O'REILLY®

Постигая Agile

ЦЕННОСТИ, ПРИНЦИПЫ, МЕТОДОЛОГИИ

Эндрю Стеллман
Дженнифер Грин

Andrew Stellman
Jennifer Greene

Learning Agile

Understanding Scrum, XP, Lean, and Kanban

O'Reilly
2015

Оглавление

ГЛАВА 1
Обучая Agile

Самая важная установка, которая может быть сформирована, — это желание учиться.
Джон Дьюи*. Опыт и образование

Сейчас самое время быть гибкими! Впервые наша отрасль нашла реальный и устойчивый способ решения проблем, над которыми бились поколения разработчиков программного обеспечения. Вот лишь некоторые примеры, которые возможны с Agile:

— Agile-проекты завершаются вовремя, что отлично подходит для тех команд, которые стремятся закончить работу в срок и не превысить смету.

— Agile-проекты обеспечивают высокое качество программного продукта, а это важно для команд, уставших создавать неэффективное, полное ошибок программное обеспечение.

— Код, написанный agile-командами, хорошо сделан и прост в обслуживании. Это большое облегчение для команд, привыкших поддерживать извилистый и запутанный спагетти-код.

— Agile-команды делают потребителей счастливыми, в этом их огромное отличие от разработчиков сложных программ, суть которых пользователи не понимают.

— Но главное, разработчики в эффективной agile-команде трудятся только в рабочее время, поэтому могут проводить вечера и выходные с семьей и друзьями — возможно, впервые за долгие годы.

* Джон Дьюи (1859–1952) — американский философ и педагог, представитель философского направления «прагматизм». Автор более 30 книг и 900 научных статей по философии, социологии, педагогике и другим дисциплинам. *Прим. ред.*

Agile-методологии популярны, потому что многие перешедшие на них команды сообщают об отличных результатах: они создают качественное программное обеспечение, успешнее работают вместе, удовлетворяют запросы своих пользователей и добиваются всего этого в спокойной рабочей обстановке. Некоторые agile-команды даже продвинулись в решении проблем, которые десятилетиями беспокоили программистов. Итак, каким образом команды используют Agile для создания хороших программ? Или, точнее говоря, как *вы* можете при помощи Agile добиться подобных результатов?

Из этой книги вы узнаете о двух самых популярных agile-методологиях — Scrum и XP (экстремальном программировании). Вы прочтете также о Lean (бережливом программировании) и Канбан (Kanban), о том, как они помогают понять принципы создания программ и развить свои навыки. Вы увидите, что четыре школы Agile сосредоточивают внимание на разных отраслях разработки, но у них есть нечто общее: они направлены на **изменение образа мыслей команды**.

Именно изменение образа мыслей превращает группу сотрудников, добавляющую в свою работу несколько agile-методов, в настоящую команду, которая действительно улучшает способ создания ПО. Цель книги — познакомить читателя с обеими сторонами Agile: методами, работающими в повседневной деятельности, а также ценностями и принципами, которые помогают вашей команде полностью изменить свой подход к разработке программ.

Что такое Agile?

Agile — это набор **методов и методологий**, которые помогают вашей команде эффективнее мыслить, работать и принимать решения.

Эти методы и методологии охватывают все области традиционного программирования, включая управление проектами, дизайн и архитектуру ПО, а также оптимизацию процессов. Все методы и методологии состоят из **процедур**, максимально четких и оптимизированных, которые легко применить.

Кроме того, Agile — это **мировоззрение,** поскольку правильное мышление может оказать большое влияние на эффективность овладения процедурами. Это мировоззрение помогает членам команды делиться друг с другом информацией и на основании этих данных самим принимать важные решения по проекту, не полагаясь только на менеджера. Agile-мировоззрение включает открытое планирование, обсуждение дизайна

и совершенствование процессов всей командой. Agile-команда использует методы, при которых все ее участники владеют одинаковой информацией и каждый имеет свой голос при обсуждении применения этих методик.

Реальность agile-методологий для многих команд, не добившихся особого успеха, не оправдывает ожиданий. Причина часто связана с мировоззрением команды, с которым она начинает работу над проектом. Большинство компаний, занимающихся созданием ПО, уже опробовали Agile. Многие достигли успеха, но результаты некоторых нельзя назвать блестящими. Они добились достаточного прогресса в работе над проектами, чтобы оправдать усилия, потраченные на переход к agile-методологиям, но не ощутили ожидаемых изменений. Это и говорит о важности смены мировоззрения всей командой при переходе на Agile.

Но что означает смена мировоззрения? Если вы входите в команду программистов, то каждый день обдумываете, разрабатываете, пишете и сдаете программное обеспечение. Что ваше мировоззрение должно делать со всем этим? Оказывается, методы, которые вы применяете в повседневной работе, во многом зависят от отношения к ним.

Вот пример. Одна из самых распространенных agile-процедур, которые берут на вооружение команды, — это **ежедневные планерки на ходу**, во время которых члены команды рассказывают, над чем работают и с какими проблемами сталкиваются. Такие собрания длятся недолго, потому все участники стоят. Многие команды, внедрившие ежедневные планерки на ходу, добились больших успехов в работе над проектами.

Итак, представьте себе, что менеджер проекта только что узнал об agile-методологии и хочет внедрить в проект ежедневные митинги. Но выясняется, что не все в команде одобряют эту идею. Один из разработчиков недоволен появлением еще одного совещания, его возмущает, что ежедневно придется ходить на встречу, где у него будут выспрашивать о текущей работе.

Что же происходит? Может быть, разработчик ведет себя нерационально? Или менеджер проекта слишком требователен? Почему такая простая и общепринятая процедура порождает конфликт?

И у менеджера проекта, и у разработчика своя — и вполне разумная — позиция. Одна из основных проблем менеджера проекта — затрата массы усилий на планирование проекта. Но когда при создании ПО команда сталкивается с проблемами, она сразу начинает отклоняться от плана. Приходится потрудиться, чтобы оставаться в роли руководителя команды, поэтому ему нужно вносить коррективы в план и помогать сотрудникам справляться с трудностями.

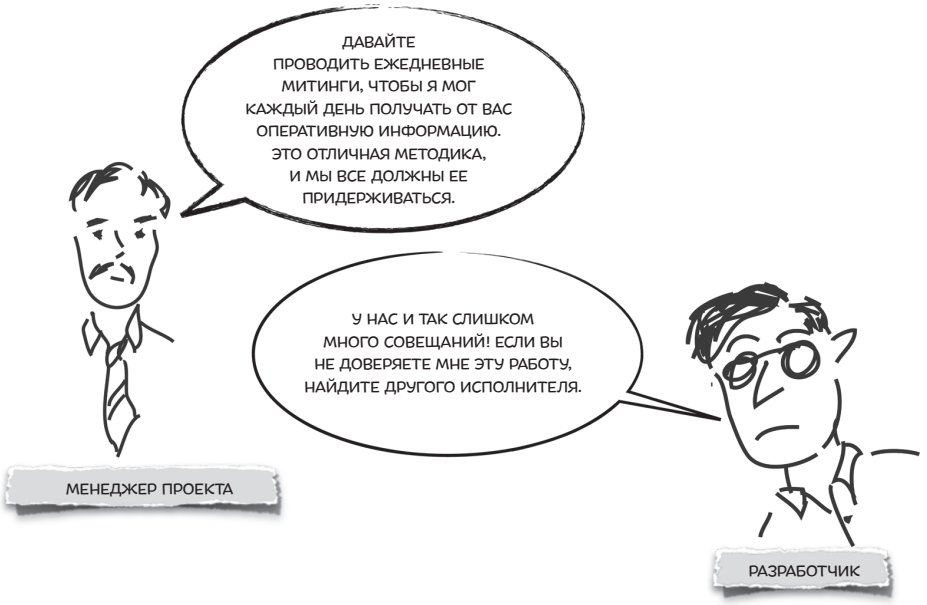

Рис. 1.1. Менеджер проекта, желающий, чтобы команда проводила ежедневные митинги, удивлен, что это нравится не всем

А разработчик недоволен тем, что его по нескольку раз в день прерывают, заставляя приходить на совещания, из-за чего ему сложно выполнить работу в срок. Он и так знает, что нужно для написания кода, поэтому не нуждается в чьих-то рассуждениях о планах и изменениях. Его цель — остаться наедине с кодом, и совещание — это последнее, что ему необходимо.

А теперь представьте себе, что менеджер проекта сумел убедить всех — даже несговорчивого разработчика — посещать ежедневные митинги. Как будут выглядеть эти встречи? Менеджер сосредоточится на том, как члены команды отклоняются от его плана, поэтому будет стараться получить информацию о ходе работы каждого сотрудника. А разработчик захочет скорейшего окончания совещания, поэтому, не особенно вслушиваясь в чужие выступления, постарается, дождавшись своей очереди, высказаться покороче, чтобы не затягивать время.

Скажем честно: именно так и проходят многие митинги. И хотя это не оптимальный способ, даже такое ежедневное совещание *принесет результаты*. Менеджер проекта выявит проблемы с выполнением плана, а разработчик получит выгоду в долгосрочной перспективе. Ведь проблемы, которые его *действительно* касаются, лучше решать на ранних этапах. То есть вся

процедура сэкономит команде больше времени и усилий, чем потребуется на ее выполнение. Так что этим стоит заниматься.

Но как быть, *если у разработчика и менеджера проекта совсем иное мировоззрение?* А каждый член команды будет относиться к ежедневному митингу совершенно по-другому?

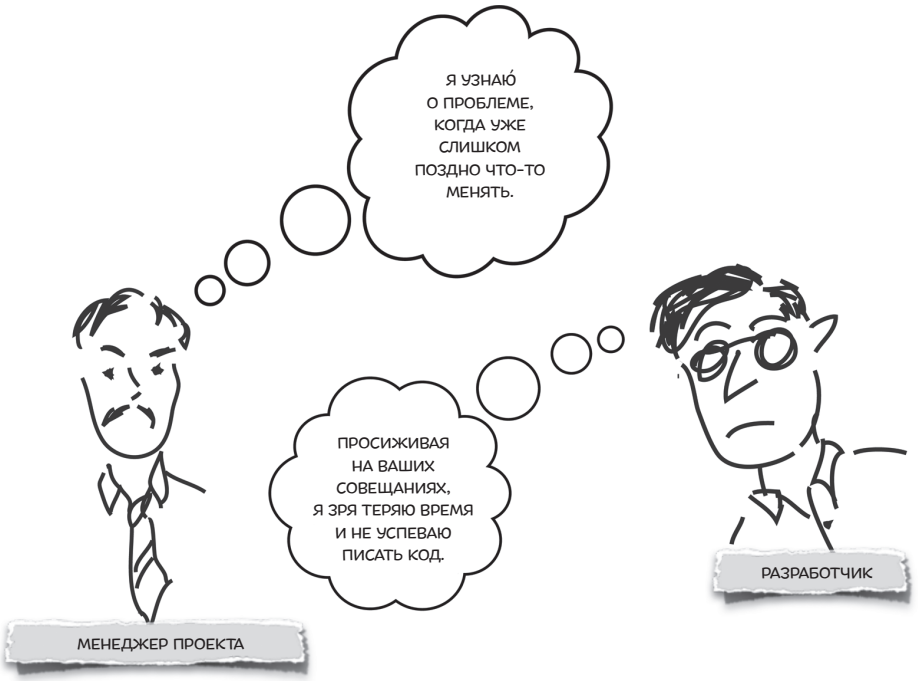

Рис. 1.2. Кажется, у обоих есть веские основания для собственного мнения о ежедневных митингах. Как это скажется на проекте?

Например, если менеджер проекта почувствует, что проект планируется всеми членами команды? Тогда он будет выслушивать каждого не для того, чтобы узнать, насколько тот отклонился от плана, а чтобы понять, какие изменения внести в план, *над которым работала вся команда*. Вместо того чтобы навязывать план и затем оценивать, насколько точно следует ему команда, менеджер будет работать вместе с ее членами, выбирая наилучший подход к проекту. То есть ежедневный митинг — это способ совместной работы, при котором люди убеждаются, что поступают наилучшим образом в каждый момент времени. Поскольку данные о проекте меняются ежедневно, команда использует митинги для принятия максимально эффективных решений. Так как команда встречается каждый день,

изменения, о которых сообщается на совещании, могут быть внедрены немедленно. Ведь теперь не нужно тратить массу времени и усилий, двигаясь в неверном направлении.

А что если разработчик почувствовал, что цель совещания — не только доложить обстановку, но и понять, как продвигается проект, ежедневно вместе искать возможности для оптимизации процесса? Тогда ежедневный митинг *становится важным для него*. Хороший разработчик, как правило, имеет собственное мнение не только о своем коде, но и об общем направлении проекта. Ежедневный митинг — это способ убедиться, что проект реализуется разумно и эффективно. Разработчик понимает, что в долгосрочной перспективе эта процедура принесет пользу его работе, поскольку все, что от него не зависит, тоже выполняется хорошо. Кроме того, он знает: если на совещании придется упомянуть о проблемах планирования, то все прислушаются к его мнению и работа над проектом пойдет еще лучше.

Рис. 1.3. Когда каждый член команды чувствует, что обладает равными правами при планировании проектом и управлении им, ежедневные митинги обретают ценность и высокую эффективность

Иными словами, если члены команды полагают, что ежедневный митинг — это очередная планерка, которую придется вытерпеть, то его все равно стоит проводить. Такое совещание лишь немногим эффективнее традиционных планерок. Но если команда верит: ежедневный митинг — это способ

убедиться, что все работают правильно и для достижения единой цели, что при обсуждении выслушают мнение каждого участника, — то такое собрание становится гораздо эффективнее и приносит настоящее удовлетворение. Разработчик понимает: такое совещание в долгосрочной перспективе помогает ему и всей команде. Менеджер проекта убежден: если в реализации плана принимают участие все сотрудники, то результаты будут выше. Тем, кто разделяет эти взгляды, ежедневные митинги помогают быстрее работать, активнее общаться и качественнее выполнять поставленные задачи.

Это только один из примеров того, как мировоззрение и отношение команды могут повлиять на успешное усвоение agile-методик. Важная цель этой книги — помочь понять, каким образом мировоззрение команды отражается на проектах и вашем отношении к Agile. Изучая Scrum, экстремальное и бережливое программирование, а также Канбан, вы узнаете обе стороны Agile — принципы и методы — и то, как они помогут лучше создавать программное обеспечение.

Кому следует прочитать эту книгу

Можно ли сказать, что одна из приведенных ниже ситуаций возникала у вас и вашей команды?

Вы попробовали agile-методики, но это не помогло. Возможно, вы внедрили ежедневные митинги, команда совещается каждый день, но у вас все равно масса проблем и вы пропускаете дедлайны. Или вы начали писать пользовательские истории и анализировать их с командой и заинтересованными лицами, но разработчики все еще сталкиваются с изменениями в последнюю минуту, добавляя дополнительные функции. А может быть, ваша команда решила полностью перейти на Agile, избрав Scrum или экстремальное программирование, но это выглядит пустой тратой времени, как будто все делают то, что от них требуется, но польза для проектов невелика.

Возможно также, что вы еще не пробовали перейти на Agile, но понимаете: команда столкнулась с серьезными проблемами, а что делать — непонятно. Вы надеетесь, что Agile поможет работать с требовательными пользователями, которые постоянно меняют свое решение. Любое изменение, которого требует заказчик, означает дополнительную работу для вашей команды и ведет к спагетти-коду, из которого так и торчат скотч и скрепки. Поэтому программы становятся все более уязвимыми и сложными для техподдержки. Возможно, ваши проекты — это просто контролируемый хаос; программы пишутся за счет многочасовой работы и личного героизма, так что вы рассматриваете Agile как единственный выход.

Или вы как руководитель обеспокоены тем, что команды, работающие над важными проектами, могут подвести? Возможно, вы слышали об Agile, но плохо представляете, о чем идет речь. Можно ли заставить команду перейти на Agile, или нужно сначала изменить и свое, и командное мировоззрение?

Если какая-то из описанных ситуаций вам знакома и вы хотите улучшить работу команды, то эта книга для вас.

Мы рассказываем, почему agile-методологии были разработаны именно таким образом, с какими проблемами призваны бороться, какие ценности, принципы и идеи они воплощают. Мы объясняем не только «как», но и «почему», то есть помогаем понять принципы, которые применимы к конкретным проблемам развития, характерным для вашей команды, компании и проектов. И мы покажем, как пользоваться этой информацией при выборе методологии и практик.

Еще одна группа людей, для которых написана эта книга, — **agile-коучи.** Команды и компании все чаще полагаются на них, поэтому они должны помочь усвоить agile-методологии и процедуры и изменить мировоззрение членов команды. Если вы agile-коуч, то мы предоставим вам инструменты, чтобы помочь лучше донести идеи до обучаемых и преодолеть некоторые проблемы, с которыми ежедневно сталкиваются те, кто решил перейти на Agile.

Цели, которые мы ставим в этой книге

Мы хотим, чтобы вы:

— усвоили идеи, которыми руководствуются эффективные agile-команды, а также объединяющие их ценности и принципы;

— познакомились с самыми популярными школами — Scrum, экстремальным и бережливым программированием и техникой Канбан — и поняли, как они могут считаться agile-методологиями, несмотря на все их различия;

— научились конкретным agile-методам, которые сможете сразу внедрить в свои проекты. Мы стремимся дать представление о базовых ценностях и принципах, которые понадобятся, чтобы это внедрение было эффективным;

— лучше понимали свою команду и компанию, смогли выбрать тот agile-подход, который соответствует вашему мировоззрению (или максимально близок к нему), а также чтобы начали усваивать новое мышление, которое поможет стать эффективным agile-коллективом.

Каким образом различные agile-методологии и процедуры обеспечивают создание более совершенного программного обеспечения? Почему они дают команде возможность лучше управляться с изменениями? Что значит гибкость? Действительно ли важно использовать карточки для планирования или, например, стоять во время совещаний? Все эти вопросы нередко смущают людей, начинающих свой путь по освоению agile-методологии. Но к концу книги вы сможете ответить на них.

Почти все блоги и статьи, в которых обсуждается гибкая разработка ПО, начинаются с утверждения «Agile — хорошо, водопад — плохо». Но почему Agile — это хорошо, а водопад — нет? Почему они противостоят друг другу? Можно ли работать в команде, которая практикует водопадную модель* (Waterfall), и оставаться гибким? Прочитав книгу до конца, вы найдете ответы и на эти вопросы.

Продвижение Agile в ваше сознание любыми необходимыми средствами

Книга называется «Постигая Agile», потому что мы *действительно* хотим, чтобы вы постигли Agile. В течение более чем 20 лет мы активно сотрудничали с командами, постоянно создающими ПО для реальных пользователей. Последние десять с лишним лет мы пишем книги о создании ПО (две из них, очень успешные, вышли в издательстве O'Reilly в серии Head First, посвященной управлению проектами и обучению программированию). Нам удалось научиться доносить до сознания читателя сложные технические идеи, не нагоняя на него тоску.

Мы сделали все возможное, чтобы наш материал был максимально интересным и увлекательным… но нам нужна ваша помощь. Вот способы и методы, которые помогут удержать в голове все эти идеи.

ПОВЕСТВОВАНИЯ

Обозначаются значком  . Вспомните последнюю техническую книгу, которую вы читали. Можете ли вы воспроизвести все основные темы, которые в ней излагались, и их порядок? Вероятно, нет. А теперь подумайте

* Водопадная модель (англ. waterfall model, иногда называют «каскадная модель») — модель процесса разработки программного обеспечения, при котором команда вначале определяет требования к продукту, планирует проект в целом, разрабатывает программное решение, а затем создает код и тестирует продукт.

о фильме, который смотрели недавно. Вы помните основные элементы сюжета и порядок, в котором они происходили? Наверняка да. Дело в том, что мозг лучше запоминает то, что вызывает у нас эмоциональную реакцию. В книге мы стараемся учитывать это обстоятельство. Мы будем использовать повествования — с участием людей, диалогами и конфликтом, — чтобы показать, как в действительности выглядит переход на Agile. Обычно у этих людей возникают проблемы.

Чего мы хотим от вас. Постарайтесь представить себя в похожей ситуации: это обеспечит эмоциональную связь с данными идеями, поэтому будет проще их запомнить и понять. Отнеситесь к этим рассказам без предубеждения, особенно если вы не большой любитель художественной литературы. В каждом повествовании заключен свой урок, и все они раскрывают основную тему книги.

ИЛЛЮСТРАЦИИ

Люди обучаются по-разному. Некоторым важны визуальные образы, поэтому они легче воспринимают идеи, увидев картинки. Мы хотим снабдить вас максимальным количеством инструментов обучения, поэтому включили в книгу множество иллюстраций. В некоторых случаях мы полностью положились на визуальные метафоры, например используя геометрические формы для передачи различных функций или шестеренки, символизирующие комплексные программы.

Чего мы хотим от вас. Если визуальные образы не имеют для вас большого значения, то некоторые иллюстрации могут показаться избыточными, даже бессмысленными. Это хорошая возможность для обучения: нужно попытаться понять, что человек с визуальным подходом может извлечь из этой иллюстрации. Это поможет лучше понять общую идею.

ИЗБЫТОЧНОСТЬ

В большинстве технических книг существует определенный порядок: идея подается, полностью описывается, а затем автор переходит к следующей. Это эффективный способ передать как можно больше информации, но наш мозг работает иначе. Иногда необходимо не раз и не два взглянуть на идею, прежде чем вы воскликнете «Понятно!». Вот почему мы порой будем возвращаться к одной и той же теме по несколько раз в течение всей книги. **Это намеренная избыточность** — таким способом мы хотим помочь вам поскорее сказать «Понятно!».

Чего мы хотим от вас. Когда вы читаете об одном и том же по несколько раз, возникает искушение спросить: «Разве об этом уже не говорили?» Говорили! И очень хорошо, что вы это заметили. Но есть читатели, которые не обратили на это внимания, да и вы сами наверняка не каждый раз замечаете избыточность. Все это делается для того, чтобы помочь вам учиться.

УПРОЩЕНИЕ (В ПЕРВУЮ ОЧЕРЕДЬ!)

Иногда сложную тему проще понять, если сначала едва коснуться ее и лишь затем погрузиться полностью. В книге мы так и поступаем: сначала вводим упрощенную (но технически верную!) версию идеи, а затем конкретизируем ее. Этот метод работает на двух уровнях. Если вы глубоко понимаете данную идею, то сразу заметите упрощение и эмоционально на него отреагируете, что сохранит вашу заинтересованность. Но если вы незнакомы с идеей, упрощение послужит легким толчком, который подготовит вас к более глубокому описанию.

Чего мы хотим от вас. Если вы отметили чрезмерное упрощение, то не надо дистанцироваться от этого или думать, что мы упустили из виду основную идею, приукрасили действительность или забыли нечто важное. Скорее всего, то, что вас насторожило, будет разъяснено в книге позже. Можете считать упрощенное введение сложной идеи чем-то вроде неформального приветствия: это ободряет незнакомых с идеей читателей, они чувствуют, что находятся на верном пути, а это закладывает основы для более глубокого понимания.

РАЗГОВОРНЫЙ (ПРИВЫЧНЫЙ) СТИЛЬ

На протяжении всей книги мы пользуемся разговорным языком, чтобы сделать излагаемый материал максимально привлекательным. Мы не забываем о юморе и время от времени — ссылках на литературные источники, а порой обращаемся непосредственно к вам или к себе, используя личные местоимения. На самом деле это научно обоснованно: когнитивные исследования* показали, что мозг запоминает больше информации, если во время разговора вы испытываете чувства.

Чего мы хотим от вас. Обычно людям нравится разговорный стиль, но не всем. Некоторые терпеть не могут сокращений. Для других разговорный стиль — признак недостаточной авторитетности книги. Мы всё понимаем, но поверьте, вы к этому привыкнете даже раньше, чем думаете.

---

* Узнать больше о том, как разговорный стиль помогает в обучении, можно в книге Clark, Ruth, Mayer, Richard. E-Learning and the Science of Instruction. Wiley, 2011.

КЛЮЧЕВЫЕ МОМЕНТЫ

Обозначаются значком  . В каждой главе мы будем резюмировать основные положения, которые были введены. Это поможет убедиться, что вы все усвоили, не пропустили ничего важного. Кроме того, мозг получит краткий отдых после обучения.

Чего мы хотим от вас. Не закрывайте разделы «Ключевые моменты» сразу. Уделите им хотя бы минутку. Вы помните все, что в них описано? Если нет, то не поленитесь вернуться на несколько страниц назад, чтобы освежить память.

ЧАСТО ЗАДАВАЕМЫЕ ВОПРОСЫ

Обозначаются значком  . Основную часть времени мы работаем в командах программистов, создавая реальное ПО для реальных пользователей, но при этом немало лет посвятили чтению лекций и проведению презентаций на тему гибкой разработки. В ходе этих мероприятий мы беседовали со многими людьми и можем утверждать, что некоторые вопросы интересуют людей особенно часто.

Чего мы хотим от вас. Прочтите в конце каждой главы раздел «Часто задаваемые вопросы». Есть ли среди них тот, который волнует именно вас? Если да, то удовлетворяет ли вас ответ? Не все ответы подходят к вашему случаю, но постарайтесь и в них найти рациональное зерно. Если все вопросы в разделе неактуальны для вас, то подумайте, с какой целью их могли задать. Так можно по-новому взглянуть на материал (в главе 2 вы узнаете, почему это важно для команды).

ЧТО ВЫ МОЖЕТЕ СДЕЛАТЬ УЖЕ СЕГОДНЯ

Обозначаются значком  . Самый эффективный способ чему-то научиться — сделать это! В конце каждой главы мы приводим небольшой раздел, в котором есть список того, что можно сделать прямо сейчас — как самостоятельно, так и вместе с командой.

Чего мы хотим от вас. Разумеется, самое лучшее — это просто начать делать! Но не всем это по душе, а один из главных постулатов книги состоит в том, что попытки внедрять практику в условиях, когда мировоззрение компании не совпадает с ней, могут окончиться плохо. Так что прежде всего подумайте, как отреагирует команда. Это может быть не менее эффективным инструментом обучения, чем само выполнение задания.

ГДЕ ВЫ МОЖЕТЕ УЗНАТЬ БОЛЬШЕ

Обозначаются значком  . Исаак Ньютон однажды сказал: «Если я видел дальше других, то потому, что стоял на плечах гигантов». Нам повезло: сейчас существует множество революционных книг по гибкой разработке ПО. После каждой главы мы предлагаем список из нескольких источников, в которых можно найти больше сведений по конкретной теме.

Чего мы хотим от вас. Продолжайте учиться! Наша книга подробно рассказывает о Scrum, XP, Lean и Канбан, но, конечно, мы не можем исследовать все эти идеи детально. Большинство идей, описанных здесь, придумано не нами. К счастью, вы можете учиться и у тех, кому они принадлежат.

ПОДСКАЗКИ

Обозначаются значком  . Agile-коуч — это человек, который помогает командам овладеть Agile. Книга написана для тех, кто только обучается Agile, но ее могут использовать и опытные agile-коучи, внедряющие эти идеи в свою команду. Ищите советы для тренеров в конце каждой главы. Они помогут применить идеи и подход, которые используем мы, и адаптировать их к вашей команде.

Чего мы хотим от вас. Даже если вы не коуч, все равно стоит прочитать советы для тренеров. Дело в том, что один из самых эффективных методов обучения — попробовать себя в роли наставника. Если вы узнаете об этих идеях впервые, то подумайте, как можно использовать советы для тренеров, чтобы помочь команде больше узнать об Agile.

Структура книги

Эта книга устроена так, чтобы помочь вам понять agile-методологии, усвоив ценности и принципы эффективной команды разработки, школ, которые воплощают эти ценности, и методик, при помощи которых они реализуются.

Две следующие главы помогут понять ценности и принципы, которые будут способствовать переходу на гибкое мировоззрение. В них приведены способы, благодаря которым можно оценить, готовы ли вы и ваша команда принять agile-методологии, какие именно ее части найдут отклик у коллег и что может оказаться наиболее сложным для внедрения.

— Глава 2 описывает ключевые ценности Agile. Мы расскажем о команде, бьющейся над программным проектом, и объясним, что основной

источник затруднений — это «искаженная перспектива». Мы изложим ценности Agile и при помощи метафоры поможем увидеть, как они дают возможность команде увидеть общую перспективу.

— Глава 3 рассказывает о принципах, в соответствии с которыми agile-команды принимают решения о том, как управлять проектами. Мы поясним, какие цели и идеи лежат в основе этих принципов, проиллюстрировав их практическим примером из программного проекта.

В следующих шести главах говорится о самых популярных школах Agile: Scrum, XP, Lean и Канбан. Вы узнаете, как их применять и внедрить в практику работы вашей команды.

— Глава 4 описывает Scrum, популярный agile-подход, чтобы рассказать о том, как работают самоорганизующиеся команды. Мы дадим несколько советов, как применить технологию Scrum к вашим проектам и обучить команду самоорганизации.

— Глава 5 демонстрирует конкретные процедуры, которые используются в scrum-командах для управления проектами, и объясняет, как эти процедуры помогают команде объединиться и создать качественные программы. Мы покажем, что успех реального перехода на Scrum зависит от того, насколько полно ценности Scrum соответствуют культуре вашей команды и компании, и поясним, что делать, если различия слишком сильны.

— Глава 6 рассказывает об основных методах экстремального программирования, его ценностях и принципах. Вы узнаете, как каждому члену команды прививается мировоззрение, необходимое для улучшения работы над кодом: вместо того чтобы ненавидеть перемены, все сотрудники учатся принимать их с готовностью.

— Глава 7 рассказывает о трех основных методах экстремального программирования и о том, как они помогают избежать серьезных проблем с кодом и проектированием. Вы поймете, что все методы экстремального программирования образуют единую экосистему, которая ведет к созданию лучшего кода — более гибкого, изменяемого и простого в обслуживании.

— Глава 8 знакомит с бережливым программированием и принципами, которые помогут обрести соответствующее мировоззрение. И мы покажем, что методы размышлений, предлагаемые бережливым программированием, могут помочь вашей команде найти излишне предпринимаемые действия и избавиться от них, а также увидеть общую картину системы, в рамках которой вы разрабатываете ПО.

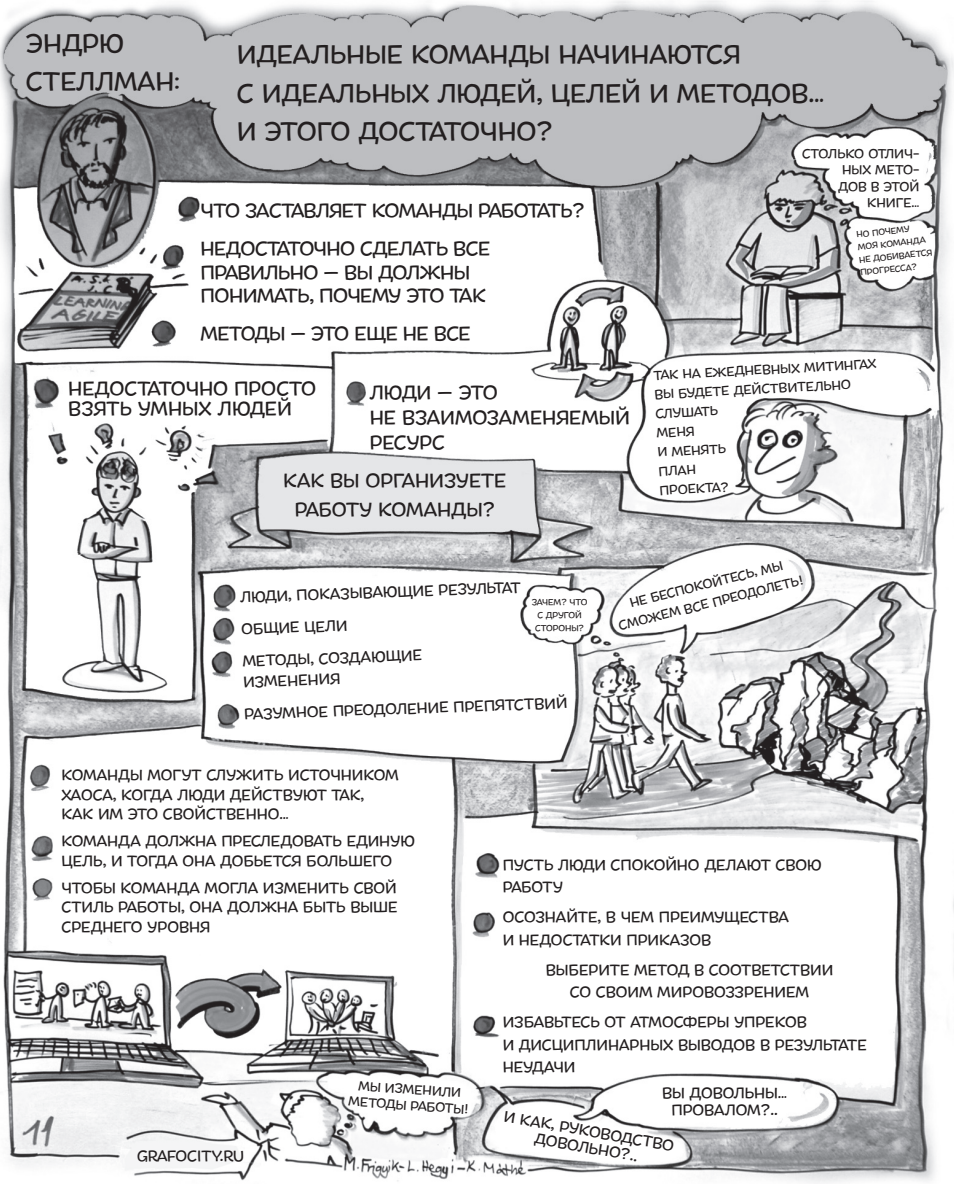

Рис. 1.4. Графическая запись выступления Эндрю Стеллмана на конференции Stretch 2013 в Будапеште (Венгрия). Выступление было основано на материалах этой главы Графическая запись: Ката Мате и Марти Фридик, www.remarker.eu

— Глава 9 рассказывает о Канбане, его принципах и взаимоотношениях с бережливым программированием, а также о методах. Вы узнаете, как концентрация на потоке и теории массового обслуживания поможет вашей команде претворить в жизнь идеалы бережливого программирования. Также вы поймете, как Канбан может создать в команде культуру постоянного совершенствования.

В мире Agile существуют не только мировоззрения, методологии и школы мышления. Компании все чаще полагаются на agile-коучей, способных помочь командам взять Agile на вооружение. Вот почему мы включили в книгу последнюю главу.

— Глава 10 рассказывает о работе agile-коучей: как учатся команды, как коуч помогает изменить мировоззрение, чтобы легче было взять на вооружение agile-методологии и стать более гибкими.