинув взглядом картину, они могут определить, в каком месте требуется нанести пару мазков.

# Используйте простые карты во время семинаров по историям

В каждом цикле разработки вы определяете в карте истории, которые должны пойти в работу следующими. Вы приносите эти истории на заключительные, самые продуктивные обсуждения, проходящие во время семинара по историям.

Во время семинара вы можете применить визуализацию, построив самую простую карту. Возможно, она будет состоять всего из трех-четырех шагов, которые делает пользователь, работая с обсуждаемой сейчас функциональностью. Но размещение этих нескольких шагов в виде стикеров на стене наглядно проиллюстрирует последовательность и сделает дискуссию живее. Начав обсуждать критерии приемки, напишите их на стикерах и добавьте на эту мини-карту. В конце вы получите очень простую визуализацию, которая поддержит обсуждение, состоявшееся на семинаре.

## *Визуализация рабочего бэклога*

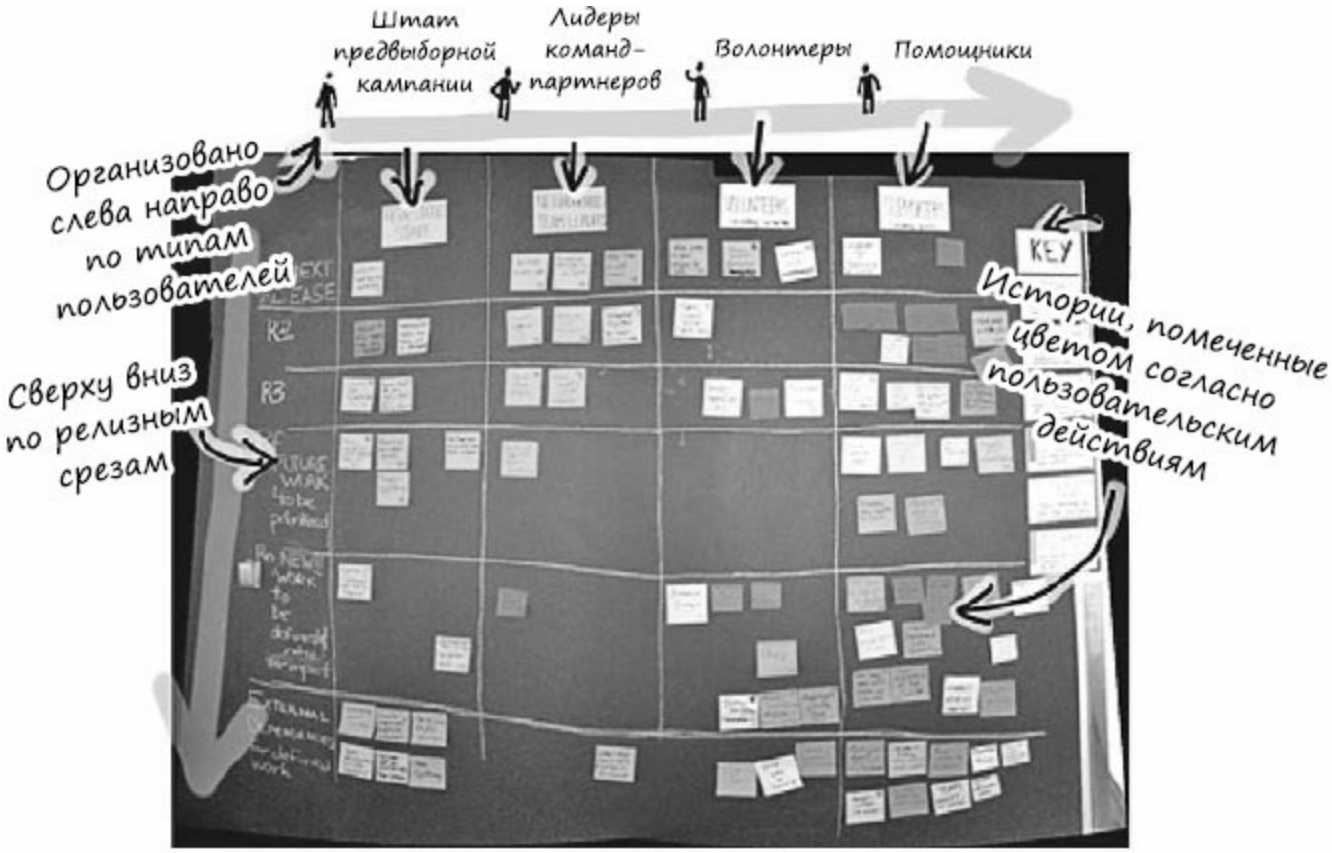
*Крис Гансен и Джейсон Канеш*

Интернет навсегда изменил политику, что доказала избирательная кампания президента Обамы в 2008 году. Онлайн-стратегия Барака Обамы сыграла непосредственную и значительную роль в его продвижении и последующем избрании. Поэтому стратегия выборов 2012 года включала в себя обеспечение сервисов, которые поддерживали массовый сбор средств и традиционное движение снизу, при использовании технологий в качестве коэффициента силы, чтобы компенсировать огромное количество денег со стороны, вращающихся в предвыборной гонке. Мы использовали специальные инструменты, такие как Pivotal и Basecamp, для отслеживания своей работы по созданию личного кабинета кампании Обамы 2012 года, а для того, чтобы все остальные могли легко понять, что происходит, мы использовали стену, которую вскоре полностью заклеили стикерами. Вы можете удивиться, зачем мы тратили время на возню с этими дурацкими стикерами? А вот почему.

Так вышло, что в одном пространстве и над одной целью работали две совершенно разные культуры. Мы сидели в углу, выкрутив лампочку, чтобы работать в относительной темноте. Мы притащили собственные шумные клавиатуры. Мы носили гигантские наушники, чтобы заглушить шум новостей и интервью по кабельному телевидению, звонки и аплодисменты, которые проникали в пространство нашего офиса. Мы даже носили футболки метал-групп вместо официальных костюмов. Все люди, работавшие в предыдущих кампаниях, все эти ребята в наглухо застегнутых костюмах с галстуками имели традиционные взгляды на разработку программного обеспечения. «Мы описываем функции, которые нам нужны, а вы их для нас разрабатываете. Срок – вчера». Они не привыкли к постепенному предъявлению функций и итеративным изменениям или улучшениям, и они уж точно не будут искать оптимальные соотношения, чтобы получить нужные функции в тот момент, когда они могут дать наибольший эффект. Это предвыборная кампания! Нельзя сместить дату представления продукта, разве

что указом конгресса! На следующий день после выборов все заканчивается независимо от того, успели мы или нет. Поэтому о том, чтобы они получили абсолютно все, чего хотели, и речи не шло.


Президент Обама в костюме и Джейсон в футболке:
самая стрессовая демонстрация в истории




Первая доска

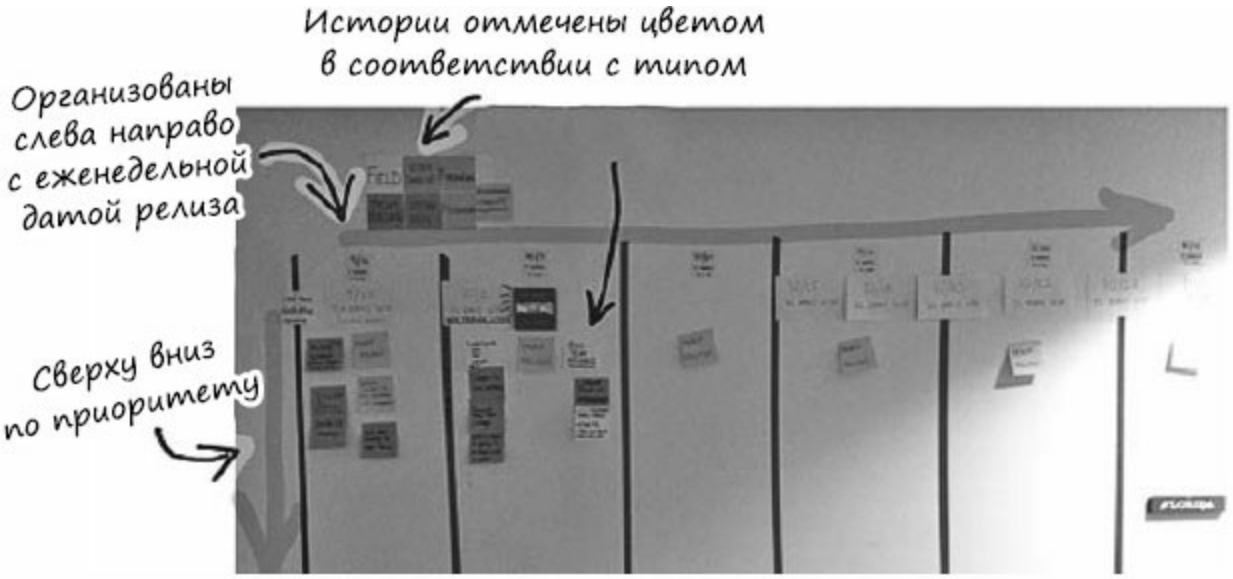
Приступив к работе, мы использовали некий базовый подход к составлению карты, чтобы описать людей, использующих нашу систему, и самые различные их потребности. Мы могли затем организовать работу над релизами, разделив их по времени. Это вполне сработало бы, но для людей, проводящих кампанию, было бы сложно оставаться вовлеченными в процесс, пытаясь понять способы работы с системой людей, которые будут использовать ее позднее, более чем через год от сегодняшнего дня. Их головы были заняты тем, что они хотели видеть готовым *сегодня*. Они строили множество догадок о том, что волонтеры и группы лидеров будут делать. Это было неизбежно, особенно если мы меняли свои представления о том, как все это будет работать.

Через семь месяцев мы выпустили маленький, минимально жизнеспособный продукт – только то, чего достаточно было для работы наших пользователей в штате Айова. Именно тогда все изменилось. Первая реакция была резко отрицательной, так как мы выпустили совершенно не то, что представляли себе люди. Было пропущено немало важных вещей. Попадались баги. Нас забросали вопросами: «Почему вы не сделали все как надо с первого раза?» И хотя мы уверяли, что будем вносить улучшения и исправлять неполадки каждую неделю, никто не собирался нам верить до того, как увидит это своими глазами. Но по мере того, как мы воплощали свои обещания в жизнь, росло и доверие.

Именно здесь сыграла важную роль стена со стикерами. Мы использовали Pivotal Tracker и Basecamp, но все остальные люди не хотели пользоваться этими инструментами. Нужен был абсолютно прозрачный способ показать всем, над чем мы работаем и к чему вскоре придем. Наша стена с историями была организована слева направо по календарным неделям, а сверху вниз – по приоритету. Самым важным для нас было время. На стене висели огромные часы, показывавшие, сколько дней, часов, минут и секунд осталось до дня выборов. Все понимали, что некоторые элементы сейчас не очень важны, а некоторые станут более критичными по мере приближения дня выборов. Каждая идея на стене была отмечена своим цветом, соответствующим определенному виду деятельности, связанному с этой идеей, например работе с избирателями, построению команды, регистрации голосов и явке избирателей. На этой картинке много фиолетового потому, что это ранний этап работы и построение команды, соответствующее фиолетовому цвету, сейчас более важно, чем другие моменты, например явка избирателей.

Обычно элементы на доску мы добавляли совместно с людьми, управлявшими кампанией. Мы говорили о том, что уже было запланировано на эту неделю и какие у нас шансы закончить эту работу в течение недели. Мы говорили о реальных проблемах, от которых избавит эта функциональность. Вместе мы решали, насколько эта функциональность важна по сравнению с другими, уже имеющимися на стене. Когда приходило время начинать разработку, программист, ответственный за данную функциональность, работал напрямую с тем из наших партнеров, кто знал ее лучше всего. Они вместе продумывали все детали, что порой занимало довольно много времени. Иногда мы выделяли день на то, чтобы создать простые прототипы графического интерфейса.


Обратный отсчет




Еженедельные релизы слева направо, сверху вниз по приоритету, отмеченные цветом в соответствии с областью продукта

Эта огромная стена со стикерами была ключевым моментом в создании моста между теми, кто разрабатывал программное обеспечение, и теми, кто должен был им пользоваться. Именно это им было нужно, чтобы визуализировать, что, как и когда происходит, а также активно участвовать в принятии решений.

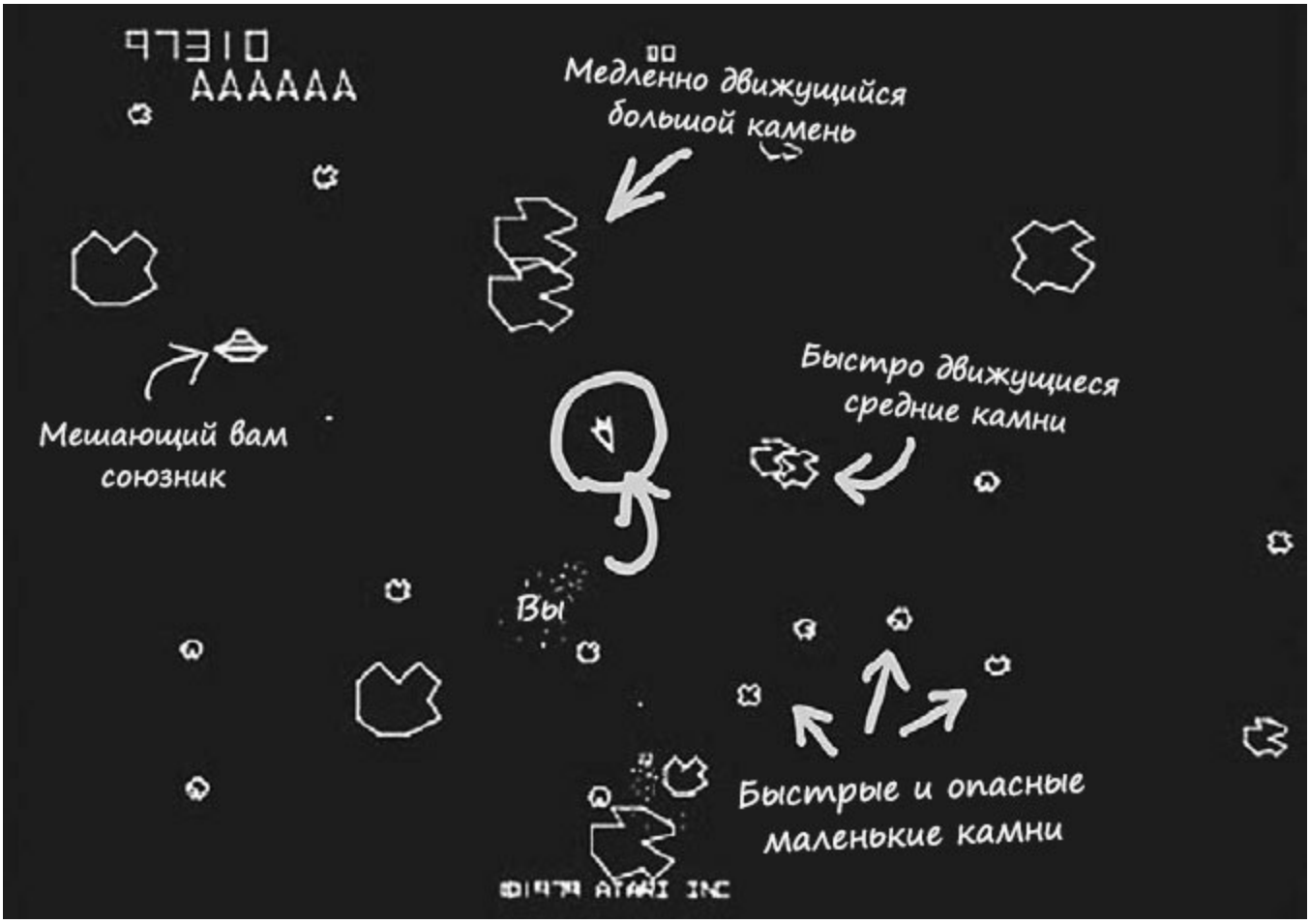
# Глава 17. Истории – это нечто вроде астероидов

Если вы примерно одного возраста со мной, то, может быть, вспомните (с ностальгией), как играли в одну из первых видеоигр под названием «*Астероиды*». Давайте на минутку предадимся воспоминаниям: обещаю, это вполне в рамках темы.

В игре «Астероиды» вы играли за небольшой космический корабль, летящий где-то далеко в космическом пространстве. Но дорогу вам преграждал пояс огромных астероидов, и нужно было расстреливать их, чтобы расчистить себе путь и остаться в живых! Если вы стреляли в большой астероид, он распадался на несколько маленьких. Еще интереснее игру делало то, что эти маленькие астероиды двигались быстрее и в разных направлениях, что значительно усложняло вашу задачу уклониться от столкновений с ними. Если вы стреляли в один из этих меньших астероидов, он распадался на еще меньшие, которые двигались еще быстрее в разных направлениях. Очень скоро экран был весь в астероидах разных размеров, двигавшихся во всевозможных направлениях. К счастью, если вы стреляли в самые маленькие астероиды, они просто взрывались, что помогало немного расчистить путь.

Самой худшей стратегией в этой игре было стрелять по большим камням и разбивать их на маленькие. Экран очень скоро заполнялся маленькими камнями, летящими во все стороны, после чего вас настигала быстрая, но мучительная смерть.

Очень плохая стратегия управления бэклогом – разбить все большие истории на части так, чтобы все они входили в ближайший цикл разработки. Ваш бэклог заполнится множеством маленьких, хаотично летящих историй, которые очень быстро вас укокошат. Конечно, на самом деле вы не умрете, но окажетесь погребены заживо под тяжестью абсолютно ненужной сложности. Вы, как и все остальные, будете жаловаться на то, что перестали видеть общую картину, скрытую всеми этими мелкими деталями.




Разбивайте истории на части постепенно и лишь тогда, когда это необходимо.

Во всех обсуждениях историй, на всех стадиях разбиения работайте, помня следующее.

1. Обсуждая *возможности*, рассмотрите, для кого они предназначены, какие проблемы решают и хорошо ли сочетаются с вашей бизнес-стратегией. Есть смысл на этом этапе разделить слишком объемные возможности.

2. Во время *исследования* обсуждайте более конкретно, кто, зачем и как использует продукт. Цель вашей команды – представить продукт ценный, полезный и реализуемый. Большая часть дробления камней происходит здесь. Лучше всего, если вы передвинете в релизный бэклог лишь небольшую часть историй, которая позволит вам выпустить минимально жизнеспособный продукт.

3. Планируя *стратегию разработки*, обязательно обсуждайте риски, вызванные предположениями о том, что понравится пользователям и закрепится в их работе, а также риски, связанные с технической реализацией. Разбивайте камни, нацеливаясь на изучение, создавайте то, что вам нужно в первую очередь, чтобы узнать как можно больше.

4. Планируя *следующий цикл разработки*, проведите финальные, самые качественные обсуждения, чтобы точно решить, что вы создаете, и прийти к соглашениям, как будете проверять готовность программного продукта. Каждое из этих соглашений позволяет разбить историю на еще более мелкие истории так, чтобы каждый из них соответствовал хотя бы одному соглашению.

Очевидно, что если вы попытаетесь провести все четыре обсуждения за одну встречу, это займет очень много времени и будет слишком утомительно. Для того чтобы рассмотреть все необходимые аспекты, потребуется очень много людей. А из моих предыдущих предупреждений и из собственного опыта вы наверняка знаете, что большая группа людей не

может работать эффективно, по крайней мере не все присутствующие в одной комнате одновременно. Именно поэтому мы разбиваем истории постепенно, проводя довольно много отдельных обсуждений.

# Обратная сборка разбитых камней

В игре «Астероиды» нужна была большая осторожность, чтобы случайно не расстрелять не те астероиды, которые нужно, потому что, однажды разбив, вы не сможете собрать их вновь. А вот однажды разделенные истории собрать заново *можно*.

Чтобы избежать переполнения бэклога множеством маленьких историй, возьмите группу историй, объединенных одной темой, и запишите все их названия на одной карточке в виде маркированного списка. Дайте им одно общее название и озаглавьте так свою новую карточку. Вуаля – у вас снова есть одна большая история.

Если вдуматься, это просто фантастика. Карточка с написанным на ней названием делает осязаемой и управляемой любую витающую в воздухе идею. Идеи гораздо более податливы, чем камни или горы документов. Иногда мы забываем, что другое название разработки программного обеспечения – труд, основанный на знаниях. Если мы в самом деле забываем это и слишком увлекаемся документами и процессами, наша работа становится чем-то скучным и бюрократичным. А когда я общаюсь с людьми, управляющими огромными бэклогами, сплошь заполненными маленькими историями, то чувствую, что они просто задыхаются от бюрократии.

### *Объединяйте маленькие истории, чтобы очистить бэклог*

Я часто работаю с командами разработки продуктов, бэклоги которых заполнены сотнями элементов. И разумеется, в этом случае чаще всего звучит жалоба на большие сложности с расстановкой приоритетов. Когда я заглядываю в их бэклоги, то часто вижу, что они переполнены множеством маленьких историй. Обсуждение каждого из них и вынесение решения о приоритете может занять долгие часы, а то и дни. Но необходимости в этом нет.

Если бы речь шла об игре «Астероиды», вы бы уже проиграли. Но поскольку это не так, попробуйте просто объединить маленькие истории в несколько больших.

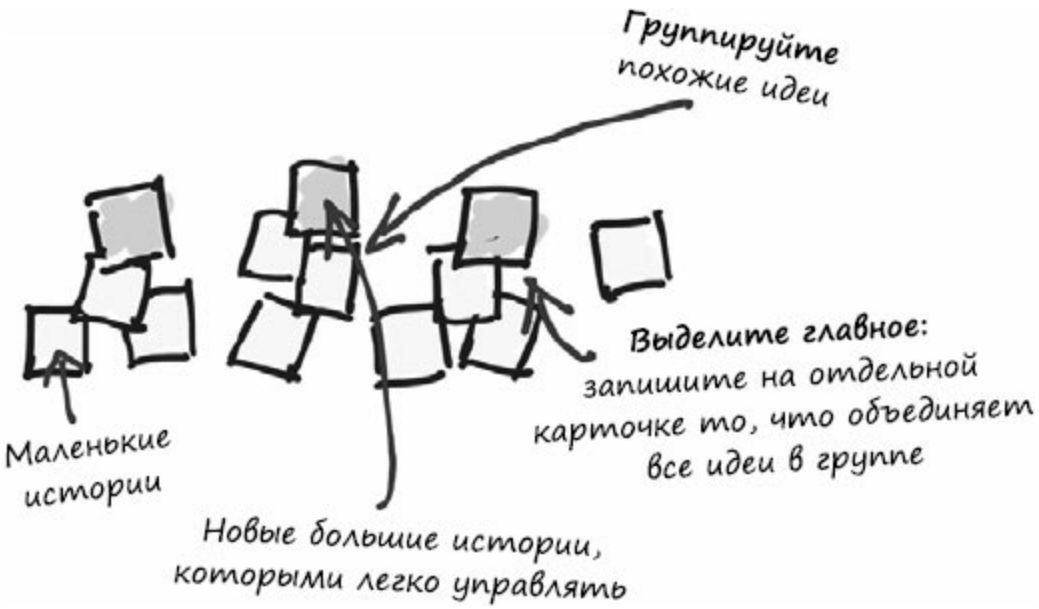
1. Если ваши истории собраны в электронный бэклог, перенесите их на карточки или стикеры. Какой бы инструмент вы ни использовали, там наверняка есть функция печати или экспорта компактного списка. Я использую функцию почтовой рассылки в текстовом редакторе, чтобы создать этикетки для всех историй и затем или наклеить их на карточки, или просто напечатать прямо на карточках.
2. Попросите членов команды, которые понимают систему, помочь вам. Запланируйте совещание в комнате, где много места на стенах или столах.
3. Дайте каждому набор карточек и попросите разложить их на столе или расклеить на стенах.
4. Увидев карточку, которая выглядит похожей на ту, что у вас уже есть, разместите их рядом. Пока не слишком задумывайтесь над тем, что значит похожа, – просто следуйте своей интуиции.
5. Эту работу следует выполнять молча, по крайней мере поначалу. Как вы

непременно убедитесь, обсуждение скорее замедляет процесс. Кроме того, использование модели и языка жестов для коммуникации – отличное упражнение для развития.

6. Передвигайте и меняйте местами любые карточки, какие хотите. Моделью управляют все, никто не может решать единолично, где должна находиться какая-либо карточка. Если что-то, как вы считаете, находится не на своем месте, переместите это. Если кто-то не согласен с вами, он (-а) может переместить карточку обратно. Это сигнал к началу дискуссии.

7. После того как все карточки оказались на своих местах, возьмите стикеры или карточки другого цвета и создайте заголовок для каждой группы. Напишите на этих карточках имя общей истории – то, что объединяет все карточки в группе. Название «Улучшения графического интерфейса» может быть слишком общим – лучше что-то вроде «Улучшить добавление и редактирование комментариев», так становится ясно, что все улучшения касаются комментариев.

8. Таким образом у вас получатся новые большие истории. Остальные карточки составят основу для маркированного списка в их описании. Теперь можно занести получившееся обратно в бэклог. Или, если вся эта работа не срочная, занесите ее в бэклог возможностей.




Этот способ замечательно работает как для огромных бэклогов, переполненных множеством маленьких элементов, так и для больших скоплений багов. Вы знаете, всегда есть множество низкоприоритетных багов, которые никто никогда не исправляет. Объедините их в группы и заведите заново как баги определенной области системы с более высоким приоритетом. Когда разработчик доберется до исправлений багов высокого приоритета в этой области, заодно исправит и низкоприоритетные. Ваши заказчики и пользователи будут вам благодарны.

# Не перестарайтесь, составляя карты

Я часто слышу от людей, пытающихся освоить методику работы с картами историй, что там слишком много работы. Задав вопрос, что же показалось им таким сложным, я, как правило, слышу трагическую повесть о попытке создания огромной карты всей их системы для обсуждения небольшой простой функциональности. Они правы – это слишком. Не следуйте их примеру.

Составьте карту только того, что вам нужно для обсуждения вашей функциональности.

Например, я работал с компанией, которая хотела немного изменить функцию комментирования в их программном продукте для совместного редактирования документов. Сначала команда составила высокоуровневую карту процесса редактирования документов, использовав для этого лишь несколько карточек. Затем, подойдя к области комментирования, они добавили еще немного карточек, которые кратко представляли, что их продукт может делать сейчас, с помощью множества записей на каждой из них. После этого началось обсуждение изменений, которые они хотели внести, добавляя уже гораздо больше карточек для всех деталей и вариантов, которые принимались во внимание.

Добавляя функциональность в существующий продукт, составляйте карту необходимой области, начиная с того момента, где функциональность начинается в пользовательской истории, и заканчивая чуть позже того места, где она кончается. Нет нужды строить карту всего вашего продукта.

Помните, что карты историй должны поддерживать обсуждение ваших пользователей и идей продукта. Хорошее правило выборки: если вам не нужно что-то обсуждать, то и карта для этого тоже не нужна.

# Не волнуйтесь по пустякам

Я описал весь процесс разбиения камней от начала до конца и даже предостерег вас от преждевременной обработки этих камней по аналогии со старой игрой Atari, так что вы не будете пытаться разбить их слишком рано. Но говоря обо всех этих стратегиях, мы исходили из того, что все новые истории, которые мы рассматриваем, – крупные. На самом деле во многих случаях это не так. После того как вы предъявили новый продукт или функциональность пользователям, вы неизбежно обнаружите множество очевидных мелких недоделок – всякой ерунды, о которой следовало подумать до предъявления, но, к сожалению, вышло не так. Во всяком случае, у меня это происходит постоянно. Для таких вещей я не провожу обсуждение возможностей и не собираю группу для исследования продукта, потому что необходимость этих поправок очевидна всем. Поэтому эти недоделки отправляются прямиком в бэклог ближайшего релиза и обрабатываются членами команды на ближайшем семинаре, в результате чего в продукт быстро вносятся исправления. Тот же самый процесс существует для багов и множества маленьких улучшений.

***Спасая мир, идите маленькими шажками***

Это снова мой друг Шериф из Atlassian. На этой фотографии он объясняет мне, что члены команды разработки продукта постоянно собирают и обрабатывают множество мелких улучшений. Они очень волнуются за своих пользователей. Они знают, что стаи мелких багов и неполадок бесят пользователей, и этот факт бесит саму команду. По их словам, это примерно как умереть, «тысячу раз поранившись бумажным листом». Поэтому на стене, рядом с которой команда работает над продуктом Green Hopper, много сгруппированных пометок. Каждый раз, когда какой-нибудь член команды исправляет какой-то мелкий недостаток, он или она делает пометку на стене. Похоже, что в следующем релизе будут исправлены 47 мелких неполадок. Если вы пользуетесь JIRA или Confluence, чуть позже вы можете сказать ребятам спасибо.

# Глава 18. Извлекайте уроки из всех своих разработок

Если вы приверженец традиционного подхода к разработке программного обеспечения, то, наверное, думаете, что работа закончена, когда приложение готово. Но разработка Agile и истории предназначены для обучения. Перед тем как приступить к написанию кода, мы тратим много времени на то, чтобы убедиться, стоит ли вообще что-то разрабатывать, и если да, то что именно. А после того, как работа закончена, мы оцениваем полученный результат и спрашиваем себя, стоило ли это создавать и насколько хорошо у нас вышло.

Давайте поговорим о возможностях для обучения, которые предоставляются нам после окончания разработки.

# Оцените свою командную работу

Настало время отпраздновать победу. В конце цикла разработки и тестирования должна состояться вечеринка. Вы рассмотрели несколько идей, провели множество обсуждений, нарисовали множество эскизов, что в конце концов воплотилось в работающем (к счастью) программном продукте. С использованием традиционного подхода к написанию требований времени понадобилось бы намного больше, а вы и ваша команда не ощущали бы своей вовлеченности в процесс.

После того как вы все поздравили друг друга, всем вместе нужно сесть и честно оценить то, чего удалось достичь. Если вы честны с самими собой, то, скорее всего, найдете кое-что, что стоило бы сделать иначе. Нужно для каждой такой шероховатости написать отдельную историю и добавить ее в релизный бэклог. Затем надо решить, нужно ли вносить эти изменения немедленно или можно отложить их на какое-то время и обработать в порядке очередности.

В процессах наподобие Scrum такая процедура называется *оценкой спринта*. Если вы практикуете Scrum, то, наверное, слышали рекомендацию приглашать на это мероприятие всех и каждого, но я собираюсь предложить вам нечто иное.

Команде, которая тесно сотрудничала, проводила самые детальные обсуждения, приходила к соглашениям о сути создаваемого продукта и работала над его реализацией, необходимо время и спокойное место, чтобы открыто обсудить свою работу. Мнение людей, не входящих в команду, включая руководство организации, очень важно, и, конечно, команда должна его услышать. Но все-таки эти люди не присутствовали на обсуждениях, где было выработано одинаковое понимание деталей создаваемого продукта. Они не участвовали в обсуждениях, во время которых составлялся детальный план разработки. И наконец, они не работали вместе с командой над воплощением всех этих соглашений и дискуссий в работающее программное приложение. Итак, сначала мы должны оценить, удалось ли нам построить то, что мы себе представляли, с тем уровнем качества, на который мы надеялись. Для этого и нужны такие мероприятия, как *командное осмысление и оценка продукта*.

### *Рецепт командного осмысления и оценки продукта*

Команда, которая совместно работала над тем, чтобы понять истории, а также над составлением краткосрочных планов по их разработке, должна остановиться и задуматься о качестве своей работы. Используйте для этого небольшой семинар.


*Пригласите на этот семинар только тех*, кто непосредственно работал над изучением объема работы и ее планированием. Включите в число участников владельца продукта и любых других участников команды разработки, например программистов, тестировщиков и всех тех, кто активно участвовал в реализации. Да, участие партнеров по бизнесу не обязательно. Скоро они подключатся к обсуждению, но пока нам нужно безопасное пространство для общения.

*Запаситесь закуской*. Несколько лет назад такие семинары в моей команде просто не могли начаться, пока кто-нибудь не приносил бублики.

На семинаре необходимо уделить внимание *трем темам: продукту, плану и процессу*.

**Продукт**

Начните с обсуждения программного продукта, созданного в результате работы с историями. Обязательно откройте его на ближайшем компьютере и продемонстрируйте то, что получилось, в действии. *Все*, что получилось. Для многих членов больших команд, возможно, это единственный шанс увидеть работу других.

Субъективно, как команда оцените качество получившегося продукта. Обсуждение качества может стать началом многих интересных дискуссий.

• *Обсудите качество пользовательского взаимодействия*. Не только то, как выглядит графический интерфейс, но и то, насколько комфортна работа с продуктом. Соответствует ли качество вашим ожиданиям? Оцените свою работу по пятибалльной шкале (5 – высшая оценка).

• *Обсудите функциональное качество*. Прошло ли тестирование гладко, или обнаружилось много багов? Тестировщики обнаружили больше багов, потому что добавилось больше кода или потому, что получили больше времени на тестирование? Оцените свою работу по пятибалльной шкале (5 – высшая оценка).

• *Обсудите качество кода*. Код, который вы написали, будет легко расширять и поддерживать? Или вы просто написали очередную порцию унаследованного кода? Оцените свою работу по пятибалльной шкале (5 – высшая оценка).

Запишите истории для исправления проблем качества, обнаружившихся в вашем продукте.

Если вы участвуете как в работе по исследованиям, так и в реализации (как и должно быть), обсудите свою исследовательскую работу в последнем цикле. Что вы делали? Что нового узнали?

**План**

Если ваша работа организована с помощью ограниченных во времени спринтов или итераций, то ее первый шаг – составление плана и оценка того, сколько вы можете сделать за это время. Как обстоит дело с вашим планом?

• *Посмотрите, какие истории реализованы до конца, а какие – нет.* Это может быть сложнее, чем вы думаете. Обсуждение поможет вашей команде прийти к общему определению того, что считать сделанным. Включает ли в себя «сделано» готовые автоматические тесты? Означает ли это, что закончено ручное тестирование?

Означает ли это, что владельцы продукты или UI-дизайнеры проверили полученный результат?

• Общее количество историй, которые вы считаете сделанными, – это ваша *производительность*.

• Общее количество историй, взятых в разработку, но незаконченных. Если их много, значит, процесс планирования требует пересмотра. Я называю этот показатель *вылетом*. Кто-то, с кем я работал в прошлом, употреблял выражение *«после вчерашнего»* по аналогии с остаточными явлениями от злоупотребления горячительным, которые тоже ведут к головной боли.

• Обсудите количество времени, выделенное для *исследовательской работы*. Использовали ли вы его полностью? Понадобилось ли больше, чем было запланировано? Если потрачено очень мало, это может позднее выйти вам боком, когда вы окажетесь не готовы к разработке и не будете достаточно уверены в правильности выбранного направления. А превышение запланированного времени просто-напросто грозит тем, что вы не уложитесь в сроки разработки.

**Процесс**

Обсудите, как протекала ваша работа в последнем цикле разработки. Можете ли вы внести какие-то изменения в свою манеру работы, которые изменят качество к лучшему, сделают ваши оценки времени более точными или хотя бы сделают ежедневную работу веселей? Я серьезно: если вы получаете от своей работы

удовольствие, честное слово, ваша эффективность растет[32].

• Начните с обсуждения изменений, которые вы запланировали в прошлом цикле. Сработали ли они? Хотите ли вы сохранить их или отменить?

• Обсудите изменения, которые хотите опробовать в следующем цикле. Не выбирайте значительные. Небольшие изменения работают лучше всего. Попытка изменить слишком много всего за раз – то же самое, что попытка сразу сделать слишком много работы. Вы просто потерпите поражение и разочаруетесь.

Вот и все. Вы благополучно извлекли уроки из историй, которые реализовали, и значение для этого имеет *вся* проделанная вами работа.

# Оцените свою работу с помощью других сотрудников организации

После того как команда вынесла справедливую оценку своей работе, расширьте аудиторию и включите туда всех заинтересованных из вашей организации. Эти люди должны получить представление о дискуссиях, которые вы провели в рамках команды, и компромиссах, на которые пошли. Помните: с точки зрения законченного продукта те истории, которые вы воплотили в работающее программное обеспечение, – лишь маленькие камушки. Люди, не входящие в команду, могут ожидать продукта с большей степенью готовности. Они, скорее всего, укажут на что-то пропущенное, потому что не участвовали в сессиях планирования и не знают, что вы решили отложить эти аспекты на более позднее время. Будьте готовы к этому. Объясните им, как те фрагменты, что вы сделали, встраиваются в большой план. Проведите все это на сессии *оценки заинтересованных сторон*.

### *Рецепт оценки продукта заинтересованными сторонами*

В вашей организации работает много людей, которые кровно заинтересованы в том, над чем вы работаете и чего достигли. Вам нужно сделать эту работу прозрачной для них. В отличие от вашей команды, эти люди вряд ли знают, что именно представляет собой ваша разработка и как она встраивается в общую картину. Следовательно, вы должны связать то, что узнали и чего достигли, с общей картиной продукта. Кроме того, это отличная возможность поучиться у них и получить их поддержку.

Приглашайте всех заинтересованных. Вы должны получить широкую публичную оценку, поэтому присутствовать может любой интересующийся. Не забудьте также пригласить всю команду. Видеть реакцию других людей на свою работу, как позитивную, так и негативную, очень полезно для понимания ее значения.

Запаситесь едой. Уверяю вас, все будут более благожелательны к вашим словам, как следует заправившись углеводами. Даже плохие новости легче воспринимаются, если заесть их печенюшками.

Необходимо рассмотреть информацию в двух категориях: исследовательская работа, в которую вы были вовлечены, а также истории, которые реализовали.

**Оценка исследовательской работы**

Оценка исследований очень важна. Лучший момент для того, чтобы услышать мнение партнеров, – перед тем как потратить уйму времени на разработку чего-то. Если вы продемонстрируете им реальные уроки, извлеченные из тестирования взаимодействия ваших заказчиков и пользователей с продуктом, то партнеры поймут ценность изучения образа мыслей заказчиков и пользователей. Единственное, что может эффективно повлиять на чье-то мнение, – холодные беспристрастные факты.

Коротко обсудите каждую возможность, которую вы рассматривали: для кого она, почему ее решили реализовать и каких результатов ожидаете в случае успеха.

Обсудите и продемонстрируйте работу, которую вы проделали, чтобы разобраться в проблеме и ее решении.

Обсудите и продемонстрируйте прототипы, которые вы создали, и эксперименты, которые провели. Можно оценить также, что говорят о вашем решении пользователи и заказчики.

**Оцените разработку**

По моему опыту, больше всего представителей бизнеса интересует, что и когда вы предъявите заказчикам и пользователям. Так и должно быть, ведь вы будете наблюдать реальные результаты не только после выпуска минимально жизнеспособного решения. Ваши партнеры заинтересованы в прогрессе, которого можно ожидать, двигаясь в выбранном направлении.

Оцените работу по предъявлению продукта, которую вы завершили на уровне «решение за решением». В данном случае вы должны думать о минимально жизнеспособном решении, как о большом камне, что больше соответствует видению партнеров.

**Для каждого решения**

Рассмотрите цели решения, заказчиков и пользователей, которым оно адресовано, а также полученные результаты. Очень полезно вспомнить, почему мы взялись разрабатывать это и что в данном случае означает успех.

Обсудите и продемонстрируйте результаты историй, реализованных для каждого решения. Выслушайте мнение о них, которое выскажут партнеры. Если у них была возможность оценить вашу работу, когда вы проводили исследования, сейчас они могут сказать: «Все по-прежнему хорошо».

Обсудите истории в целом. Если вы следуете стратегии наподобие *«Моны Лизы»*, придется объяснить партнерам, почему программный продукт пока что выглядит незаконченным. Помните: возможно, ваши собеседники предпочли бы увидеть квадратный дюйм законченного портрета, а не набросок во весь холст в эквиваленте программного обеспечения.

Поделитесь с ними тем, насколько вы приблизились к состоянию готовности своего решения. Сколько еще осталось сделать? Узнали ли вы, разрабатывая решение, что-либо, что может повлиять на его успешное предъявление?

Будьте готовы записать истории для новых возможностей или для изменений, которые необходимо сделать.

Может оказаться, что другие присутствующие в комнате не знакомы с тем, что вы разрабатываете, и поэтому будут вносить явно неудачные предложения. Похвалив идею как таковую и поблагодарив за нее, напомните им о целевой аудитории и ожидаемых от решения результатах и объясните, почему их предложение не будет способствовать достижению последних.

Сохраняйте работу прозрачной для всех в компании. Не упускайте возможности получить одобрение за то, что вы делаете и чему учитесь.

# Достаточно

Если я пользуюсь каким-то продуктом, который мне нравится, то, уверен, я не обращаю внимания на все его крохотные детали и стоящие за ними решения. Фактически, если программа работает неплохо, я вообще не обращаю на нее слишком много внимания. Я не замечаю, как мой смартфон теряет и затем восстанавливает соединение с Интернетом. Я не концентрируюсь на том, что, обновив какой-нибудь элемент в мобильном приложении для управления задачами, сразу могу увидеть эти изменения в веб-версии. Но все это очень важные атрибуты качества, и если их не будет, я непременно это замечу. Через вашу команду прошло огромное множество деталей, но, как ни парадоксально, вам не нужно, чтобы все они были замечены пользователями. Скорее, вам нужно, чтобы они их не замечали.

Основные знания вы получаете от представителей бизнеса в своей организации, заказчиков, которые купили ваш продукт, и отдельных пользователей, которые, получив ваше приложение, будут использовать его и смогут оценить, насколько оно помогает им достичь их целей.

**Бизнес-партнеры.** Для них может быть достаточно добавления функциональности, критической с точки зрения привлечения новых заказчиков. Или, возможно, им достаточно полученных вами сведений о деталях, которые обязательно нужно включить в эту функциональность для обеспечения конкурентоспособности.

**Заказчики.** Для них может быть достаточно добавления функциональности, которая обеспечит им реальную выгоду, когда они или сотрудники их организации начнут использовать обновленное приложение.

**Пользователи.** Для них может быть достаточно добавления функциональности, которая обеспечит достижение ими одной из их целей.

Если вы приложили достаточно стараний в процессе дробления камней, то должны были получить много маленьких, простых в реализации частей. Каждая из этих частей позволяет вам и вашей команде что-либо узнать, но если вы сделали все правильно, это может быть не так для других групп.

Я представляю, что мы складываем вместе эти маленькие кусочки программного продукта, как детали LEGO. Я складываю их на старомодные весы с двумя чашами и противовесом на одной стороне. В качестве противовеса выступает больший кирпичик LEGO, представляющий понятие «достаточно» – достаточно, чтобы позволить пользователю выполнить задачу или достичь цели, достаточно для заказчиков, чтобы извлечь какую-то выгоду для себя, достаточно для бизнес-партнеров, чтобы они ощутили реальную помощь в достижении организацией ее целей. Как только деталей становится достаточно и их кучка перевешивает, наступает время тестировать приложение с пользователями, давать его на оценку заказчикам или бизнес-партнерам.

Вы как команда не просто коллег, но единомышленников, должны рассмотреть результаты каждой отдельной истории, чтобы изучить и улучшить не только ваш продукт, но и способ планирования и манеру взаимодействия. Получая обратную связь и обучаясь у других групп, всегда учитывайте, что означает «достаточно» для них.

# Учитесь у пользователей

На начальном этапе мы можем быть твердо уверены, что разработка идет в верном направлении, но, чтобы сохранять эту уверенность, очень важно тестировать программный продукт с участием пользователей.

Обратите внимание: я сказал «тестировать». Просто продемонстрировав пользователям программный продукт и рассказав о нем, ничему научить их не получится. Что толку, если пользователи увидят новые возможности, должны будут вообразить, как их использовать, и решить, нравится им это или нет. Это все равно что рассматривать новую машину в салоне и представлять себе, понравится ли вам ее водить. Понять, сможет ли новая функциональность помочь добиться целей, пользователи смогут только на тест-драйве продукта. Да и мы, команда, узнаем больше, наблюдая за тем, как пользователи работают с программой. Если вы и ваша команда проводили качественные обсуждения историй, то, наверное, говорили и о пользователях, о том, почему они могут оценить то, что вы разрабатываете, и как могут это использовать. Только наблюдение за ними может по-настоящему подтвердить или опровергнуть эти гипотезы.

Как только вы разработаете достаточно, чтобы позволить пользователям сделать что-то значимое для них, настало время для тестов. Тестировать можно не только совершенно новое. Вы можете, к примеру, проверить изменения или небольшие улучшения, внесенные в уже существующую функцию. Уделите немного времени наблюдению за пользователями, решающими с помощью вашего продукта реальные задачи.

# Извлекайте уроки из пользовательских релизов

Вы создали небольшую часть программного обеспечения и рассмотрели каждый небольшой кусочек как команда. Периодически вы оценивали их с представителями бизнеса в своей организации, а также с заказчиками, которые купят ваш продукт, и с пользователями, которые будут с ним работать. Но на самом деле нас интересуют не программа, а результаты, которые мы получим после того, как приложение будет предъявлено пользователям, а они приступят к работе с ним.

Как только вы почувствуете, что создали достаточно и уверены в возможности получить эти результаты, настало время выпустить ваше произведение в свет.

Обязательно нужно запланировать, что вы изучите с каждым релизом. Пожалуйста, никогда не выпускайте программный продукт просто так, чтобы потом сидеть и ждать, пока заказчики и пользователи не начнут жаловаться. Эти жалобы – тоже результаты, но, как правило, эти индикаторы не совсем своевременно отражают впечатления ваших пользователей от работы с продуктом. После каждого релиза всей командой обсуждайте, какие параметры вы можете использовать и как организовать наблюдение за пользователями своего продукта, чтобы увидеть, удалось ли вам достичь запланированных результатов. Обсудите и придите к соглашению, как вы будете:

• встраивать в продукт показатели, которые позволят вам отслеживать использование новых функциональностей;

• планировать мероприятие, на котором будете наблюдать, как пользователи работают с вашим новым продуктом.

Регулярно обсуждайте всей командой, что нового вы узнали, а затем на основе своих идей предлагайте улучшения и составляйте больше историй. Возможно, кое-что из увиденного вами будет заслуживать немедленной реализации, а все остальное отправится в бэклог возможностей.

# Результаты по расписанию

Работая с рядом компаний и рядом продуктов, мы можем позволить себе выпускать релизы по состоянию «достаточно». Но существует довольно много компаний и продуктов (если не сказать, что их большинство), где мы обязаны выпускать релизы в определенное время. Если в этом случае эффективно использовать релизную стратегию, то можно заложить крепкую основу в начальных, дебютных историях, разработать основную часть с помощью историй миттельшпиля, а когда придет время релиза, красиво разыграть истории эндшпиля.

А сейчас я должен напомнить вам несколько горьких истин о разработке программного обеспечения.

*Программное обеспечение никогда не бывает законченным.*

Вы закончили реализацию каждой истории, которые ваша команда взяла в разработку в очередном коротком цикле. Но, скорее всего, вы не закончили каждую историю, которую представляли себе на начальном этапе или считали необходимым реализовать по итогам изучения. Вы используете эффективную стратегию разработки, однако ваш продукт будет хорош настолько, насколько этого можно достичь в заданное время.

*Результаты всегда непредсказуемы.*

Независимо от того, сколько работы вы проделали, чтобы убедиться в том, что разработка идет в верном направлении, пользователи чаще всего будут вести себя совершенно не так, как вы ожидали. Поэтому планируйте, как извлекать уроки из каждого релиза. Планируйте изменения, которые вы внесете на основе того, что вам удалось узнать.

*Улучшения, внесенные после релиза, самые ценные.*

Самые удивительные открытия и самые полезные уроки вы извлечете, когда пользователи начнут решать реальные задачи с помощью вашего продукта. Если вы выделили время на то, чтобы увидеть и измерить реальные результаты, то будете вознаграждены за это, когда пользователи по-настоящему полюбят ваш продукт, а организация начнет получать прибыль.

# Используйте карты для оценки готовности релиза

Вы продвигаете свой продукт к релизу – историю за историей. Как только намеченная вами дата релиза начинает угрожающе приближаться (а такая дата есть всегда), задайтесь вопросом о каждой основной пользовательской линии действий: «Если бы нужно было выпустить продукт прямо сейчас, какую оценку мы бы поставили сами себе?» Если вы используете буквенные оценки вроде тех, что приносят из школы мои дети, то в конце концов получите целый табель для своего продукта.

Например, если у вас продукт или функциональность с пятью основными линиями действий пользователей, а в табеле написано: A, A-, B+, D, B+[33], то, разумеется, за оставшиеся до релиза недели вам следует сосредоточиться на той части, которая получила оценку D. Если в конце релиза вы получаете только A и B, это довольно хорошо. Конечно, быть круглым отличником лучше, но уложиться в срок в мире разработки гораздо важнее.


По мере приближения к дате релиза регулярно все вместе оценивайте готовность продукта. Уверяю вас, это очень интересно.

Книга тоже подходит к концу. Если вы дочитали до этого места, то, наверное, можете высказать мнение и о ее готовности к релизу. Вы можете открыть оглавление и поставить буквенную оценку каждой главе. Затем сфотографируйте страницу на смартфон и отправьте мне – будет очень интересно увидеть, что получилось.

# Конец или не конец?

Как и любой хороший программный продукт, на самом деле эта книга не закончена. На протяжении всего текста я приводил множество удивительных примеров, о которых узнал от своих друзей и знакомых. Они рассказывали мне замечательные истории использования историй и карт историй. Еще больше историй так и осталось на моем жестком диске, и я ужасно расстроен, что из-за нехватки времени не смог обработать их и включить в книгу.

За кадром также осталось много деталей об историях и карт историй, которые я с удовольствием описал бы. Кроме того, я уверен, что у вас есть вопросы без ответов о том, как использовать истории в вашем конкретном случае. Сейчас, почти дописав книгу, я очень обеспокоен этим.

Как и любой человек, когда-либо работавший программистом, дизайнером графического интерфейса или менеджером продукта, я могу сказать вам, что редко бываю доволен релизом продукта. Так происходит потому, что я знаю обо многих вещах, которые мог бы включить в книгу, но не включил, а также обо многих деталях, которые могли бы быть проработаны лучше, но не доведены до совершенства, так как времени на шлифовку не хватило. Если вы тоже переживаете о продукте, который создаете, уверен, вы понимаете, что я чувствую.

Повторю цитату да Винчи, которую я приводил в главе 4.

> *«Работу над великим произведением искусства невозможно завершить – ее можно только прервать.»*

Я вовсе не хочу сказать, что эта книга – великое произведение искусства, но скажу, что прервал работу над ней, когда еще очень многое можно было бы сделать. Я оставляю это многое вам и хотел бы услышать от вас, какие новые методы сотрудничества для создания великих проектов вы нашли.

# Благодарности

Эта часть книги далась мне труднее всего. Мне повезло: поддержку в течение всей моей карьеры мне оказывало великое множество людей. Я получил и продолжаю получать значительное поощрение от всех, с кем мне довелось встречаться и работать. Поэтому меня очень беспокоит, что, начав благодарить людей, я могу нечаянно забыть кого-нибудь. Если я пропустил вас, то очень сожалею об этом и подозреваю, что вы в хорошей компании.

Еще одно, в чем я твердо уверен, – идеи, изложенные в этой книге, не только мои. Я и раньше слышал, что оригинальных идей совсем не осталось. Но лично я всему, что знаю, научился у мудрых людей, с которыми мне посчастливилось работать последние 20 лет. Друзья и приятели познакомили меня в теории и на практике с множеством новых идей, практик и открытий. За время наших долгих дискуссий я научился интерпретировать и глубоко понимать опыт, который получил, осваивая свое ремесло. Мне сложно получать признание за любые идеи, озвученные в этой книге, поскольку большинство из них я позаимствовал или, честно признаться, украл у других людей.

Если мне приходит в голову что-то, что я считаю оригинальной идеей, я немедленно вспоминаю о *криптомнезии*. Это забавное слово означает неумышленный плагиат, в котором замечены в том числе и такие уважаемые люди, как Джордж Харрисон и Умберто Эко. Криптомнезия означает, что мы вспоминаем нечто давно забытое, не осознавая, что это воспоминание. Криптомнезик уверен, что блестящая идея, которая только что пришла ему в голову, – это что-то новое и оригинальное, а вовсе не мысль, которую он когда-то прочитал или услышал, а затем забыл. Поэтому я говорю, что люди, которым я приношу здесь благодарности, скорее всего, являются истинными авторами идей, которые я ненамеренно украл.

Вот теперь я могу начать.

Когда-то я практически отказался от идеи написать книгу. На протяжении десяти лет я пытался это сделать, но ничего не получалось. Я был способен написать короткую статью или произнести речь, но любая попытка создать текст длиннее пары тысяч слов заканчивалась поражением. Процесс написания книги был схож с таксидермией: я словно брал что-то живое, а затем убивал это и делал чучело. Лучшее, на что я мог надеяться, – мое произведение хотя бы выглядит живым. Питеру Экономи удалось вырвать меня из этого замкнутого круга. Его многолетний писательский опыт, а также неубиваемый позитивный настрой и стабильная поддержка наконец помогли мне обрести свою писательскую манеру. Я очень благодарен Питеру. Если у вас тоже никак не получается написать книгу, именно он – тот, кто вам нужен.

Мартин Фаулер, Алан Купер и Марти Коган – три моих героя. Для меня огромное счастье быть знакомым с ними, работать с ними и вести долгие разговоры. Их умы влияли на меня в течение всей моей карьеры. Двое из них думали, что три предисловия в книге – плохая идея, но я очень рад, что настоял на своем, а они со мной согласились. Их знания в инженерном деле, пользовательском взаимодействии и мышлении разработчика чрезвычайно важны для создания успешных продуктов. Я думаю, очень важно, что и вы, читатель, познакомились с ними.

Алистер Коберн – мой друг и наставник более десяти лет. Я совершенно уверен, что большинство мыслей, которые я считаю своими блестящими идеями, были украдены у

Алистера, из наших долгих бесед с ним. Само название модели из карточек с историями, которые я раскладывал на столах и развешивал на стенах, – карты историй, – родилось во время одной из таких бесед. Я хотел объяснить Алистеру суть и постоянно повторял: «Это просто карта из историй». В конце концов он спросил: «Почему бы не назвать эту штуку именно так?» Это название превосходно заменило все те глупые имена, которые я пытался придумать.

Сама практика изложения историй и построения продуктовых бэклогов – то, что я делаю с помощью карточек уже много лет, произошла из опыта, который передал мой друг Ларри Константайн. Метод карт историй и принципы пользовательского взаимодействия, возможно, никогда бы не родились, не имей я возможности учиться у Ларри.

Дэвид Хассман долгие годы является моим другом, сторонником и просто близким человеком. Именно наблюдения за Дэвидом, рассказывающим истории, и его поддержка помогли мне обрести тот путь, по которому я иду сегодня. Дэвид составлял карты историй еще до того, как было придумано это название.

Эта книга никогда не была бы закончена без поддержки Тома и Мэри Поппендек. Том, в частности, прочел некоторые мои худшие таксидермические работы за последние десять лет, но все еще продолжает меня поддерживать. Несколько месяцев назад он отказался покидать мой дом до тех пор, пока я не отправлю последний вариант рукописи в O'Reilly. Если бы он не сделал этого, я бы до сих пор продолжал копаться в тексте и никогда не счел бы его достаточно хорошим.

Другие мои друзья, которые поддерживали меня и давали хорошие советы на всем протяжении работы, – Йон и Кей Йохансен, Аарон Сандерс и Эрика Янг, Джонатан Хауз, Нэйт Джонс и Кристина Дель Прет.

Особую благодарность я хочу выразить Гэри Левитту, всем сотрудникам Globo.com, Эрику Райту из Liquidnet, а также всем моим друзьям из Workiva, которые позволили мне рассказать свои истории в первых главах книги.

За последние годы очень многие люди приходили ко мне, чтобы рассказать, как они используют карты историй или как применили какой-нибудь данный мною совет. Сейчас я открою свой самый страшный секрет: думаю, что научился у них гораздо большему, чем они у меня. Я был счастлив использовать в этой книге материал, который получил от нескольких из них. Особое спасибо хочу сказать тем, кто внес вклад в написание книги, но был упомянут весьма коротко: это Джош Сайден, Крис Шинкл, Шериф Мансур, Бен Кротерс, Майкл Вэз, Мартина Луэнзман, Андреа Шмиден, Сидни Дойл, Эрин Бейерволтес, Аарон Уайт, Мэт Кроппер, Крис Гансен и Джейсон Кунеш, Рик Кьюзик, Никола Адамс и Стив Барретт.

Есть также большая группа людей – я имел счастье общаться с ними и учиться у них, но из-за жесткого дедлайна не смог включить их истории в книгу. В их числе Ахмад Фахми, Тобиас Хильденбрэнд, Кортни Хэмпфил, Сэмьюэл Боулс, Роуэн Баннинг, Скаут Эддис, Холли Белава и Джейб Блум. И вы, и все остальные, кто читает это, – мне все еще нужны ваши истории! Возможно, я еще выпущу специальную режиссерскую версию книги, куда войдут все вырезанные сцены.

На финальном отрезке подготовки этой книги я получил детальные и очень ценные для меня рецензии от Барри O'Рейли, Тодда Уэбба и Петры Уилл. Все эти детальные комментарии помогли мне навести на книгу окончательный лоск.

И наконец, спасибо Мэри Треселер и производственному отделу O'Reilly за терпеливое отношение к моим задержкам и беспорядочному графику, в результате чего мы все

благополучно дошли до конца.

---

| notes |
| --- |

# Примечания

# 1

Бэклог – набор функциональностей, которые планируется внедрить в проект. – *Примеч. пер*.

# 2

Подробности этой истории многократно освещались в прессе, вот одна из статей: http://edition.cnn.com/TECH/space/9909/30/mars.metric.02/. – *Примеч. пер*

# 3

В точной терминологии, а также разнице между объемом работы и конечным результатом я разобрался, прослушав речь Роберта Фабриканта Behavior is Our Medium. До этого я, как и многие другие, затруднялся внятно изложить то, что хорошо понимал. К счастью, Роберту удалось внести ясность. – *Примеч. авт.*

# 4

Я перефразирую предостережение Кента Бека о неверном использовании термина «требования», которое он высказывает в своей книге «Экстремальное программирование», и полностью согласен с ним. – *Примеч. авт.*

# 5

Автор фото – Piutus, пользователь Flickr, фотография защищена Creative Common Attribution License. – *Примеч. авт.*

Mad Mimi, или Music Industry Marketing Interface. – *Примеч. пер.*

**7**

Позитивное исследование (appreciative enquiry) – модель организации изменений, в которой участники фокусируются не только на том, что плохо, что не работает, что нужно изменить (как это принято в традиционном методе решения проблем – problem solving), но и на том, что достигнуто, что хорошо в настоящем благодаря действиям в прошлом. Иначе говоря, здесь ценят интересы не только новаторов, но и консерваторов. – *Примеч. пер*.

Книга вышла в 2014 году. – *Примеч. ред*.

Около 93 м$^2$. – *Примеч. пер.*

Марти впервые описал, что подразумевает под исследованием продукта, в своей статье 2007 года. Позднее он дал более детальное описание в книге «Вдохновение: Как создавать продукты, которые обожают пользователи» (Inspired: How to Create Products Customers Love by Marty Cagan, SVPGPress). – *Примеч. авт.*

# 11

Я взял идею этой простой визуализации из опубликованной в 2004 году статьи Джона Эрмитеджа «Подходят ли методы Agile для дизайна?». Джон описывает применение данного подхода при проектировании пользовательского опыта. Я думаю, эту метафору можно распространить и на разработку программного обеспечения. – *Примеч. авт.*

# 12

Кале – сорт капусты, по вкусу и текстуре напоминающий бумагу, но ее необычайная полезность заставляет людей, ведущих здоровый образ жизни, считать кале вкуснейшим в мире продуктом. – *Примеч. пер*.

# 13

Научно доказано, что факт, рассказанный в виде истории, намного лучше запоминается, чем просто изложенный факт, – в 22 раза лучше, по словам психолога Джерома Брунера. – *Примеч. авт.*

*Jeffries R. et al.* Extreme Programming Installed. – Addison-Wesley Longman Publishing.

Слово backlog в данном случае можно перевести как «объем невыполненной работы», что уже звучит двояко, кроме того, у него есть значения «задолженность», «недоработка». – *Примеч. пер.*

*DeMarco T. et al.* Adrenaline Junkies and Template Zombies: Understanding Patterns of Project Behavior. – Dorset House.

# 17

Автор фото – Рут Хартнап, размещено на Flickr и лицензировано Creative Commons Attribution License.

# 18

Десятичная классификация Дьюи – система классификации книг, разработанная американским библиотекарем Мелвиллом Дьюи в конце XIX века. – *Примеч. пер*.

# 19

Поговорка Eat their own dog food появилась в Америке в 1970-х годах. Она означает, что сотрудники компании сами пользуются продуктом, который производят, с целью продвижения и/или тестирования. По одной из версий, выражение впервые прозвучало на совещании акционеров Kal Kan Pet Food, где президенту было предложено съесть банку собачьего корма, который производила компания. – *Примеч. пер*.

*Cockburn A.* Agile Software Development: The Cooperative Gameby. – Addison-Wesley Professional, 2004.

# 21

«Ты не можешь всегда получать то, что хочешь, но, однажды постаравшись, обнаружишь, что обладаешь всем, что тебе нужно». – *Примеч. пер*.

# 22

Капкейк (cupcake) – пирожное, напоминающее маленький тортик: основа – кекс, похожий на те, что продавались в свое время в школьных буфетах, сверху – шапка крема. Название происходит от того, что капкейки выпекают в формах-чашечках. – *Примеч. пер*.

Свадебный торт Мэри (фото любезно предоставлено Мэри Трэслер).

Изначально Лиза включила эту прекрасную формулировку в комментарий к речи, произнесенной на IXDA в 2009 году. Ознакомьтесь с ее последней статьей здесь: http://www.disambiguity.com/designbycommunity/. – *Примеч. авт.*

# 25

Методику проведения семинара по историям по методу трех амиго объясняет Райан Томас Хьюитт на сайте Scrum Alliance, http://bit.ly/Utg8er.

*Osterwalder A., Pigneur Y.* Business Model Generation. – Wiley.

*Johnson J.* Chairman of the Standish Group, «ROI, It's Your Job» (keynote) // Third International Conference on Extreme Programming, 2002. – May 26–29. – Alghero, Italy.

*Gage D.* The Venture Capital Secret: 3 Out of 4 Start-Ups Fail // Wall Street Journal, 2012. – September 20. (http://on.wsj.com/UtgMZl).

*Ries E.* The Lean Startup. – Crown Business.

*Blank S.* The Four Steps to Epiphany. – K&S Ranch.

# 31

Любезно предоставлено ITHAKA, © 2014 ITHAKA, все права защищены. – *Примеч. авт.*

Обсуждение улучшений процесса чаще всего называется *ретроспективой*. Существует множество подходов к ее проведению. Если хотите глубже изучить различные подходы, обратитесь к книге: *Derby E., Larsen D.* Agile Retrospectives. – Pragmatic. – *Примеч. авт.*

Оценки приблизительно соответствуют 5, 5–, 4+, 2, 4+. – *Примеч. пер*.