

Содержание

Предисловие Кена Швабера	7
Предисловие	9
Глава 1. Аджайловая парадигма	13
1.1. Меняться или не меняться	13
1.2. Происхождение аджайла.....	21
1.3. Определение аджайла.....	24
1.4. Итеративно-инкрементальный континуум.....	28
1.5. Гибкость нельзя спланировать	33
1.6. Сочетание аджайла и бережливого производства.....	38
Глава 2. Скрам	53
2.1. Дом Скрама	53
2.2. Скрам, что в имени твоём?	55
2.3. Что я там вижу – гориллу?.....	59
2.4. Фреймворк, а не методология	66
2.5. Как играть	69
2.6. Основные принципы скрама	92
2.7. Ценности скрама.....	107

Глава 3. Тактики достижения цели	115
3.1. Визуализация продвижения в работе	116
3.2. Вопросы ежедневного скрама	119
3.3. Уточнение бэклога продукта	121
3.4. Пользовательские истории	122
3.5. Покер планирования	124
3.6. Длина спринта	126
3.7. Как масштабировать скрам	129
Глава 4. Будущее скрама	139
4.1. Да, мы работаем по скраму. И.....	139
4.2. Мощь потенциального продукта	142
4.3. Принятие скрама топ-менеджментом.....	146
Словарь	153
Об авторе	158

Предисловие Кена Швабера

Эта книга написана чрезвычайно компетентно. Гюнтер рассказал о скраме, используя хорошо структурированные определения, наполненные инсайтами. Видно, что он понимает саму суть скрама. В то же время эти инсайты никогда не ошарашивают. Вы понимаете, для чего они нужны в книге, и думаете: «Это было очень-очень полезно. Я нашел именно то, что мне нужно знать, и понял именно то, что хотел понять, и меня не отвлекала лишняя информация».

Мне было нелегко писать это предисловие: кажется, что оно должно быть написано так же хорошо, как и сама книга. А этого добиться очень трудно. Прочитайте книгу Гюнтера, прочитайте только одну часть или всю книгу целиком — вам понравится.

Скрам прост, но его достаточно, чтобы грамотно решать комплексные задачи. А этой книги достаточно, чтобы понять простой фреймворк для решения комплексных задач.

Кен, август 2013 года

Предисловие

Интерес к использованию гибких методов растет, при этом скрам — самый распространенный подход для реализации аджайл-манифеста. Общий интерес к скраму уже достаточно велик и продолжает расти как в сфере разработки софта, так и в других сферах.

Изменение подхода к разработке софта — непростой вызов для организаций. Скрам — это не готовый рецепт с детальными инструкциями для любой возможной ситуации. Скрам — это *фреймворк* принципов, ролей и правил, который хорошо работает благодаря *людям*, использующим скрам. Подлинный потенциал скрама заключается в открытии, *развитии* и оптимизации практик, инструментов и техник для каждой конкретной ситуации.

Успешное применение скрама зависит от желания устранить барьеры, мысленно выходить за пределы организационных стен и препятствий и готовности пуститься в путешествие за открытиями. Скрам — это больше про поведение, нежели про процесс.

Путешествие начинается с понимания правил игры в скрам. Эта книга будет вашим попутчиком на протяжении всего путешествия. Она покажет, как скрам воплощает аджайл-мышление, каковы правила игры в скрам и как они помогают разнообразить тактики внутри игры. Эта книга подходит для сотрудников, команд, менеджеров и агентов изменений, независимо от того, применяют ли они уже скрам или только собираются отправиться в это путешествие.

Мое путешествие стартовало в 2003 году: мой путь к бизнес-гибкости начался с экстремального программирования и скрама. Этот путь был сложным, но по-другому быть и не могло. За время моего путешествия я использовал скрам со множеством команд, на разнообразных проектах, в разных организациях. Я работал в больших и маленьких организациях и тренировал как команды, так и высшее руководство. Я написал первое издание этой книги. Мне посчастливилось сотрудничать с Кеном Швабером, одним из создателей скрама в Scrum.org.

Кто бы мог предположить, что спустя пять лет после почти случайного создания первого издания в 2013 году возникнет потребность во втором издании моей книги?

Я помню, как описал ценности скрама в первом издании, а в июле 2016 года они были добавлены в «Руководство

по скраму». Я охарактеризовал традиционные три вопроса как хорошую, но необязательную практику, которую можно использовать на ежедневном скраме, и это тоже было добавлено к руководству в ноябре 2017 года.

Однако с 2013 года возникло еще больше вызовов и они стали еще более сложными. Баланс форм труда в обществе продолжает быстро сдвигаться от индустриальных (часто физических) к цифровым (часто виртуальным). Во многих отраслях непредсказуемость в сфере труда продолжает увеличиваться. Индустриальная парадигма становится бесполезной. Сейчас, более чем когда-либо, необходимы гибкая парадигма и основательный фреймворк, который помогает людям и организациям увеличивать бизнес-гибкость для сложной работы в непредсказуемых обстоятельствах.

Скрам — это *простой фреймворк, который помогает отвечать на комплексные вызовы*, а не только способ создания сложных софтверных продуктов. Все больше людей хотят понять, как применять скрам в областях, которые находятся за пределами разработки софта. Организации перестраивают свои структуры и бизнес-процессы для применения скрама и пытаются четко понимать его правила. Это требует более общего рассказа о скраме, других слов, других углов зрения на его

правила. Сейчас идет третья волна скрама, и, если вы хотите ее поймать, эта книга вам поможет. Это, второе, издание рассказывает об основах скрама, чтобы вы смогли научиться его использовать.

Я благодарю Кена Швабера за предисловие и рецензирование первого издания, так же как и других рецензентов: Дэвида Старра, Патрисию Конг и Ральфа Джохамма — за обратную связь по первому изданию. Я очень признателен Блейк МакМиллан и Доминик Максимиани за рецензирование второго издания. Я благодарю всех переводчиков за их труды по распространению моей книги на разных языках.

Я благодарю издательство Van Haren Publishing и в особенности Иво ван Харрена за то, что мне был дан шанс рассказать о своем видении скрама в этой книге.

Наслаждайтесь чтением и... продолжайте скрамить.

*Гюнтер,
август 2018 г.*

Глава 1

Аджайловая парадигма

1.1 МЕНЯТЬСЯ ИЛИ НЕ МЕНЯТЬСЯ

В индустрии разработки ПО долгое время доминировала парадигма индустриальных взглядов (рис. 1.1). Фактически она была калькой со старых добрых промышленных практик и теорий. Важной частью фундамента этих практик было тейлористское убеждение¹ о том, что «рабочим» нельзя доверять осмысленную и креативную работу. От них можно ждать выполнения только механических задач. Вся их работа должна быть подготовлена, спроектирована и спланирована более старшими по иерархии сотрудниками. Более того, старшие должны

¹ Фредерик Тейлор (1856–1915) — американский инженер, который известен прежде всего исследованиями в области оптимизации производительности, эффективности и стоимости труда. Он пропагандировал внедрение стандартизации и применение системных методов и практик. Контроль выполнялся исключительно менеджментом, тогда как рабочие, по его мнению, могли выполнять только непосредственно свою работу. — *Здесь и далее, за исключением специально оговоренных случаев, примечания автора.*



Рис. 1.1. Индустриальная парадигма

неусыпно надзирать за исполнением этих тщательно подготовленных задач.

Качество работы в индустриальной парадигме удостоверяется приемкой хороших и отбраковыванием плохих партий изделий. Денежное вознаграждение используется, чтобы стимулировать желаемое поведение. Нежелательное поведение наказывается. Используются старые добрые стратегии кнута и пряника.

Серьезные недостатки этой парадигмы уже известны и тщательно описаны. В частности, Отчет о состоянии ИТ-проектов от Standish Group² раз за разом выявляет

² Standish, 2011; Standish, 2013.

низкий уровень успешности традиционной разработки ПО. Количество проблем и ошибок в ПО, которое разрабатывали традиционным способом, превышало все возможные пределы. Столь обескураживающие результаты повлияли на ожидания от разработки в целом. Считалось нормальным, что только 10–20% проектов успешны. Успех в индустриальной парадигме определяется выполнением трех требований: в заданное время, в рамках бюджета и в запланированном объеме. *Хотя можно спорить с этими критериями успеха, это то, что нам обещает индустриальная парадигма.* Стало общепринятым, что качество остается низким и более половины функциональности софта³ никогда не используется⁴.

Мы до конца не осознаем, однако именно индустриальная парадигма привела разработку софта к серьезному кризису. Многие пытались преодолеть его, усиливая индустриальный подход. Они создавали больше планов,

³ Эти цифры являются предметом дискуссии, см., например <https://www.mountangoatsoftware.com/blog/are-64-of-features-really-rarely-or-never-used> и <https://scrumcrazy.wordpress.com/2015/08/06/a-response-to-mike-cohns-comments-on-64-of-software-features-rarely-or-never-used/>. В моей личной практике бывали примеры создания вовсе не нужного софта. — *Прим. пер.*

⁴ Standish, 2002; Standish, 2013.

планировали больше фаз, выполняли больше подготовительной работы в надежде на то, что разработка пройдет более эффективно. Подготовительные работы старались сделать все более исчерпывающими. Базовая идея оставалась той же — «рабочими» нужно управлять с помощью подробных инструкций. Надзор усиливался и становился все более интенсивным. *Если степень успешности не увеличивалась, индустриальная парадигма предполагала, что инструкции недостаточно ясные и детальные.*

И все же улучшений было мало. Приходилось мириться со множеством недоработок, дефектов и низким качеством ПО.

Прошло некоторое время, и на базе наблюдений за значительной неэффективностью индустриальной парадигмы начали появляться новые идеи.

Семена нового мировоззрения были посеяны уже в 1990-е годы. Проросли они в 2001 году, когда появилось формальное наименование «аджайл» — это событие стало новой точкой отсчета в истории разработки софта. Новая парадигма родилась в софтверной индустрии, но почти сразу стала распространяться на другие сферы общества. В ее основе креативность, уважение творческой природы работы и интеллигентности «рабочих».



Рис. 1.2. Аджайловая парадигма

У индустрии разработки ПО есть весомые причины продолжать двигаться в сторону аджайловой парадигмы: существующие проблемы серьезны и широко известны, а использование софта растет экспоненциально, делая ПО критически важным аспектом современного мира. Однако переход к новой парадигме требует времени. И кажется, что старая парадигма глубоко укоренилась: индустриальный подход к разработке софта продолжает преподаваться в учебных заведениях и пропагандируется как наиболее приемлемый.

Многие говорят, что аджайл слишком радикален, поэтому они выступают за постепенное внедрение гибких практик в существующие, традиционные рамки. Однако есть причина скептически относиться к постепенной эволюции, медленному продвижению от старой парадигмы к новой, от водопада к аджайлу.

Высока вероятность, что постепенная эволюция никогда не выйдет за рамки поверхностных изменений. На словах все будет по-новому, на бумаге будут введены новые практики, но мировоззрение и поведение людей и организаций останется неизменным. Существенные недостатки останутся; в частности, сохранится неуважение к людям — креативных и интеллигентных людей будут по-прежнему относить к безмозглым «рабочим» и «ресурсам».

Из-за того, что сохранятся основы, останутся нетронутыми и существующие метрики и стандарты, и новая парадигма будет измеряться по этим старым метрикам. Различные парадигмы зачастую базируются на разных, часто взаимоисключающих, концепциях и идеях, поэтому сравнивать эти две парадигмы не имеет никакого смысла. Надо честно признать серьезные недостатки старых подходов. Требуется лидерство, видение, предпринимательский дух и настойчивость, чтобы переключиться на новые подходы, изменив старый образ мыслей.

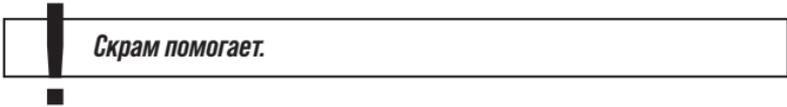
Постепенные изменения — это фактически сохранение статус-кво, то есть индустриальная парадигма остается в неизменном виде.

Существует огромное количество доказательств того, что старая парадигма не работает. Но раньше доказательства преимуществ аджайла были единичными историями. Отчет о состоянии ИТ-проектов⁵ в 2011 году, подготовленный Standish Group, — это поворотная точка. Он впервые продемонстрировал однозначные результаты исследований, которые были подтверждены во всех последующих отчетах. Было проведено масштабное исследование/сравнение традиционных проектов с проектами, которые использовали гибкие методы. Отчет показал, что гибкий подход к разработке софта дает значительно более высокие результаты даже в рамках традиционных ожиданий, когда софт должен быть поставлен в заявленные сроки, в рамках бюджета и во всем обещанном объеме функциональности. Отчет показал, что аджайловые проекты были в три раза более успешными, а провальных аджайловых проектов было в три раза меньше, чем традиционных. Для больших проектов, однако, разница в степени успешности была не такой

⁵ Chaos Manifesto (The Laws of Chaos and the Chaos 100 Best PM Practices). The Standish Group International, 2011.

явной, что может быть связано с неправильными ожиданиями, т. е. с комбинацией время + бюджет + объем. Ясно, что при правильно сформированных ожиданиях с фокусом на активном сотрудничестве с заказчиком и частой поставке *ценности* новая парадигма будет работать еще лучше, преодолевая проблему объема за счет частой поставки срезов *ценности*.

Тем не менее аджайл — это выбор, а не требование. Это один из путей улучшения индустрии разработки софта. Исследование показывает, что этот путь более успешен.



Скрам помогает.

Четкие правила скрама позволяют понять новую парадигму. Небольшая система предписаний помогает начать действовать немедленно и приводит к более быстрому и глубокому усвоению новой парадигмы. Скрам — это действенный способ адаптации аджайл-парадигмы. Используя скрам, люди начинают работать по-новому — через открытие, обучение, основанное на экспериментах и сотрудничестве. Они переходят в новое состояние: состояние *гибкости*, состояние постоянного изменения, движения, эволюции и улучшения. Этот процесс помогает их организациям трансформироваться

в такое *состояние* бизнес-гибкости, которое высвобождает время, людей и энергию, помогая сохранять инновационность.

Несмотря на положительные стороны, переход к скраму — это огромное изменение. Трудности возникают из-за неопределенности, которую рождают перемены, из-за того, что приходится расставаться со старыми устоями, даже если они уже не слишком надежны. Необходимо время, решимость и тяжелый труд для существенных изменений. Скрам простой, но не легкий.

1.2 ПРОИСХОЖДЕНИЕ АДЖАЙЛА

Несмотря на доминирование индустриальных взглядов, в которых главное — тщательное планирование, эволюционный подход к разработке ПО совсем не нов. Крэйг Ларман подробно описал предшественников аджайла в своей книге «Аджайл и итеративная разработка» (Agile & Iterative Development).

Но официальное название «аджайл» родилось в начале 2001 года, когда 17 лидеров по разработке программного обеспечения собрались на горнолыжном курорте Сноубёрд в штате Юта. Они обсуждали свои взгляды на разработку ПО в то время, когда неработающий

водопадный подход был заменен тяжеловесной методологией RUP⁶, но и она не привела к лучшим результатам по сравнению с традиционными процессами. Лидеры разработки шли различными путями и использовали разные методы, каждый из которых был воплощением новой парадигмы: скрам, экстремальное программирование, адаптивная разработка ПО, Crystal, разработка, управляемая функциональностью (Feature driven development), метод разработки динамических систем (DSDM) и другие.

В результате этой встречи было решено объединить общие принципы, убеждения и методы под одним названием «аджайл». Они были опубликованы как «Аджайл-манифест».

Слишком часто слышно о желании использовать аджайл. И слишком часто это желание подразумевает поиск волшебного решения, еще одного универсального решения для всех проблем. Именно поэтому я утверждаю, что *«аджайла не существует»*. Аджайл — это не какой-то фиксированный процесс, метод или практика. Аджайл — это совокупность общих принципов, которые свойственны всем методам гибкой разработки

⁶ Методология разработки программного обеспечения, созданная компанией Rational Software.

ПО. Аджайл — это мышление, убеждения и предпочтения, выраженные в аджайл-манифесте.

Манифест помогает понять идеи, на которых основан аджайл. Если вы используете его как источник более глубокого понимания аджайла, я рекомендую посмотреть на 12 принципов⁷, которые стоят за формулировками четырех ценностей.

Мы постоянно открываем для себя более совершенные методы разработки программного обеспечения, занимаясь непосредственно разработкой и помогая в этом другим. Благодаря проделанной работе мы смогли осознать, что:

Люди и взаимодействие важнее процессов и инструментов.

Работающий продукт важнее исчерпывающей документации.

Сотрудничество с заказчиком важнее согласования условий контракта.

Готовность к изменениям важнее следования первоначальному плану.

То есть, не отрицая важности того, что справа,
мы все-таки больше ценим то, что слева.

Рис. 1.3. Аджайл-манифест

⁷ См. <http://agilemanifesto.org/principles.html>.

1.3 ОПРЕДЕЛЕНИЕ АДЖАЙЛА

В отсутствие четкого определения я предпочитаю описывать аджайл с помощью трех его ключевых характеристик. Эти черты общие для всех гибких методов и типичны для данного способа работы:

- движимый людьми;
- итеративно-инкрементальный процесс;
- мера успеха — ценность.

1.3.1 Движимый людьми

Аджайл — это не работа по плану, который описывает реализацию тщательно проанализированных, спроектированных и архитектурно разработанных требований. Аджайл признает, что требования изначально не могут быть предсказаны до мельчайших деталей.

В аджайле не бывает так, чтобы различные типы промежуточных результатов передавались в разные специализированные отделы и каждый из них работал отдельно от остальных.

Аджайл — это постоянное *сотрудничество* людей из всех необходимых отделов: ИТ, маркетинг, продажи, поддержка пользователей, и др. Аджайл, конечно,

не признает традиционные разногласия между бизнесом и ИТ. Они оба необходимы для создания пригодного к использованию, полезного, обладающего ценностью ПО.

Чтобы сотрудничество, взаимодействие и общение были эффективными, нужен другой стиль управления. Например, когда командам ставятся цели и указывается направление, а далее происходит самоуправление в определенных границах. Контроль происходит за счет установки границ.

Сотрудничество и фасилитация заменяют традиционные командно-административные механизмы ежедневного инструктирования людей, которые бездумно исполняют микрозадания, и тоталитарных руководителей, которые осуществляют всепроникающий контроль.

Аджайл — это когда людей уважают за их креативность, интеллигентность и способность к самоорганизации. Их уважают за готовность понимать и решать проблему без церемоний и бюрократии. Обилие церемоний лишь заменяет интеллектуальное сотрудничество, инновацию и ответственность людей на бюрократию, результаты на бумаге, перекладывание задач с одного ответственного на другого и административные отговорки.

Людей уважают за то, что они могут работать в *устойчивом темпе* бесконечно долго. Работа в аджайле организована без авралов, поэтому темп может быть примерно одинаковым, устойчивым, сколь угодно долго.

1.3.2 Итеративно-инкрементальный процесс

Аджайловые процессы — это не игра без правил. Эти процессы *определены* и требуют высокой *дисциплины*.

Продукты создаются слой за слоем, в этом заключается «инкрементальность» процесса: каждый новый слой создается из расширений, улучшений, удалений и модификаций. Уже созданные слои и продукт целиком часто пересматриваются, чтобы удостовериться в целостности продукта, — это «итеративность».

Аджайл требует от всех игроков большого внимания к качеству и мастерству. Аджайл не разделяет идею о том, что качества продукта могут быть воплощены в документах или описаниях на бумаге, потому что эти описанные качества никак не соотносятся с конечным результатом и готовым к выпуску продуктом.

Итеративно-инкрементальный процесс необходим, так как требования и реализация — независимо от того, сколько времени, энергии и денег потрачено на то, чтобы

предсказать их изначально, — подвержены изменениям. Назовем лишь несколько причин: рынки и конкуренты развиваются; пользователи начинают понимать, что им действительно нужно, только тогда, когда получают это в свои руки; корпоративные стратегии изменяются. Все это требует осознанности и открытости к изменениям.

В противоположность традиционному процессу, изменения не исключаются из гибкого процесса. Новые идеи, меняющиеся точки зрения и приоритеты — сердце аджайла. Суть аджайла в расширении, развитии и последующем постепенном изменении требований, планов, идей, архитектуры и дизайна. Изменение не разрушительно, потому что оно составляет естественную часть гибкого способа работы. Аджайл *поощряет* изменения, так как это источник инноваций и улучшения.

1.3.3 Мера успеха — ценность

Прогресс при разработке софта не может быть измерен или гарантирован простым соответствием установленным планам и вехам, документам, подписями, согласованиями и другими церемониальными обязательствами, как это бывает в индустриальной парадигме. Аджайл вводит новые способы измерения успеха и продвижения в работе.

В аджайле успех и степень продвижения при разработке софта могут быть определены только путем частых проверок *работающих версий софта* (а не его промежуточных описаний) и фактической *ценностью*, которую продукт представляет для использующих его людей.

Естественной составляющей продуктовой разработки является то, что люди становятся уверены в удобстве продукта и его пригодности только тогда, когда попробуют его в действии. Никакая письменная документация или виртуальный процесс не заменят этого. Поэтому так важен итеративно-инкрементальный процесс с регулярной обратной связью с пользователями: измерением воздействия продукта на них, их оценки. Циклы обратной связи выступают как источник входной информации для дальнейшего развития продукта.

1.4 ИТЕРАТИВНО-ИНКРЕМЕНТАЛЬНЫЙ КОНТИНУУМ

При гибком подходе время делится на интервалы — итерации, периоды с фиксированными датами начала и завершения. У техники *тайм-боксинга* есть много преимуществ, одно из главных — *фокус*. Благодаря этой технике тайм-менеджмента можно пережить поворотные

моменты или резкие изменения. Во время регулярных проверок учитываются извлеченные уроки и работа представляет собой непрерывный поток от итерации к итерации. Основная цель каждой итерации — создание обладающих ценностью работающих версий продукта не позднее чем в конце итерации и получение быстрой обратной связи.

В аджайле вся разработка организована таким образом, чтобы оптимизировать способность отвечать на *бизнес-вызовы* и использовать их.

Ценность — это ответ на требования пользователей, рынка и бизнеса, общая мера продвижения вперед и успеха. Ценность является внутренней гипотезой, предположением до тех пор, пока продукт не будет выпущен на рынок. Выпуск версий продукта на рынок — единственный способ проверить предположение относительно ценности продукта. Регулярный выпуск на рынок — единственный способ получить обратную связь и оценку со стороны рынка и изменить продукт в соответствии с ними. Это делается путем последующего развития продукта: ценность постоянно оптимизируется от итерации к итерации. Риск снижается с помощью регулярного выпуска работающих инкрементов продукта, выполненных на основе четко определенных стандартов разработки.

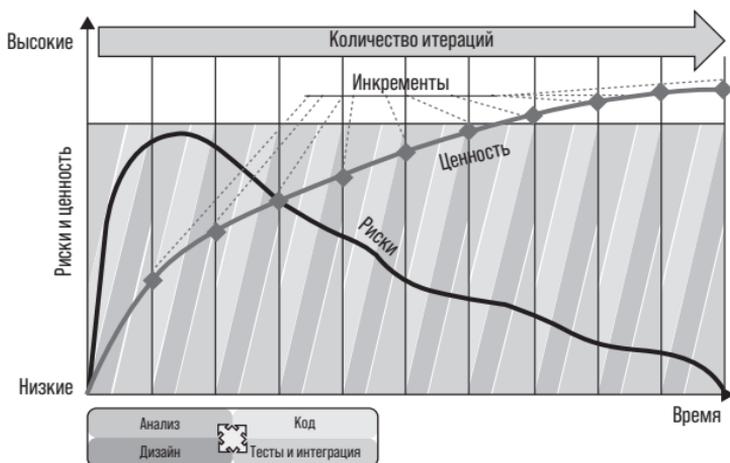


Рис. 1.4. Гибкая поставка ценности

Мы привыкли к тому, что риск в ИТ-контексте определяется как нечто техническое (*Будет ли система работать? Масштабируется ли она?*) и покрывает аспект пригодности для использования продукта (*Выдержит ли технически? Не сломается?*). Но этот взгляд на риск часто игнорирует конечную цель гибкой разработки — обеспечить бóльшую удовлетворенность конечных пользователей и клиентов, то есть продукты должны быть полезны. Продукт, пригодный для использования с технической точки зрения, — это только начало.

Риск также относится к бизнесу. Любой современный процесс разработки должен учитывать риск того, что вы

не сможете извлечь выгоду из неизвестных или непредвиденных возможностей рынка, риск невыпуска продукта достаточно быстро, риск разочарования клиентов, например при выпуске непротестированного софта, риск того, что пользователи не оценят функциональность продукта, риск проиграть конкурентам.

Гибкая разработка устроена так, чтобы максимально смягчить этот бизнес-риск. Требования, имеющие высокую ценность, удовлетворяются в первую очередь. Версии продукта и обновления выпускаются быстро и часто. Они удовлетворяют существующие потребности, а также предлагают неожиданные, инновационные функции. Благодаря этому пользователи платят за продукт, а инвестиции возвращаются. Версии продукта качественные, что позволяет минимизировать сопровождение и поддержку, оптимизировать общую стоимость владения (Total Cost of Ownership, TCO).

Аджайл признает значение «обычных» ИТ-активностей (на рис 1.4 они представлены как анализ, дизайн, кодирование и тестирование/интеграция), но ломает их последовательную организацию. Чтобы производить ПО с необходимой скоростью и получать выгоды быстрее, структура работы меняется. Цель в том, чтобы обеспечить гибкость и скорость, а не мешать им. В аджайле все эти действия осуществляются нелинейно, параллельно,

инкрементальным образом, разнопрофильными командами, которые постоянно сотрудничают и обмениваются идеями.

Цель такого интегрированного кросс-функционального подхода — изначально делать продукт качественно и предотвращать дефекты, а не пытаться создать качество за счет вылавливания ошибок в фазе постразработки. Обязательно не просто хотеть выпускать версии продукта часто, а действительно это делать. Качество не может быть добавлено к законченному продукту. Задержки и бюджет имеют тенденцию выходить из-под контроля, когда недостаток качества обнаруживается после того, как закончен процесс разработки.

Чтобы получить реальную и долговременную выгоду от гибких методов разработки, надо выходить за границы ИТ и других технических отделов. Придется меняться всей организации. Но это не только необходимость, это *возможность*. Всей организации пойдет на пользу использование аджайл-философии: работа короткими циклами с частыми промежуточными результатами и эволюционной адаптации. Аджайловые взгляды и подходы позволяют все большему количеству организаций и отделов наконец-то перестать пытаться предсказывать непредсказуемое. Аджайл включает в себя работу с ответами, решениями

и конкурирующими идеями, которые возникают *по ходу* создания продукта.

Нужно время, чтобы понять, что непрерывное обучение, присущее аджайлу, в действительности увеличивает контроль в беспокойных условиях. Нужно время, чтобы сместить фокус руководства с оценок на основе прошлого, например фактических показателей и регистраций времени работы на проекте. Нужно время, чтобы обрести уверенность за счет оптимизации и поставки бизнес-ценности через инкрементальные результаты гибкого процесса разработки.

Нужно время, чтобы принять, что бизнес-гибкость требует времени. Нужно время, чтобы принять, что гибкость не надо анализировать, проектировать и планировать до того, как начнется трансформация.

1.5 ГИБКОСТЬ НЕЛЬЗЯ СПЛАНИРОВАТЬ

Гибкость — это то, к чему стремятся при переходе к аджайловым способам работы. Гибкость — состояние постоянного потока, высокой отзывчивости, скорости и приспособляемости. Она необходима, чтобы справиться с меняющимися рынками и с неопределенностью, столь свойственной креативной деятельности, такой как разработка продукта.

Гибкость не имеет смысла, если упомянутые выше характеристики потока, отзывчивости, скорости и адаптивности не распространяются на взаимоотношения компании с рынками, на которых она работает, сообществами и потребителями. Использование аджайла — основа такой полномасштабной *бизнес-гибкости*. В результате внедрения аджайла появляются новые процессы, а также новая организационная культура обучения, совершенствования и постоянного приспособления и возрожденное уважение к людям. В ДНК организации внедряются и укореняются важные знания.

Есть несколько основополагающих истин для формирования правильных ожиданий от перехода к бизнес-гибкости. Внедрение аджайловых методов без принятия этих истин на самом деле будет препятствовать, а не способствовать гибкости.

- Гибкость нельзя спланировать.
- Гибкость нельзя продиктовать сверху.
- Гибкость нельзя скопировать.
- У гибкости нет конечного состояния.

Часто введение аджайловых методов рождает нереалистичные ожидания. Однако переход к новой парадигме вызывает значительные организационные потрясения, которые затронут существующие процедуры, отделы

и их функции. Эти изменения комплексные, и потому непредсказуемы. В процессе перехода к аджайлу невозможно предсказать, когда и с какими изменениями вы столкнетесь, как вы будете работать с этими изменениями, как будут восприниматься новые способы деятельности и каков будет результат. Будет невозможно планировать и контролировать следующие шаги. Невозможно предсказать скорость, с которой станут распространяться и укореняться изменения.

Гибкость сама по себе — это не просто новый процесс, это новое поведение. Это изменение *культуры*. Решение двигаться в сторону аджайла — это решение оставить прежние индустриальные методы работы. Надо не просто принять, а оценить по достоинству тот факт, что гибкость — это искусство возможностей. Требуется мужество, честность и решимость, чтобы действовать в соответствии с реальностью, которая раскрывается с помощью итеративно-инкрементальной информации о прогрессе. Быть гибким — это делать лучшее из возможного в каждый конкретный момент, в рамках ограниченных ресурсов и сталкиваясь с новыми требованиями. Планирование внедрения аджайловой трансформации игнорирует саму сущность аджайла, которая заключается в том, чтобы справляться со сложностью посредством хорошо продуманных экспериментов, обучения