НАУКА DEVOPS

УСКОРЯЙСЯ!

Как создавать и масштабировать
высокопроизводительные цифровые организации

Перевод с английского

Никколь Форсгрен,
Джез Хамбл, Джин Ким

альпина
ПАБЛИШЕР
Москва
2020

Эта книга — своего рода предвидение, в котором так отчаянно нуждаются генеральные директора, финансовые директора и ИТ-директора компаний, если они собираются выжить в этом новом программно ориентированном мире.

Любой, кто не читает эту книгу, будет заменен тем, кто ее читал.

Томас А. Лимончелли, соавтор книги The Practice of Cloud System Administration

«Вот, сделай это!» Данные, представленные в книге «Ускоряйся! Наука DevOps», являются триумфом исследований, упорства и проницательности, которые доказывают не только зависимость, но и причинно-следственную связь между грамотным техническим и управленческим поведением и эффективностью бизнеса. Они также разоблачают миф о «моделях зрелости» и предлагают реалистичную, действенную альтернативу. Как независимый консультант в сфере, где пересекаются люди, технологии, процессы и организационное проектирование, я могу вам сказать: это манна небесная!

Как следует из Главы 3, вы можете проложить свой путь к лучшей организационной культуре, внедряя эти практики в технологических компаниях». Нет никакой мистической культурной магии, есть только 24 конкретных специфических возможности, которые приведут не только к лучшим результатам в бизнесе, но и, что более важно, к более счастливым, здоровым и мотивированным людям и к формированию организации, в которой люди хотят работать. Я буду раздавать копии этой книги всем своим клиентам.

Дэн Норт, независимый консультант по технологиям и организации

Независимо от того, признают они это или нет, деятельность большинства организаций сегодня так или иначе связана с разработкой программного обеспечения. И большинство из них страдают от постоянных задержек из-за медленной разработки и внедрения, багов и ошибок при выпуске и сложного функционала, который увеличивает расходы и расстраивает пользователей. Так не должно быть. Авторы книги Форсгрен, Хамбл и Ким проливают свет на то, что такое DevOps, чтобы вы тоже могли испытать, как потрясающе это выглядит и ощущается на практике.

Карен Мартин, автор книг Clarity First и The Outstanding Organization

«Ускоряйся! Наука DevOps» выполняет фантастическую работу по объяснению не только того, какие изменения должны произвести организации для повышения эффективности доставки программного обеспечения, но и почему, тем самым позволяя людям на всех ступенях организации по-настоящему понять, как поднять свои компании на новый уровень.

Рин Дэниэлс, инженер по эксплуатации инфраструктуры компании Travis CI, автор книги Effective DevOps

Искусство строительства здания в настоящее время является хорошо изученной инженерной практикой. Тем не менее в мире программного обеспечения мы искали модели и методы, которые могут обеспечить те же предсказуемые и надежные результаты, сокращая отходы производства и обеспечивая все более высокую производительность, которой требует наш бизнес.

«Ускоряйся!» предоставляет научно обоснованные, поддающиеся количественной оценке и реально работающие принципы создания высокопроизводительных ИТ-команд мирового класса, позволяющих достичь удивительных результатов бизнеса.

Поддержанная двумя идейными лидерами сообщества DevOps (Ким и Хамбл) и исследованиями мирового класса от доктора наук (Форсгрен), эта книга настоятельно рекомендуется к прочтению!

Джонатан Флетчер, CTO группы компаний Hiscox

В своей книге «Ускоряйся!» Николь Форсгрен, Джез Хамбл и Джин Ким не разрушают никаких новых концептуальных основ в отношении Agile, Lean и DevOps. Вместо этого они представляют нечто, что может быть еще более ценным, — возможность заглянуть внутрь строгой методологии их подхода к сбору и анализу данных, который привел их к более ранним выводам о ключевых возможностях, увеличивающих вклад ИТ-компаний в бизнес. Я с удовольствием поставлю эту книгу на свою книжную полку рядом с другими великими работами.

Кэмерон Хайт, вице-президент и CTO компании VMware

Организации, которые будут процветать в будущем, будут использовать все преимущества цифровых технологий для улучшения своих коммерческих предложений и операций. «Ускоряйся» обобщает лучшие показатели, методы и принципы, которые можно использовать для улучшения доставки программного обеспечения и производительности цифровых продуктов на основе многолетних хорошо документированных исследований. Мы настоятельно рекомендуем эту книгу всем, кто занимается цифровой трансформацией, чтобы получить четкое представление о том, что работает, что не работает и что вообще не имеет значения.

Том и Мэри Поппендайк, авторы серии книг по бережливой разработке ПО (Lean Software Development)

Этой работой авторы внесли значительный вклад в понимание и применение DevOps. Они показывают, что при правильном понимании DevOps — это больше, чем просто причуда или новое имя для старой концепции. Их работа иллюстрирует, как DevOps может улучшить нынешнее положение дел в области организационного дизайна, культуры разработки программного обеспечения и архитектуры систем. И помимо простой демонстрации они продвигают качественные результаты сообщества DevOps с помощью основанных на исследованиях знаний, о которых я не слышал больше ни из одного источника.

Барон Шварц, основатель и CEO компании VividCortex и соавтор книги High Performance MySQL

Книга о том, как быть, а не казаться цифровой организацией

«Ускоряйся!» — говорит вам заголовок этой книги. И если вы рассчитываете устоять в эпоху фундаментальных трансформаций организации и менеджмента, которая наступила на наших глазах, вам придется последовать этому совету. Николь Форсгрен, Джез Хамбл и Джин Ким предлагают для этого несколько практических решений, пусть и не самых простых.

Эта работа — редкий образец глубокой и смелой аналитики, демонстрирующей свежий взгляд на подлинную природу вещей. На основе строгих данных о практике и эффективности DevOps авторы одними из первых объясняют сущность и базовые принципы цифровой трансформации организаций.

На фоне могучих сдвигов в экономике и повседневной жизни человека сформировалась особая культура разговора об изменениях. И как бы мы ни относились к концепции технологической сингулярности Рэймонда Курцвейла, она близка многим. Ведь мы все так или иначе пытаемся осмыслить сегодняшнее время, когда радикальное усложнение мира идет одновременно со взрывным ускорением развития.

И все же мы стремимся к упрощению. Мы признаем, что мир становится иным: более сложным, динамичным, устроенным на новых принципах, но, описывая его, интуитивно опираемся на представления и концепции из своего прошлого. А это часто уже не работает.

В результате нас окружают мемы и лозунги. Они внушают нам ложную уверенность в том, что мы якобы понимаем происходящее и можем принимать адекватные решения. В последнее время по понятным причинам такими мемами часто становятся технологические термины.

Вначале была Big Data, обещавшая решение если не всех, то многих проблем и радикальный рост эффективности. От нас требовалось лишь собрать на порядок больше данных и правильно их обработать. Потом возник интернет вещей. Казалось, если оснастить датчиками всевозможные предметы, это точно сработает. Затем пришла очередь Agile, роботизации, искусственного интеллекта… И наконец, появилась цифровая трансформация.

Этот термин сразу полюбили аналитики и журналисты. Он позволяет опустить детали и одним махом обозначить нашу эпоху, дать символ исторических изменений, которые несут с собой технологии.

Но цифровая трансформация — это сложное явление, которое по своей контринтуитивности иногда напоминает теорию относительности. Даже людям с математическим образованием сложно абстрактно представить, что время, воспринимаемое нами совершенно однозначно, в разных

точках пространства может течь по-разному. Тем не менее на этом сегодня базируется наше понимание устройства мира. Так и здесь, разбираясь с цифровой трансформацией, мы вынуждены опираться на концепции, подходы и инструменты, многие из которых непонятны, противоречат нашему прошлому опыту и слабо отработаны на практике.

Для авторов книги цифровая трансформация — это вполне конкретный процесс перестройки организации. В его основе лежит запуск оптимального механизма создания и обновления программного обеспечения.

Казалось бы, это узкий технологический вопрос, но это совсем не так. Дело в том, что любая современная организация, будь то банк, государственная служба или производитель автомобилей, — это большая цифровая платформа со сложной архитектурой, состоящей из массы собственных или сторонних программных и инфраструктурных решений. Эта платформа больше не поддерживает ваш бизнес. Сегодня она и есть ваш бизнес.

Работа Форсгрен, Хамбл и Ким скрупулезно прослеживает, как связаны технологическая платформа, ее процессы и архитектуры с конкретными параметрами эффективности бизнеса и организации. Мы много слышали о том, что бизнесу пора принимать форму цифровых или ИТ-компаний. Книга «Ускоряйся!» — одна из первых, где показано, как это сделать практически. По сути, это готовый учебник по трансформации. Книга поможет справиться со сложными задачами управления и станет источником аргументов для дискуссий любого уровня — от рядового сотрудника до топ-менеджера и акционера.

Эта книга подводит важный промежуточный итог исследованиям эффективности гибких методологий. В ней есть практические метрики, инструменты, принципы и просто адекватные термины из сферы Agile, Lean и DevOps. Они необходимы каждому, кто включен в проекты по цифровизации, управлению изменениями, созданию или модернизации технологических платформ.

И наконец, авторы на практическом уровне обсуждают фундаментальные вопросы о человеке, лидерстве и культуре в цифровой организации. Пожалуй, это наиболее важное достижение книги. Мы интуитивно считаем, что цифровая трансформация принадлежит сфере технологий. Но это не так. В первую очередь она явление социальное, из сферы коммуникаций и взаимодействия между людьми.

Дело не в том, что культуры ИТ-компаний становятся доминирующими в корпоративной среде. Важно, что технологическая платформа — ее циклы развития, потребности, архитектура, эффективность — предопределяет и организационную структуру бизнеса, и профиль сотрудника с точки зрения его компетенций и ценностей.

Но что, если цифровая платформа устарела, перегружена и не справляется с растущим потоком данных? Именно это и происходит сегодня в большинстве российских компаний. Старые ИТ-системы с трудом и медленно меняются, слабо защищают от киберугроз и санкционных рисков. Вместо того чтобы стать основой трансформации бизнеса, они тормозят его развитие.

На мой взгляд, единственный способ преодолеть технологические ограничения сегодняшнего дня — создавать принципиально новые ИТ-платформы. Их основой должны стать достижения цифровой эпохи: передовые архитектуры, революционные технологии, прогрессивные подходы к управлению. Я уверен, что книга «Ускоряйся!» будет полезна каждому, кто твердо решил встать на путь к свободе цифрового развития.

Сергей Мацоцкий,
основатель и член совета директоров IBS

Предисловие Мартина Фаулера

Несколько лет назад я прочитал отчет, в котором говорилось: «Теперь мы можем с уверенностью утверждать, что высокая эффективность ИТ коррелирует с высокой эффективностью бизнеса, помогая увеличить производительность, прибыльность и долю рынка». Когда я читаю что-то подобное, моя первая реакция — со всей силы швырнуть это в мусорное ведро, потому что такие слова обычно говорят о какой-то высосанной из пальца ерунде, маскирующейся под науку. Однако на этот раз я заколебался, потому что у меня в руках был «Отчет о состоянии DevOps за 2014 год». Одним из его авторов был Джез Хамбл, мой коллега и друг, у которого, как я знал, тоже была аллергия на подобную чепуху. (Хотя я должен признаться, что еще одной причиной, по которой я не бросил его, было то, что я читал его на своем iPad.)

Вместо этого я отправил Джезу письмо по электронной почте, чтобы узнать, что стоит за этим заявлением. Несколько недель спустя я созвонился с ним и Николь Форсгрен, которая терпеливо ввела меня в курс дела. Хотя я не эксперт по методам, которые они использовали, она сказала достаточно, чтобы убедить меня, что здесь речь идет о реальном анализе, гораздо большем, чем я обычно вижу даже в научных работах. Я следил за последующими отчетами о состоянии DevOps с интересом, но и с растущим разочарованием. Отчеты представляли результаты их работы, но никогда не содержали тех подробных объяснений, которые Николь дала мне во время телефонного разговора. Это сильно подорвало к ним доверие, поскольку было мало доказательств того, что эти отчеты основаны на чем-то большем, чем просто гипотезы. Наконец, те из нас, кто уже «побывал за кулисами»,

убедили Николь, Джеза и Джина раскрыть свои методы, написав эту книгу. Для меня это было долгое ожидание, но теперь я рад, что у меня есть то, что я могу искренне рекомендовать как способ взглянуть на эффективность доставки ИТ-решений, основанный на чем-то большем, чем разрозненный опыт нескольких аналитиков.

Картина, которую они нарисовали в своей книге, просто неотразима. Они описывают, как компаниям с эффективными ИТ-доставками требуется около часа, чтобы довести код от «сохранения в магистраль» (committed to mainline) до «запуска в производство», — путь, который в некоторых организациях занимает месяцы. Таким образом они обновляют свое программное обеспечение много раз в день, а не один раз в несколько месяцев, увеличивая свою способность использовать программное обеспечение для изучения рынка, реагирования на события и выпуска новых функций быстрее, чем конкуренты. Такое огромное увеличение быстродействия не связано с затратами на стабильность, так как эти организации обнаруживают, что их обновления вызывают сбои, быстрее их менее эффективных коллег и эти сбои обычно фиксируются в течение часа. Их доказательства опровергают бимодальное представление о том, что вам нужно выбирать между скоростью и стабильностью, — напротив, скорость зависит от стабильности, поэтому хорошие ИТ-практики дают вам и то и другое.

Итак, как вы можете представить, я в восторге, что они запустили эту книгу в производство, и волей-неволей я буду рекомендовать ее в течение следующих нескольких лет. (Я уже использовал много битов из черновиков этой книги в своих выступлениях.) Тем не менее я хочу внести несколько предостережений. Авторы книги хорошо объясняют, почему их подход к опросам делает их прочной основой для их данных. Однако это все еще опросы, которые отражают субъективное восприятие. И мне интересно, как представленная ими выборка отражает ИТ-мир в целом. У меня будет больше уверенности в их результатах, когда другие команды, используя разные подходы, смогут подтвердить рассуждения авторов. В книге уже есть некоторые из таких подтверждений. Так, работа, проделанная Google по формированию командной культуры, дает дополнительные доказательства в поддержку суждения авторов о том, насколько важна организационная культура, основанная на модели Веструма, для эффективных команд, занятых разработкой ПО. Подобная дальнейшая работа также заставила бы меня меньше беспокоиться о том, чтобы их выводы подтверждали большую часть моих доводов при их отстаивании — подтверждение предвзятости является мощной силой (хотя я в основном замечаю это в других ;-)). Мы также должны помнить, что их книга фокусируется на доставке программных продуктов, то есть на пути от коммита[1] к запуску в производство, а не на всем процессе разработки программного обеспечения.

Но эти придирки не должны отвлекать нас от основного направления этой книги. Эти исследования и их тщательный анализ дают одно из лучших объяснений методов, которые могут значительно продвинуть вперед большинство ИТ-компаний. Каждый руководитель ИТ-группы должен внимательно изучить эти методы и работать над их использованием для улучшения своей деятельности. Любой, кто работает с ИТ-группой — либо внутри компании, либо из компании, которая занимается доставками ИТ (такой как наша), — должен искать возможности применить эти практики на месте и выработать устойчивую программу непрерывного совершенствования, чтобы развиваться вместе с ними. Форсгрен, Хамбл и Ким нарисовали картину того, как эффективно ИТ выглядит в 2017 году, и практикующие специалисты в сфере ИТ должны использовать ее как карту, чтобы присоединиться к высокоэффективным лидерам.

Мартин Фаулер, главный научный сотрудник компании ThoughtWorks

Предисловие Кортни Кисслер

Мой путь начался летом 2011 года. Я работала в Nordstrom, и мы приняли стратегическое решение сосредоточиться на цифровом секторе как двигателе роста. К этому моменту наша ИТ-компания прошла через оптимизацию расходов. В своей презентации на DevOps Enterprise Summit 2014 я поделилась информацией о том, что для меня одним из моментов прозрения был переход к оптимизации скорости. Я сделала много ошибок на этом пути, и хотела бы я тогда иметь доступ к этой книге и изложенной в ней информации. Я наступила на все классические грабли. Например, в попытке взять и внедрить Agile сверху вниз, думая, что он подходит всем, не фокусируясь на измерениях (или правильных параметрах измерений). При этом лидерское поведение не меняется и рассматривает трансформацию как программу вместо создания обучающейся организации (чего никогда не делалось).

На протяжении всего пути мы фокусировались на командных структурах, основанных на результатах: мы знали наше время цикла (понимали нашу карту ценностей), ограничивали «радиус взрыва» (начинали с 1–2 команд вместо того, чтобы пытаться «вскипятить океан»), использовали данные для управления действиями и решениями и признавали, что работа — это работа (цель — отсутствие недоработок и отставаний по функционалу, техническому обеспечению и операционной работе; вместо этого иметь только одно отставание, потому что NFRs (Nonfunctional Requirements — нефункциональные требования. — Прим. ред.) тоже являются функциями, а сокращение технических недоработок повышает стабильность продукта). Ничто из этого не было

достигнуто в одночасье, а процесс потребовал множества экспериментов и корректировок.

Основываясь на своем опыте, я знаю наверняка, что применение руководства из этой книги сделает вашу организацию более эффективной. Оно работает для всех типов доставки программного обеспечения и является независимой методологией. Я лично испытала это на себе, и у меня есть несколько примеров применения этих методов в командах программистов, традиционных командах доставки коробочных программных приложений и продуктовых командах. Это может работать по всем направлениям. Эти методы требуют дисциплины, настойчивости, трансформационного лидерства и концентрации на людях. В конце концов, люди — это актив № 1 любой организации, но зачастую это далеко от реальности.

Несмотря на то, что путь будет нелегким, я могу сказать, что он определенно того стоит, и вы не только увидите результаты, но и ваша команда станет счастливее. Например, когда мы начали измерять eNPS (employee Net Promoter Score — индекс чистой лояльности сотрудников. — Прим. ред.), команды, практикующие эти методы, получили самые высокие баллы во всей нашей технологической организации.

Еще одна вещь, которую я узнала по пути, — это то, насколько важно иметь поддержку высшего руководства. И поддержка должна выражаться в действиях, а не в словах. Высшее руководство должно продемонстрировать свою приверженность созданию обучающейся организации. Я поделюсь поведением, которое я пытаюсь моделировать с моими командами. Я страстно верю в необходимость четко представлять реальное положение вещей. Если я лидер и моя команда не чувствует себя комфортно, беря на себя риски, то я никогда не буду по-настоящему знать реальность. И если я не искренне заинтересована и появляюсь только тогда, когда случается неудача, то считайте, что я не состоялась как лидер. Важно построить доверие и продемонстрировать, что неудача приводит к обучению (см. модель Веструма в этой книге).

Вы столкнетесь со скептиками на этом пути. Я слышала такие вещи, как «DevOps — это новый Agile», «Lean не применяется к доставке программного обеспечения», «Конечно, это сработало для команды мобильных приложений. Они — единорог». Когда я столкнулась со скептиками, я попыталась использовать внешние примеры, чтобы повлиять на обсуждение. Я опиралась на поддержку наставников — без них мне было бы сложно сосредоточиться. Наличие информации из этой книги было бы мне тогда чрезвычайно полезно, и я настоятельно рекомендую вам использовать ее в своей организации. Я провела большую часть своей карьеры в розничной торговле, и в этой отрасли все более и более важной становилась способность меняться, а программное обеспечение для доставки теперь является частью ДНК каждой организации. Не игнорируйте науку, изложенную в этой книге.

Она поможет вам ускорить ваше превращение в высокоэффективную технологическую организацию.

Кортни Кисслер, вице-президент по разработке цифровых платформ компании Nike

Краткая справка: возможности управления оптимизацией

Наше исследование выявило 24 ключевые возможности, которые способствуют улучшению эффективности доставки программного обеспечения. Эта справка укажет вам на их расположение по ходу книги. Подробное руководство вы найдете в Приложении А. Возможности представлены в произвольном порядке.

Они подразделяются на пять категорий:

- непрерывная доставка;
- архитектура;
- продукт и процесс;
- бережливое управление и мониторинг;
- культурные возможности.

Возможности непрерывной доставки

1. Контроль версий: Глава 4.
2. Автоматизация развертывания: Глава 4.
3. Непрерывная интеграция: Глава 4.
4. Магистральная разработка: Глава 4.
5. Автоматизация тестирования: Глава 4.
6. Управление тестовыми данными: Глава 4.
7. «Сдвиг влево»2 по безопасности: Глава 6.
8. Непрерывная доставка (НД): Глава 4.

Возможности архитектуры

9. Слабосвязанная архитектура: Глава 5.
10. Уполномоченные команды: Глава 5.

Возможности продукта и процесса

11. Обратная связь от клиентов: Глава 8.
12. Поток создания ценности: Глава 8.
13. Работа небольшими партиями: Глава 8.
14. Командные эксперименты: Глава 8.

Возможности бережливого управления и мониторинга

15. Процесс утверждения изменений: Глава 7.
16. Мониторинг: Глава 7.
17. Упреждающее уведомление: Глава 13.
18. Пределы НЗП (незавершенного производства): Глава 7.
19. Визуализация: Глава 7.

Культурные возможности

20. Организационная культура Веструма: Глава 3.
21. Поддерживающее обучение: Глава 10.
22. Взаимодействие между командами: Главы 3 и 5.
23. Удовлетворенность работой: Глава 10.
24. Трансформационное лидерство: Глава 11.

Вступление

В конце 2013 года мы приступили к четырехлетнему научному путешествию, чтобы исследовать, какие возможности и методы важны для ускорения разработки и доставки программного обеспечения и, в свою очередь, какие из них представляют ценность для компаний. Их результативность проявляется в прибыльности компании, ее производительности и доле рынка. Мы видим аналогичный эффект и в некоммерческих результатах, в частности, когда речь идет об удовлетворении клиента.

Это исследование отвечает на потребность, которую в настоящее время не закрывает рынок. Используя строгие методы исследования, которые традиционно встречаются только в академических кругах, и делая их доступными для отрасли, мы преследуем цель продвинуть на новый уровень состояние разработки и доставки программного обеспечения. Помогая отрасли выявлять и понимать возможности, которые на самом деле приводят к повышению эффективности статистически значимым образом, мы выходим за пределы одного эпизода. Основываясь на

опыте одной или нескольких команд, мы можем помочь всей отрасли улучшиться.

Для проведения исследований, представленных в этой книге (в дополнение к тем, над которыми мы работаем до сих пор), мы используем перекрестные межгрупповые исследования. Те же методы используются в медицинских исследованиях (например, для изучения взаимосвязи между потреблением пива и ожирением, Бобак и соавторы, 2003), исследованиях рабочей среды (например, для изучения взаимосвязи между рабочей средой и сердечно-сосудистыми заболеваниями, Джонсон и Холл, 1988) и исследованиях памяти (например, для изучения различий в процессах развития и снижения памяти, Алловэй и Алловэй, 2013). Поскольку мы хотим по-настоящему изучить отрасль и понять, что приводит к значительному улучшению программного обеспечения и организационной эффективности, мы используем строгие методы построения научных исследований и публикуем большую часть нашей работы в академических журналах. Дополнительную информацию о методах нашего исследования вы найдете в Части II «Исследование».

Исследование

В рамках исследования мы собрали по нашим опросникам более 23 000 ответов со всего мира. Мы получили обратную связь от более чем 2000 уникальных организаций, от небольших стартапов с количеством сотрудников до пяти человек до крупных предприятий со штатом более 10 000 человек. Мы собрали данные от маленьких стартапов и передовых интернет-компаний, а также организаций, работающих в строго регулируемых отраслях, таких как финансы, здравоохранение и государственные структуры. Наши данные и анализ включают программное обеспечение, разработанное на совершенно новых платформах, так называемых greenfield («зеленое поле» — проекты, созданные с нуля. — Прим. ред.), а также обслуживание и разработку существующего кода ПО.

Выводы, приведенные в этой книге, применимы независимо от того, используете ли вы традиционную каскадную методологию (также известную как закрытая, структурированная или плановая) и только начинаете трансформацию технологии, или же уже много лет внедряете методы Agile и DevOps. Это верно, потому что доставка программного обеспечения — это упражнение в непрерывном улучшении и наше исследование показывает, что год за годом лучшие продолжают достигать новых высот, а те, кто не смог этого сделать, отстают все больше и больше.

Улучшение возможно для всех

Наши поиски понимания того, как оценить и улучшить доставку программного обеспечения, были полны открытий и сюрпризов. Мораль

этой истории, подтвержденная данными, заключается в следующем: улучшения в доставке программного обеспечения возможны для каждой команды и в каждой компании, пока руководство обеспечивает последовательную поддержку — включая время, действия и ресурсы — и, разумеется, пока члены команды ответственно выполняют свою работу.

Цель этой книги — поделиться тем, что мы узнали, чтобы помочь организациям преуспевать, вырастить более счастливые команды, которые быстрее доставляют лучшее программное обеспечение, и помочь процветанию организаций и отдельных людей. Остальная часть этого вступления кратко описывает, как началось и как проводилось это исследование. Более подробно о его научной основе вы прочтете в Части II этой книги.

Проделанная работа и данные

Нас часто спрашивают об истории происхождения этого исследования. Оно основано на непреодолимом интересе к тому, что делает лучшие технологические организации великими и как программное обеспечение делает организации лучше. Сначала авторы работали параллельно над пониманием исключительных технических характеристик, прежде чем объединить усилия в конце 2013 года.

- Николь Форсгрен имеет степень доктора философии в области управления информационными системами. До 2013 года она провела несколько лет, исследуя факторы, которые делают технологии эффективными в организациях — особенно среди профессионалов, которые создают программное обеспечение и инфраструктуру поддержки. Она является автором десятков научных статей на эту тему. До получения докторской степени она была инженером программного и аппаратного обеспечения и системным администратором.
- Джез Хамбл — соавтор книг «Непрерывная доставка» (Continuous Delivery), «Бережливое предприятие» (Lean Enterprise) и «Руководство по DevOps» (The DevOps Handbook). Его первой работой после колледжа был стартап в Лондоне в 2000 году, а затем в 2005–2015 годах он провел десять лет в компании ThoughtWorks, занимаясь доставкой программных продуктов и консультируя клиентов в качестве специалиста по инфраструктуре, разработчика и продукт-менеджера.
- Джин Ким изучает высокоэффективные технологические организации с 1999 года. Он был основателем и техническим директором компании Tripwire в течение 13 лет и является соавтором многих книг, включая «Проект Феникс» (The Phoenix Project) и «Руководство по Visible Ops» (The Visible Ops Handbook).

В конце 2013 года Николь, Джез и Джин начали работать вместе с командой компании Puppet Inc. в рамках подготовки «Отчета о состоянии DevOps 2014 года»3. Объединив практический опыт и академическую строгость, команда смогла создать нечто уникальное в отрасли: отчет, содержащий информацию о том, как сделать так, чтобы технологии создавали ценность для сотрудников, организаций и клиентов прогнозируемыми способами. В течение следующих четырех отчетов Николь, Джез и Джин продолжали сотрудничать с командой Puppet Inc., чтобы повторно применять методологию данного исследования и постоянно улучшать отраслевое понимание того, что способствует отличной доставке программного обеспечения, позволяет создать выдающиеся технические команды и как компании могут стать высокоэффективными организациями и выигрывать на рынке, используя технологии. Эта книга охватывает четыре года исследований, начиная с отчета (с 2014 по 2017 год).

Чтобы собрать данные, каждый год мы отправляли приглашения по нашим спискам рассылки и использовали социальные сети, включая Twitter, LinkedIn и Facebook. Наши приглашения были адресованы техническим специалистам, особенно тем, кто знаком с парадигмами разработки и доставки ПО, а также с DevOps. Мы призывали наших читателей приглашать друзей и коллег, которые тоже могли быть заняты в области разработки и доставки программного обеспечения, чтобы помочь нам расширить охват аудитории. Это называется «выборка по методу снежного кома», и мы поговорим о том, почему это был подходящий метод сбора данных для этого исследовательского проекта, в Главе 15 «Данные для проекта».

Данные для нашего проекта были получены из опросов. Мы использовали опросы, потому что это лучший способ собрать большой объем данных из тысяч организаций за короткий промежуток времени. Детальное обсуждение того, почему хорошие исследования могут быть проведены на основе опросов, а также шагов, которые мы предприняли для обеспечения достоверности и точности собранных данных, см. в Части II, которая охватывает научную основу и методологию исследования, лежащие в основе книги.

А вот краткий обзор исследования и того, как оно развивалось на протяжении нескольких лет.

2014 год: закладывая основы. Эффективность доставки ПО и организационная эффективность

Наши исследовательские цели в течение первого года состояли в том, чтобы заложить основу для понимания разработки и доставки программного обеспечения в организациях. Вот некоторые ключевые вопросы.

- Что значит доставка программного обеспечения и можно ли ее измерить?
- Влияет ли доставка ПО на организации?
- Имеет ли значение культура и как мы ее измеряем?
- Какие технические практики представляются важными?

Мы были приятно удивлены многими результатами в первый год. Мы обнаружили, что разработка и доставка программного обеспечения могут быть измерены статистически значимым образом и что успешные компании делают это постоянно с помощью методов, которые значительно лучше, чем у многих других компаний. Мы также обнаружили, что скорость и стабильность движутся вместе и способность организации создавать программное обеспечение положительно влияет на прибыльность, производительность и долю рынка. Мы увидели, что культура и технические практики имеют значение, и нашли способ их измерить. Об этом мы расскажем в первой части этой книги.

Группа также пересмотрела способ измерения большинства данных, перейдя от простых вопросов «да/нет» к вопросам по шкале Ликерта (в которых респонденты выбирают из спектра вариантов от «категорически не согласен» до «полностью согласен»). Это простое изменение в вопросах позволило команде собрать больше нюансов — оттенки серого вместо черно-белого. Это позволило провести более детальный анализ. Почему авторы решили использовать опросы для этого исследовательского проекта и почему вы можете доверять их данным, мы обсудим в Главе 14 «Зачем использовать опрос».

2015 год: расширение работы и углубление анализа

Подобно технологическим преобразованиям и росту бизнеса, проведение исследований — это итерации, постепенные улучшения и переоценка важных результатов. Вооружившись нашими выводами за первый год, мы поставили в качестве целей на второй год пересмотр и подтверждение некоторых ключевых результатов исследования (например, что доставка программного обеспечения может быть определена и измерена статистически значимым образом и что она влияет на эффективность организации), а также расширение модели исследования.

Ниже представлены некоторые из вопросов исследования.

- Можем ли мы повторно подтвердить, что доставка программного обеспечения влияет на эффективность работы организации?
- Влияют ли технические методы и автоматизация на доставку программного обеспечения?
- Влияют ли методы бережливого управления на доставку программного обеспечения?

- Влияют ли технические методы и методы бережливого управления на те аспекты работы, которые оказывают воздействие на человеческие ресурсы, например, беспокойство, связанное с развертыванием кода и профессиональным выгоранием сотрудников?

Еще раз мы получили великолепные подтверждения по некоторым вопросам и ряд сюрпризов. Наши гипотезы подтвердились, расширив рамки работы, проделанной нами в прошлом году. Эти результаты вы можете найти в Части I.

2016 год: расширение нашего взгляда на технические методы и изучение нечеткого внешнего интерфейса

На третьем году мы вновь опирались на основное ядро нашей модели и расширили ее, чтобы изучить значение дополнительных технических методов (таких как безопасность, магистральная разработка и управление тестовыми данными). Вдохновленные беседами с коллегами, работающими в области управления продуктами, мы еще расширили наше исследование, чтобы увидеть, можем ли мы измерить влияние текущего перехода от традиционных методов управления проектами к использованию принципов бережливого производства в управлении продуктами. Мы углубили наши изыскания, чтобы включить качественные измерения, такие как ошибки, доработки и восстановление безопасности. Наконец, мы включили дополнительные вопросы, чтобы попробовать понять, как технические методы влияют на человеческий капитал: индекс чистой лояльности сотрудников (eNPS) и приверженность работе — факторы, которые, как предполагается, должны уменьшать выгорание.

Ниже представлены вопросы нашего исследования.

- Помогает ли интеграция безопасности в разработку и доставку программного обеспечения этому процессу или замедляет его?
- Способствует ли магистральная разработка улучшению доставки программного обеспечения?
- Является ли бережливый подход к управлению продуктами важным аспектом разработки и доставки программного обеспечения?
- Способствуют ли хорошие технические практики повышению лояльности сотрудников?

2017 год: добавляем архитектуру, изучение роли лидеров и измерение успеха в некоммерческих организациях

На четвертом году исследования мы перешли к вопросам о том, как проектируются системы и как архитектура влияет на способность команд и организаций доставлять программное обеспечение и формировать ценность. Мы также расширили наше исследование, чтобы включить в него измерение ценностей, которые выходят за рамки рентабельности, производительности и доли рынка, что позволяет анализу обращаться к некоммерческой аудитории. В исследовании этого года также изучалась роль лидеров для оценки влияния трансформационного лидерства в организациях.

Ключевые вопросы четвертого года нашего исследования.

- Какие архитектурные практики способствуют повышению эффективности доставки программного обеспечения?
- Как трансформационное лидерство влияет на доставку программного обеспечения?
- Влияет ли доставка программного обеспечения на некоммерческие результаты?

Вывод

Мы надеемся, что, читая эту книгу, вы, как технический специалист и технологический лидер, откроете для себя важные элементы для улучшения своей организации, начиная прежде всего с доставки программного обеспечения. Именно благодаря улучшению нашей способности доставлять программное обеспечение организации могут быстрее предоставлять нужный функционал, при необходимости адаптироваться, реагировать на изменения в требованиях безопасности, а также использовать быструю обратную связь для привлечения новых клиентов и удовлетворения существующих.

В следующих главах мы определим ключевые возможности, определяющие эффективность доставки программного обеспечения (и определим, что такое эта эффективность), и кратко коснемся ключевых моментов каждой из них. Часть I книги представляет результаты наших исследований, Часть II рассматривает научную основу и исследования, стоящие за нашими результатами, и, наконец, в Части III представлен конкретный пример того, что происходит, когда организации принимают и внедряют эти возможности для повышения эффективности.

ЧАСТЬ I

ЧТО МЫ ВЫЯСНИЛИ

Вооружившись надежными методами сбора данных и статистического анализа (подробно рассмотренными в Части II), мы смогли обнаружить значительные и иногда неожиданные результаты за последние

несколько лет, в течение которых мы работали над «Отчетом о состоянии DevOps». Мы смогли измерить и количественно оценить эффективность доставки программного обеспечения, ее влияние на эффективность работы организации и различные возможности, способствующие достижению этих результатов.

Эти возможности делятся на различные категории — технические, технологические и культурные. Мы измерили воздействие технических практик на культуру, а также влияние культуры на доставку и организационную эффективность. Говоря о таких несопоставимых возможностях, как архитектура и управление продуктами, мы рассмотрели их общее влияние на эти и другие важные показатели устойчивого развития, такие, в частности, как профессиональное выгорание и личные мучения при развертывании ПО.

В этой части книги мы представляем наши результаты.

Глава 1

Ускоряйтесь

Вести бизнес как обычно уже недостаточно, чтобы оставаться конкурентоспособным. Это касается организаций во всех отраслях: от финансов и банковского дела до розничной торговли, телекоммуникаций и даже госструктур. Компании отказываются от выпуска новых продуктов и услуг, связанных с реализацией крупных долгосрочных проектов. Вместо этого они используют небольшие проектные команды, которые работают в коротких временных циклах и измеряют обратную связь от пользователей для создания продуктов и услуг, которые радуют клиентов и быстро приносят пользу их организациям. Такие эффективные компании постоянно работают над тем, чтобы стать лучше в своем деле, не позволяя никаким препятствиям стоять у них на пути, даже перед лицом высокого уровня риска и неопределенности в том, как им достичь своих целей.

Чтобы оставаться конкурентоспособными и преуспевать на рынке, организации должны ускорить:

- доставку товаров и услуг, чтобы порадовать своих клиентов;
- взаимодействие с рынком для выявления и понимания потребительского спроса;
- предвидение изменений в области регулирования и соблюдения требований, которые влияют на их системы;
- реакцию на потенциальные риски, такие как угрозы безопасности или изменения в экономике.

В основе этого ускорения лежит программное обеспечение. Это справедливо для организаций любой отрасли. Банки больше не создают

ценность, держа золотые слитки в хранилищах, а торгуют быстрее и надежнее, открывая новые каналы и продукты для привлечения клиентов. Компании розничной торговли завоевывают и удерживают клиентов, предлагая им превосходный выбор и обслуживание, причем обслуживание заключается в молниеносной оплате на кассе, рекомендуемых товарах или удобном онлайн/офлайн-шопинге — и все это обеспечивается технологиями. Правительственные организации, будучи традиционно прижимистыми в распределении средств налогоплательщиков, ссылаются на способность использовать технологии в качестве ключа к более успешному и эффективному служению обществу.

Программное обеспечение и технологии являются ключевыми отличительными факторами для организаций при формировании ценности как для клиентов, так и для акционеров. Мы пришли к этому выводу в наших собственных исследованиях, описанных в этой книге, но и другие исследователи тоже пришли к аналогичным результатам. Например, недавнее исследование Джеймса Бессена из Бостонского университета показало, что стратегическое использование технологий объясняет рост доходов и производительности больше, чем слияния и поглощения (M&A) и предпринимательство (2017). Эндрю Макафи и Эрик Бринольфссон также нашли связь между технологиями и рентабельностью (2008).

Программное обеспечение преобразует и ускоряет организации всех видов. Практики и возможности, о которых мы говорим в этой книге, возникли из того, что теперь известно как движение DevOps, и они трансформируют отрасли повсюду. DevOps возникло из небольшого числа организаций, столкнувшихся с серьезной проблемой: как построить безопасные, устойчивые, быстро развивающиеся распределенные и масштабируемые системы. Чтобы оставаться конкурентоспособными, организации должны научиться решать эти проблемы. Мы видим, что крупные предприятия с долгой историей и технологиями десятилетней давности также получают значительные преимущества в виде ускоренной доставки и более низких затрат благодаря использованию возможностей, которые мы описываем в этой книге.

Несмотря на то, что многие организации достигли больших успехов в своих технологических преобразованиях (заметными примерами являются такие интернет-гиганты, как Netflix, Amazon, Google и Facebook, а также более традиционные крупные компании, включая Capital One, Target, Службу технологических преобразований при Правительстве США и Цифровую службу США), впереди еще по-прежнему много работы — как в самой индустрии в широком смысле, так и в отдельных организациях. Согласно недавнему отчету Forrester (Страуд и соавторы, 2017), выяснилось, что 31% игроков в отрасли не используют практики и принципы, которые широко признаны необходимыми для ускорения

технологических преобразований. К ним относятся технологии непрерывной интеграции и непрерывного развертывания ПО, практики бережливого производства и культура сотрудничества при разработке (то есть возможности, отстаиваемые движением DevOps). Тем не менее мы также знаем, что технологические и программные преобразования обязательны к внедрению в организациях сегодня. Недавнее исследование Gartner показало, что 47% генеральных директоров испытывают давление со стороны членов совета директоров, которые настаивают на внедрении цифровых преобразований (Panetta, 2017).

Внутри самих организаций технологические преобразования находятся на разных стадиях, и отчеты показывают, что еще предстоит проделать гораздо больший объем работы, чем многие из нас сегодня представляют. В другом докладе Forrester говорится, что DevOps ускоряет развитие технологий, но организации часто переоценивают свой прогресс (Клавенс и соавторы, 2017). Кроме того, в докладе отмечается, что особенно склонны переоценивать прогресс руководители — в отличие от тех, кто фактически выполняет работу.

Эти выводы о несоответствии оценок зрелости DevOps со стороны руководителей и исполнителей обращают внимание на два аспекта, которые часто упускаются лидерами. Во-первых, если мы предположим, что оценки зрелости или возможностей DevOps от практиков более точны, потому что они ближе к работе, то потенциал для создания ценности и роста внутри организаций намного выше, чем руководители осознают в настоящее время. Во-вторых, выявленное различие в восприятии ясно показывает необходимость более точной оценки возможностей DevOps и передачи результатов этой оценки лидерам, которые могут использовать их для принятия решений и формирования стратегии в отношении технологического развития своей организации.

Сосредоточьтесь на возможностях, а не на зрелости

Технологические лидеры должны доставлять программное обеспечение быстро и надежно, чтобы преуспевать на рынке. Для многих компаний это требует значительных изменений в том, как это осуществляется. Ключами к успешному изменению являются оценка и понимание возможностей, а не зрелости продукта.

Хотя модели зрелости очень популярны в отрасли, мы не можем с уверенностью говорить о том, что они являются подходящим инструментом для использования на практике или в качестве модели мышления. Напротив, переход к модели измерения возможностей очень значим для организаций, желающих ускорить доставку программного обеспечения. Это обусловлено четырьмя факторами.

Во-первых, модели зрелости направлены на то, чтобы помочь организации прийти в зрелое состояние, а затем заявить о завершении своего пути, в то время как технологические преобразования должны следовать парадигме непрерывного улучшения. Кроме того, модели возможностей направлены на то, чтобы помочь организации постоянно совершенствоваться и прогрессировать, с учетом того, что технологический и деловой ландшафты постоянно меняются. Самые инновационные компании и самые эффективные организации всегда стремятся быть лучше и никогда не считают себя зрелыми или закончившими свой путь совершенствования или трансформации, и мы рассмотрим это в нашей работе.

Во-вторых, модели зрелости довольно часто являются жестко регламентированными стоп-шагами или представляют собой линейную формулу, предписывающую аналогичный набор технологий, инструментов или возможностей для каждой группы команд или организаций. Модели зрелости предполагают, что уровень 1 и уровень 2 выглядят одинаково во всех командах и организациях, но те из нас, кто работает в технологиях, знают, что это не так. Напротив, модели возможностей являются многомерными и динамичными, что позволяет различным подразделениям организации применять индивидуальный подход к улучшениям и фокусироваться на возможностях, которые принесут им наибольшую пользу, исходя из их текущего контекста и краткосрочных и долгосрочных целей. Команды имеют свой собственный контекст, свои собственные системы, свои собственные цели и свои собственные ограничения, и от этого зависит то, на чем мы должны сосредоточиться на следующем этапе, чтобы ускорить нашу трансформацию.

В-третьих, модели возможностей фокусируются на ключевых результатах и на том, как возможности способствуют улучшению этих результатов, то есть они основаны на результатах. Это обеспечивает техническое лидерство с четким направлением и стратегией на цели высокого уровня (с акцентом на возможности улучшения ключевых результатов). Это также позволяет руководителям групп и отдельным участникам устанавливать цели, связанные с возможностями, на которых сосредоточена их команда в текущий период времени. Большинство моделей зрелости просто измеряют техническую квалификацию или инструментальную базу в организации, не привязывая их к результатам. Они в конечном итоге являются метриками тщеславия: хотя их можно относительно легко измерить, они ничего не говорят нам о том, какое влияние они оказывают на бизнес.

В-четвертых, модели зрелости определяют статический уровень технологических, производственных и организационных возможностей, который может быть достигнут. Они не учитывают постоянно меняющийся характер технологий и бизнес-ландшафта. Наши собственные исследования и данные подтвердили, что отрасль

меняется: то, что достаточно хорошо и даже высокоэффективно сегодня, уже недостаточно хорошо в следующем году. Напротив, модели возможностей позволяют динамично изменять окружающую среду и позволяют командам и организациям сосредоточиться на развитии навыков и возможностей, необходимых для сохранения конкурентоспособности.

Сосредоточившись на парадигме возможностей, организации могут постоянно стимулировать улучшение. А сосредоточившись на правильных возможностях, организации способны добиться улучшения своих результатов, что позволит им разрабатывать и доставлять программное обеспечение с более высокой скоростью и стабильностью. Фактически мы видим, что самые эффективные компании делают именно это, постоянно достигая успехов из года в год и никогда не соглашаясь на вчерашние достижения.

Преобразования, основанные на фактических данных, сфокусированы на ключевых возможностях

Как в рамках модели возможностей, так и в рамках модели зрелости существуют разногласия относительно того, на каких возможностях следует сосредоточиться. Поставщики продуктов часто предпочитают возможности, которые согласуются с их предложениями. Консультанты предпочитают возможности, которые отвечают их опыту, предложению и их привычным инструментам оценки. Мы видели, как организации пытаются разработать свои собственные модели оценки и выбрать решения, которые соответствуют навыкам внутренних «чемпионов» или поддаются параличу анализа из-за огромного количества областей, которые нуждаются в улучшении.

Необходимо более управляемое, основанное на фактических данных решение. И подход, обсуждаемый в этой книге, описывает именно такое решение.

Наше исследование дало понимание того, что обеспечивает эффективность доставки программного обеспечения и организационную эффективность, которые отражаются на прибыльности, производительности и доле рынка компании. На самом деле наше исследование показывает, что ни один из следующих часто цитируемых факторов не предполагал эффективности:

- время и технологии, используемые для приложения (например, программа «системы записи» против «системы взаимодействия», относящейся к «зеленому полю» разработки);
- техподдержка или команда разработки осуществляет развертывание ПО;

- будет ли организован совет по утверждению изменений (CAB — change approval board. — Прим. ред.).

То, что действительно имеет значение для успеха доставки программного обеспечения и организационной эффективности, — это те практики, которые используют лидеры рынка и самые инновационные компании, чтобы продвигаться вперед. Наше исследование выявило 24 ключевые возможности, которые способствуют повышению эффективности доставки программного обеспечения и, в свою очередь, организационной эффективности. Эти возможности легко определить, измерить и улучшить4. Эта книга поможет вам начать работу по определению и оценке этих возможностей. Мы также укажем вам на некоторые фантастические ресурсы для их улучшения, чтобы вы могли ускорить свое собственное путешествие по трансформации технологий.

Значение принятия концепции DevOps

Вы можете спросить себя: откуда мы знаем, что эти возможности являются движущими силами технологий и организационной эффективности, и почему мы можем говорить об этом с такой уверенностью?

Результаты нашей исследовательской программы ясно показывают, что ценность внедрения DevOps даже выше, чем мы изначально думали, и разрыв между высокоэффективными компаниями и организациями с низкими показателями продолжает расти.

В следующей главе мы подробно обсудим, как мы измеряем эффективность доставки программного обеспечения и как работает наша группа исследователей.

Подводя итог, в 2017 году мы обнаружили, что по сравнению с низкоэффективными компаниями лидеры отрасли показывают следующие результаты:

- в 46 раз более частое развертывание кода;
- в 440 раз меньшее время выполнения от коммита до развертывания;
- в 170 раз меньшее среднее время восстановления после простоя;
- в 5 раз более низкий показатель отказов изменений (1/5, скорее для того, чтобы изменение не состоялось).

По сравнению с результатами 2016 года разрыв между компаниями с высокими и низкими показателями сократился по скорости (частоте развертывания и времени внедрения изменений) и увеличился по стабильности (среднее время восстановления и уровень отказов при внедрении изменений). Мы предполагаем, что это происходит из-за низкой производительности команд, работающих над увеличением

скорости, но недостаточно инвестирующих в выстраивание качественных процессов. Результатом являются более крупные сбои развертывания, которые требуют больше времени для восстановления работы. Компании с высокими показателями эффективности понимают, что им не нужно жертвовать скоростью ради стабильности или наоборот, потому что, создавая качество, они получают и то и другое.

Вам может быть интересно, как высокоэффективные команды достигают такой удивительной эффективности доставки программного обеспечения. Они делают это, нажимая на правильные рычаги, то есть улучшая правильные возможности. В ходе нашей четырехлетней исследовательской программы мы смогли определить те из них, которые повышают эффективность доставки программного обеспечения и влияют на эффективность организации. И мы обнаружили, что они работают для всех типов организаций. Наше исследование изучало организации всех размеров, во всех отраслях, по всему миру, использующие классические и инновационные технологии. Поэтому заключения, которые вы найдете на страницах этой книги, также применимы к командам в вашей организации.

Глава 2

Измерение эффективности

Существует много концепций и методологий, направленных на улучшение способа создания программных продуктов и услуг. Мы хотели выяснить, что работает, а что нет, в научном смысле, начиная с определения того, что означает «хорошо» в этом контексте. В этой главе представлены структура и методы, которые мы использовали для достижения этой цели, и, в частности, ключевые результаты измерений, применяемые на протяжении всей этой книги.

Мы надеемся, что к концу этой главы вы узнаете достаточно о нашем подходе, чтобы быть уверенными в результатах, которые мы представим в остальной части книги.

Измерение эффективности в области программного обеспечения является трудным — отчасти потому, что, в отличие от обычного производства, инвентарь остается невидимым. Кроме того, мы разбиваем работу относительно произвольно, и деятельность по проектированию и доставке, особенно в парадигме Agile для разработки программного обеспечения, происходит одновременно. Действительно, ожидается, что мы будем изменять и развивать наш подход, основанный на том, что мы узнаем в процессе его реализации. Таким образом, наш первый шаг должен заключаться в определении достоверного, надежного способа измерения эффективности доставки программного обеспечения.

Недостатки предыдущих попыток измерения эффективности

Было много попыток измерить эффективность команд по разработке ПО. Большинство этих измерений было сосредоточено на производительности. В общем они страдают от двух недостатков. Во-первых, они сосредоточены на производстве, а не на конечных результатах. Во-вторых, они фокусируются на индивидуальных или локальных измерениях, а не на коллективных или глобальных. Возьмем три примера: строки кода, скорость и загрузка.

Измерение эффективности с точки зрения строк кода имеет долгую историю в программном обеспечении. Некоторые компании даже требовали от разработчиков записывать строки кода, зафиксированные в в течение недели5. Однако в реальности мы предпочли бы 10-строчное решение задачи 1000-строчному. Вознаграждение разработчиков за написание строк кода приводит к раздутому программному обеспечению, что влечет увеличение затрат на обслуживание и более высокую стоимость изменений. В идеале мы должны вознаграждать разработчиков за решение бизнес–задач с минимальным количеством кода. И даже лучше, если мы можем решить проблему без написания кода вообще или путем удаления кода (возможно, путем изменения бизнес-процесса). Однако минимизация строк кода также не является идеальным параметром. В крайнем случае она тоже имеет свои недостатки: выполнение задачи в одной строке кода, которую никто другой не может понять, менее желательно, чем написание нескольких строк кода, которые легко понять и поддерживать.

С появлением методов Agile для разработки программного обеспечения появился новый способ измерения эффективности: скорость. Во многих школах сторонников Agile задачи разбиваются на истории. Истории, в свою очередь, оцениваются разработчиками, и им присваивается определенное количество очков, которые означают относительное количество усилий, требующихся, чтобы выполнить эти задачи. В конце итерации общее количество очков при сдаче работы заказчику записывается — это и есть скорость команды. Показатель скорости предназначен для использования в качестве инструмента планирования ресурсов; например, он может быть использован для экстраполяции, сколько времени потребуется команде, чтобы выполнить все, что запланировано и подсчитано. Тем не менее некоторые менеджеры также использовали его как способ измерения производительности команды или даже для сравнения команд.

Использование скорости в качестве показателя эффективности имеет несколько недостатков. Во-первых, скорость является относительной величиной и зависимым от команды параметром, а не абсолютным значением. Команды обычно работают в значительно различающихся

условиях, которые делают их скорости несоизмеримыми. Во-вторых, когда скорость используется в качестве меры производительности, команды неизбежно стремятся к тому, чтобы увеличить показатели своей скорости. Они завышают свои оценки и сосредотачиваются на завершении как можно большего количества историй за счет сотрудничества с другими командами (что может уменьшить их скорость и увеличить скорость другой команды, заставляя их выглядеть плохо). Это не только нарушает использование скорости по назначению, но и препятствует сотрудничеству между командами.

Наконец, многие организации измеряют загрузку как показатель эффективности. Проблема с этим методом заключается в том, что высокая загрузка хороша только до определенного момента. Как только параметр использования становится выше определенного уровня, нет никакой запасной мощности (или резерва), чтобы поглотить незапланированную работу, изменения в плане или работу по улучшению. Это приводит к более длительным срокам выполнения работ. Теория очередей в математике говорит нам, что по мере того, как загрузка приближается к 100%, время производственного цикла приближается к бесконечности. Другими словами, как только вы достигаете очень высоких уровней загрузки, командам требуется экспоненциально больше времени на выполнение любой задачи. Поскольку время выполнения заказа, то есть то, как быстро может быть выполнена работа, — это показатель продуктивности, не страдающий от недостатков, присущих другим показателям, которые мы рассмотрели, важно, чтобы мы управляли загрузкой, сбалансировав ее со временем выполнения экономически оптимальным способом.

Измерение эффективности доставки программного обеспечения

Успешное измерение эффективности должно обладать двумя ключевыми характеристиками. Во-первых, оно должно сосредоточиться на глобальном результате, чтобы команды не сталкивались друг с другом. Классический пример — награждение разработчиков за скорость, а техподдержки — за стабильность; это главный кирпич в «стене путаницы», через которую разработка перебрасывает код низкого качества техподдержке, а та инициирует болезненные процессы управления изменениями как способ подавления изменений. Во-вторых, наша оценка должна фокусироваться на конечных, а не промежуточных результатах. Она не должна вознаграждать людей за то, что они проделывают большое количество работы, которая на самом деле не помогает достичь организационных целей.

При поиске показателей эффективности доставки ПО, удовлетворяющих этим критериям, мы остановились на четырех: время выполнения доставки, частота развертывания, время восстановления службы и

частота сбоев при внесении изменений. В этом разделе мы обсудим, почему мы выбрали именно эти критерии.

Таблица 2.1

Разработка vs Доставка

Проектирование и разработка продукта	Доставка продукта (сборка, тестирование, развертывание)
Создание новых продуктов и услуг, которые решают проблемы клиентов с помощью гипотетической доставки, пользовательского опыта и дизайн-мышления	Быстрый переход от разработки к производству и надежным релизам продукта путем стандартизации работы и уменьшения вариативности и размеров пакета
Проектирование и реализация функций может потребовать работы, которая никогда ранее не выполнялась	Интеграция, тестирование и развертывание должны выполняться непрерывно и как можно быстрее
Оценки затрат весьма неопределенны	Время цикла должно быть хорошо известно и предсказуемо
Результаты очень разнообразны	Результаты должны иметь низкую вариативность

Время выполнения заказа как метрика является ключевым элементом теории бережливого производства. Это время, необходимое для перехода от клиента, делающего запрос, к удовлетворенному запросу. Однако в контексте разработки продукта, где мы стремимся удовлетворить множество клиентов, есть две составляющие времени выполнения заказа: время, необходимое для разработки и проверки продукта или функции, и время для доставки функции клиентам. В проектной части времени выполнения бывает неясно, когда запускать часы, и часто присутствует высокая вариативность. По этой причине Райнертсен называет эту часть времени выполнения «нечетким входным интерфейсом» (Райнертсен, 2009). Тем не менее часть времени, относящуюся к доставке, то есть время, необходимое для внедрения и тестирования работ, легче измерить, и оно имеет меньшую вариативность. Таблица 2.1 (Ким и соавторы, 2016) показывает различие между этими двумя областями.

Более короткие сроки доставки продукта лучше, так как они позволяют быстрее получать обратную связь о том, что мы строим, и позволяют нам быстрее корректировать курс. Короткие сроки выполнения также важны, когда есть дефект или имеет место простой и мы должны внести исправления быстро и с высокой степенью уверенности. Мы измерили время выполнения доставки продукта как время, необходимое для перехода от принятого кода к коду, успешно работающему в производстве, и попросили респондентов опроса выбрать один из следующих вариантов:

- менее одного часа;
- менее одного дня;
- от одного дня до одной недели;

- от одной недели до одного месяца;
- от одного месяца до шести месяцев;
- более шести месяцев.

Второй показатель, который следует учитывать, — это размер пакета. Его сокращение является еще одним важным элементом парадигмы бережливого производства — кстати, это был один из ключей к успеху производственной системы Toyota. Уменьшение размеров партии сокращает время цикла и изменчивость потока, ускоряет обратную связь, снижает риски и накладные расходы, повышает эффективность, мотивацию и срочность, а также снижает затраты и риск нарушения сроков (Райнертсен, 2009, глава 5). Однако в программном обеспечении размер пакета трудно измерить и сообщить в разных контекстах, поскольку нет никакого видимого инвентаря. Поэтому мы остановились на частоте развертывания в качестве показателя для определения размера пакета, поскольку она проста в измерении и обычно имеет низкую изменчивость6.

Под развертыванием мы подразумеваем развертывание программного обеспечения в производство или в магазин приложений. Выпуск версии ПО (изменения, которые будут развернуты), как правило, состоит из нескольких коммитов в контроле версий, если организация не достигла единичного потока, где каждый коммит может быть запущен в производство (практика, известная как непрерывное развертывание). Мы спросили наших респондентов, как часто их организации развертывают код для основного сервиса или приложения, над которым они работают, предлагая следующие варианты:

- по требованию (несколько развертываний в день);
- от одного раза в час до одного раза в день;
- от одного раза в день до одного раза в неделю;
- от одного раза в неделю до одного раза в месяц;
- от одного раза в месяц до одного раза в шесть месяцев;
- реже одного раза в полгода.

И время выполнения заказа на доставку, и частота развертывания являются показателями скорости выполнения доставки программного обеспечения. Тем не менее мы хотели исследовать, делают ли это команды, которые повышают свою эффективность, за счет стабильности систем, над которыми они работают. Традиционно надежность измеряется как время между отказами. Однако в современных программных продуктах и сервисах, которые стремительно меняют сложные системы, отказ неизбежен, поэтому ключевым становится вопрос: как быстро можно восстановить службу? Мы спросили респондентов, сколько времени обычно требуется для восстановления основного приложения или сервиса, над которым они работают, когда происходит инцидент (например, незапланированное отключение,

ухудшение обслуживания), предложив те же варианты, что и для времени выполнения (выше).

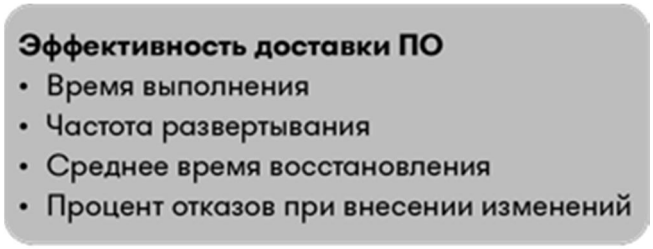


Рис. 2.1. **Эффективность доставки ПО**

Наконец, ключевым показателем при внесении изменений в систему является процент отказов в производстве (включая, например, выпуски программного обеспечения и изменения конфигурации инфраструктуры). В контексте бережливого производства это то же самое, что и процент полноты и точности для процесса доставки продукта, то есть ключевой показатель качества. Мы спросили респондентов, какой процент изменений основного приложения или сервиса, над которыми они работают, приводит к ухудшению обслуживания или впоследствии требует исправления (например, приводит к ухудшению обслуживания или отключению, требует пакета исправлений, отката системы, доработки или патча). Четыре выбранных параметра показаны на рис. 2.1.

Для анализа эффективности доставки по всей обследованной группе мы использовали метод, называемый кластерным анализом. Кластерный анализ — это основополагающий метод статистического анализа данных, который пытается сгруппировать ответы таким образом, чтобы ответы в одной и той же группе были более похожи друг на друга, чем на ответы в других группах. Каждое измерение помещается в отдельное пространство, и алгоритм кластеризации пытается минимизировать расстояние между всеми членами кластера и максимизировать различия между кластерами. Этот метод не имеет никакого представления о семантике ответов — другими словами, он не знает, что считается хорошим или плохим ответом[7].

Этот основанный на данных подход, который классифицирует данные без какого-либо смещения в сторону хорошего или плохого, позволяет нам просматривать тенденции в отрасли, не искажая результаты априори. Использование кластерного анализа также позволило нам выявить категории эффективности доставки программного обеспечения, наблюдаемые в отрасли: есть ли лидеры и отстающие и какими характеристиками они обладают?

Мы применяли кластерный анализ в течение всех четырех лет нашего исследовательского проекта и обнаружили, что каждый год в отрасли возникают заметно отличающиеся категории эффективности доставки программного обеспечения. Мы также обнаружили, что все четыре показателя эффективности доставки ПО являются хорошими

классификаторами и что группы, которые мы определили в анализе: лидеры, средние и отстающие, — существенно различаются по всем четырем показателям.

В таблицах 2.2 и 2.3 приведены подробные сведения об эффективности доставки программного обеспечения за последние два года нашего исследования (2016 и 2017).

Таблица 2.2

Эффективность доставки ПО в 2016 году

2016	Лидеры	Средние	Отстающие
Частота развертывания	По запросу (множественные развертывания в течение дня)	От одного раза в неделю до одного раза в месяц	От одного раза в неделю до одного раза в полгода*
Время внесения изменений	Менее одного часа	От одной недели до одного месяца	От одного месяца до полугода
Среднее время восстановления	Менее одного часа	Менее одного дня	Менее одного дня*
Процент отказов при изменениях	0–15%	31–45%	16–30%

Таблица 2.3

Эффективность доставки ПО в 2017 году

2017	Лидеры	Средние	Отстающие
Частота развертывания	По запросу (множественные развертывания в течение дня)	От одного раза в неделю до одного раза в месяц	От одного раза в неделю до одного раза в полгода*
Время внесения изменений	Менее одного часа	От одной недели до одного месяца	От одного месяца до полугода
Среднее время восстановления	Менее одного часа	Менее одного дня	От одного дня до одной недели
Процент отказов при изменениях	0–15%	0–15%	31–45%

* Компании с низкой эффективностью имели худшие результаты (на статистически значимом уровне), но их средние показатели были близки к показателям участников со средними результатами.

Удивительно, но эти результаты показывают, что нет никакого компромисса между повышением эффективности и достижением более высоких уровней стабильности и качества. Скорее, мы можем говорить о том, что лидеры лучше справляются со всеми этими показателями. Это именно то, что прогнозируют движения Agile и Lean, но главная догма в нашей отрасли все еще основывается на ложном предположении, что двигаться быстрее означает принести в жертву скорости другие цели эффективности.

Кроме того, за последние несколько лет мы обнаружили, что высокоэффективный кластер отрывается от всех остальных. Мантра

DevOps о непрерывном совершенствовании является одновременно захватывающей и реальной, подталкивая компании к тому, чтобы быть лучшими, и оставляя позади тех, кто не улучшается. Очевидно, что то, что было современным три года назад, просто недостаточно хорошо для сегодняшней бизнес-среды.

По сравнению с 2016 годом участники опроса с высокими показателями сохранили или улучшили их в 2017 году, постоянно наращивая темп и стабильность. С другой стороны, участники с низкими показателями поддерживали тот же уровень пропускной способности в 2014–2016 годах и начали расти только в 2017 году, вероятно, осознав разрыв со всей остальной отраслью. В 2017 году мы зафиксировали у участников с низкими показателями потерю в стабильности. Мы подозреваем, что это связано с их попытками увеличить темп («работай усерднее!»), что не позволяет устранить основные препятствия на пути повышения общей эффективности (например, решить задачи реархитектуры, совершенствования процессов и автоматизации). Данные тенденции показаны на рис. 2.2 и 2.3.

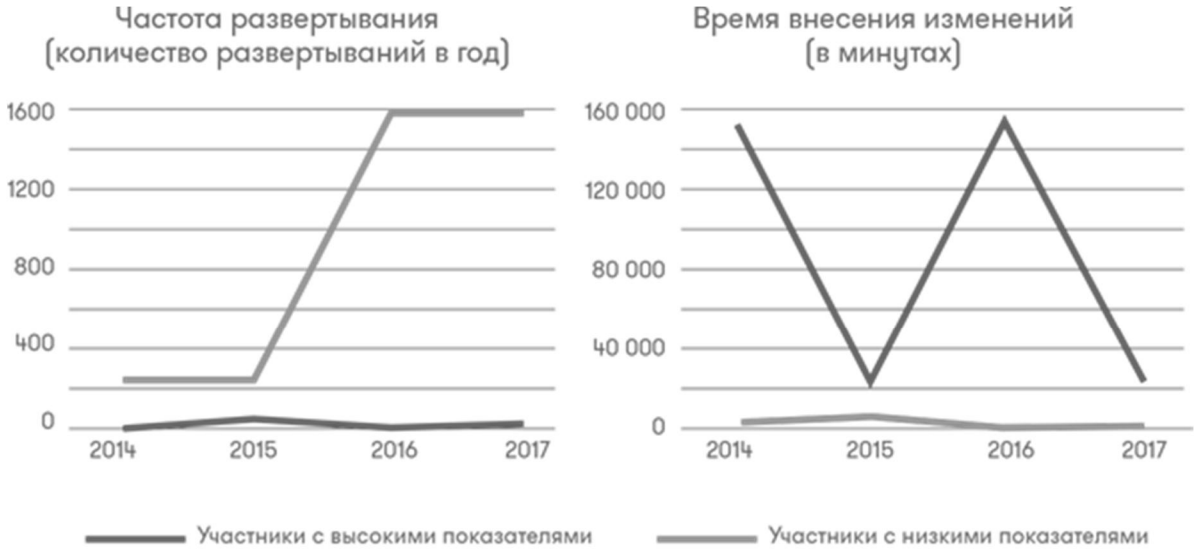


Рис. 2.2. **Тренды от года к году: скорость**

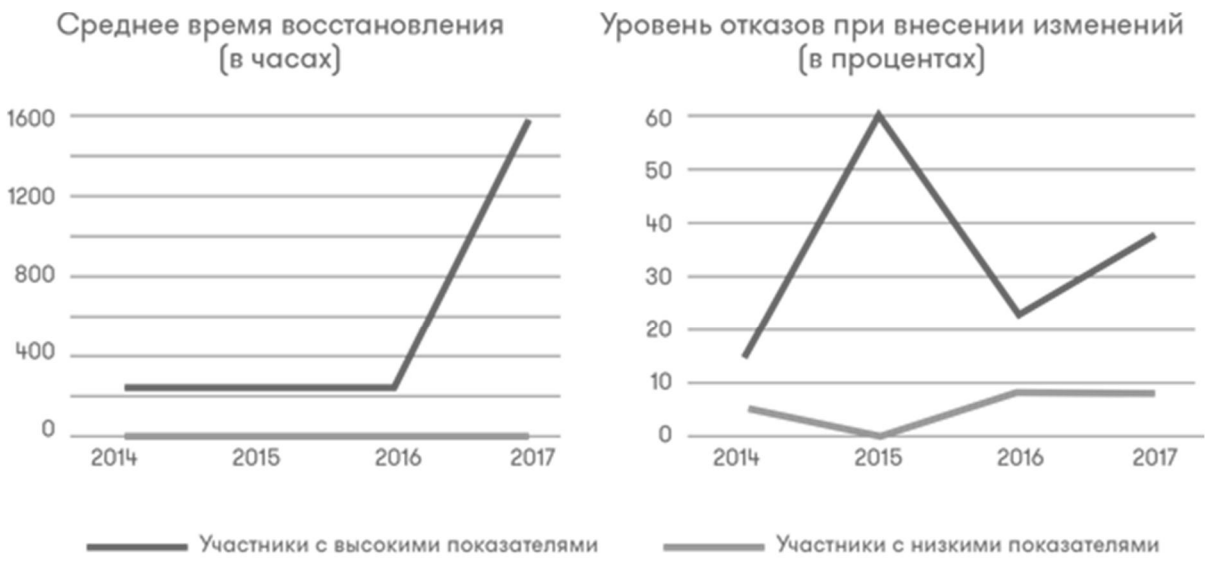


Рис. 2.3. **Тренды от года к году: стабильность**

Сюрприз!

Наблюдательные читатели заметят, что у средних участников показатель уровня отказов при внесении изменений в 2016 году хуже, чем у отстающих. 2016 год — это первый год нашего исследования, когда мы видим несколько несистемную эффективность по всем нашим показателям в каждой из групп, включая средних и отстающих. Наше исследование не дает окончательного объяснения этому явлению, но у нас есть несколько идей о том, почему это может быть.

Одно из возможных объяснений заключается в том, что средние участники работают над своей технологической трансформацией и решают задачи, связанные с крупномасштабными проектами по перестройке архитектуры, такие как, например, переход с устаревших баз кода. Это также соответствовало бы другой части данных из исследования 2016 года, где мы обнаружили, что средние тратят больше времени на незапланированную переработку, чем отстающие, потому что они тратят большую часть времени на новую работу.

Мы полагаем, что эта новая работа может происходить за счет игнорирования критических доработок, что тем самым увеличивает технический долг, а это, в свою очередь, приводит к более хрупким системам и, следовательно, более высокому показателю отказов при изменениях.

Мы нашли достоверный, надежный способ измерения эффективности доставки программного обеспечения, который удовлетворяет изложенным нами требованиям. Он фокусируется на глобальных, системных целях и измеряет конечные результаты, над улучшением которых различные функциональные команды должны работать совместно. Следующий вопрос, на который мы хотели бы ответить: имеет ли значение эффективность доставки программного обеспечения?

Влияние эффективности доставки на организационную эффективность

Для оценки эффективности работы организации респондентам было предложено оценить относительную эффективность своей организации по нескольким параметрам: прибыльность, доля рынка и производительность. Это шкала, которая была многократно проверена в предыдущем исследовании (Уайденер, 2007). Было также установлено, что этот показатель эффективности деятельности организации в значительной степени коррелирует с показателями рентабельности инвестиций (ROI) и устойчив к экономическим циклам, — отличный показатель для наших целей. Анализ, проводимый в течение нескольких лет, показывает, что высокоэффективные организации неизменно в два раза чаще превышали свои цели, чем низкоэффективные. Это показывает, что возможности организации по доставке программного обеспечения могут фактически обеспечить конкурентное преимущество для вашего бизнеса.

В 2017 году мы в нашем исследовании также изучили, как ИТ-эффективность влияет на способность компании достигать организационных целей в широком смысле, то есть целей, которые выходят за рамки простых показателей прибыли и дохода. Независимо от того, стремитесь вы получать прибыль или нет, любая организация сегодня зависит от технологий достижения своей миссии и обеспечения ценности для своих клиентов или акционеров быстро, надежно и безопасно. Какова бы ни была миссия, эффективность работы технологий в организации влияет на ее общую эффективность.

Для измерения некоммерческих целей мы использовали шкалу, которая была многократно проверена и особенно хорошо подходит для этой цели (Кавалуццо и Иттнер, 2004). Мы обнаружили, что лидеры также в два раза чаще превышают цели по количеству товаров и услуг, операционной эффективности, удовлетворенности клиентов, качеству продуктов или услуг и достижению целей организации. Мы покажем эту взаимосвязь на рис. 2.4.

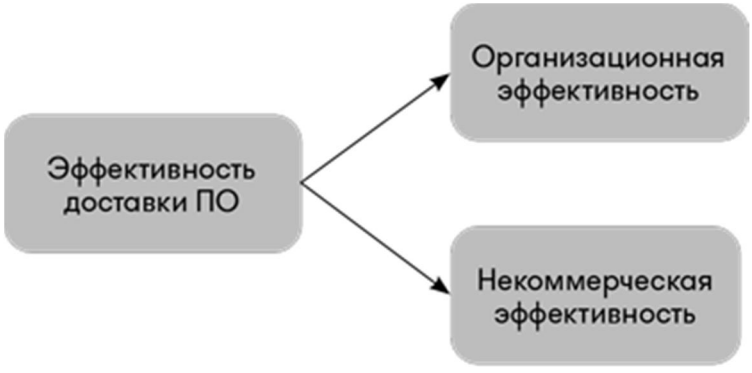


Рис. 2.4. **Влияние эффективности доставки ПО**

Как читать рисунки в этой книге

Мы включили в книгу рисунки, чтобы помочь вам на пути нашего исследования.

- Когда вы видите серое прямоугольное поле, — это конструкция, которую мы измерили. (Подробно о конструкциях см. в Главе 13.)
- Стрелки, связывающие поля, указывают на предиктивное отношение. Вы правильно прочитали: исследование в этой книге включает выводы, которые выходят за рамки корреляции и переходят в разряд прогнозов. (Подробнее см. Главу 12, посвященную прогнозированию на основе умозаключений.) Вы можете прочитать эти стрелки, используя слова «управляет», «прогнозирует», «влияет» или «оказывает воздействие». Это все положительные связи, если не указано иное.

Например, рис. 2.4 можно прочитать как «эффективность доставки программного обеспечения влияет на организационную и некоммерческую эффективность».

В софтверных компаниях способность работать и поставлять небольшие пакеты ПО особенно важна, поскольку она позволяет быстро собирать отзывы пользователей с помощью таких методов, как A/B-тестирование.

Стоит отметить, что способность к экспериментальному подходу к разработке продукта в значительной степени коррелирует с техническими практиками, которые способствуют непрерывной доставке.

Тот факт, что эффективность доставки программного обеспечения имеет большое значение, является убедительным аргументом против аутсорсинга разработки стратегического для вашего бизнеса ПО вместо внедрения этой возможности в ядро организации. Даже федеральное правительство США через такие инициативы, как Цифровая служба США и Служба трансформации технологий при Администрации общих служб, инвестировало в создание собственной разработки программного обеспечения для решения стратегических задач. Напротив, большинство программ, которые используют предприятия (например, офисные приложения и системы начисления заработной платы), не являются стратегическими и во многих случаях должны приобретаться по модели «программное обеспечение как услуга». Определение того, какое программное обеспечение является стратегическим, а какое нет, и управление ими соответствующим образом имеют огромное значение. Эта тема подробно рассматривается Саймоном Уордли, создателем метода отображения Уордли (Уордли, 2015).

Управление изменениями

Теперь, когда мы определили эффективность доставки программного обеспечения строгим и измеримым способом, мы можем принимать

основанные на фактических данных решения о том, как улучшить производительность команд, создающих программные продукты и услуги. Мы можем сравнить и оценить команды с более крупными организациями, в которых они работают, и с отраслью в более широком смысле. Мы можем измерить улучшение их показателей — или отставание — с течением времени. И, возможно, самое волнующее — мы можем выйти за рамки корреляции и начать тестировать прогнозирование. Мы можем проверить гипотезы о том, какие методы — от управления работой в процессе до автоматизации тестирования — на самом деле влияют на эффективность доставки и силу своего воздействия. Мы можем оценить другие конечные результаты, которые нам важны, такие как командное выгорание и «боль развертывания». Мы можем ответить на такие вопросы, как: «Действительно ли советы по управлению изменениями улучшают эффективность доставки?» (Осторожно, спойлер: нет, они отрицательно коррелируют со скоростью и стабильностью.)

Как мы покажем в следующей главе, можно также моделировать и измерять командную культуру количественными параметрами. Это позволяет нам оценить влияние DevOps и практик непрерывной доставки на культуру и, в свою очередь, влияние культуры на эффективность доставки ПО и организационную эффективность. Наша способность измерять деятельность, культуру и результаты является невероятно мощным инструментом, который может быть использован для достижения огромного положительного эффекта в погоне за высокой эффективностью.

Конечно, вы можете использовать эти инструменты для моделирования своей собственной эффективности. Используйте таблицу 2.3, чтобы узнать, в какой раздел нашей классификации вы попадаете. Используйте наши измерения для оценки времени выполнения, частоты развертывания, среднего времени восстановления и частоты отказов при изменениях, а также попросите свои команды установить целевые показатели для этих измерений.

Тем не менее важно осмотрительно использовать эти инструменты. В организациях с культурой обучения они невероятно сильны. Но в патологических и бюрократических организационных культурах оценка используется как форма контроля и люди скрывают информацию, которая бросает вызов существующим правилам, стратегиям и структурам власти. Как сказал Деминг, «всегда, когда есть страх, вы получаете неправильные цифры» (Хамбл и соавторы, 2014, с. 56). Прежде чем вы будете готовы применить научный подход к повышению эффективности, вы должны сначала понять и развить свою культуру. Именно к этой теме мы сейчас и обратимся.

Глава 3

Измерение и изменение культуры

Это практически прописная истина в кругах DevOps: культура имеет огромное значение. Однако культура неосязаема; существует множество определений и моделей культуры. Наша задача состояла в том, чтобы найти модель культуры, которая была бы четко определена в научной литературе, могла бы быть эффективно измерена и обладала бы прогностической силой в нашей области. Мы смогли не только добиться этих целей, но также обнаружили, что можно влиять на культуру и улучшать ее путем внедрения методов DevOps.

Моделирование и измерение культуры

В литературе существует множество подходов к моделированию культуры. Вы можете взглянуть на национальную культуру — например, той страны, в которой вы живете. Вы также можете говорить о том, какие принятые в организации культурные ценности влияют на поведение команд. И даже в рамках организационной культуры существует несколько способов определения и моделирования культуры. Организационная культура может существовать на трех уровнях: основные положения, ценности и артефакты (Шейн, 1985). На первом уровне основные положения формируются с течением времени — в процессе того, как члены группы или организации определяют смысл отношений, событий и действий. Эти интерпретации являются наименее видимыми из всех уровней — это то, что мы просто знаем, и то, что может оказаться трудно сформулировать после того, как мы достаточно долго проработали в команде.

Второй уровень организационной культуры — ценности, которые являются более видимыми для членов группы, поскольку эти коллективные ценности и нормы могут обсуждаться и даже оспариваться теми, кто их знает. Ценности являются своего рода объективом, через который члены группы просматривают и интерпретируют отношения, события и действия вокруг них. Ценности также влияют на групповые взаимодействия, устанавливая социальные нормы, которые формируют действия членов группы и обеспечивают контекстуальные правила (Бансал, 2003). Эти правила довольно часто формируют понятие «культура», о которой мы думаем, когда говорим о культуре команды и организации.

Третий уровень организационной культуры является наиболее заметным и может наблюдаться в артефактах. Они могут включать письменные заявления или убеждения, связанные с миссией компании, технологии, формальные процедуры или даже героев и ритуалы (Петтигрю, 1979).

Основываясь на дискуссиях в кругах DevOps и важности организационной культуры на втором уровне, мы решили выбрать

модель, определенную социологом Роном Веструмом. Веструм проводил исследования человеческого фактора в области безопасности систем, особенно в контексте аварий в технологических областях, которые являются весьма сложными и рискованными, таких как авиация и здравоохранение. В 1988 году он разработал типологию организационных культур (Веструм, 2014).

- Патологические (ориентированные на власть) организации характеризуются атмосферой страха и угрозы. Люди часто копят или скрывают информацию по политическим причинам или искажают ее, чтобы выставить себя в лучшем свете.
- Бюрократические (ориентированные на правила) организации защищают отделы. Те, кто в департаменте, хотят сохранить свою «территорию», настаивают на своих собственных правилах и обычно делают все по правилам — своим правилам.
- Производительные (ориентированные на результат) организации сосредоточены на миссии. Как мы достигаем нашей цели? Все подчинено эффективной работе — тому, что мы должны делать.

Следующая идея Веструма состояла в том, что организационная культура влияет на то, как информация распространяется внутри организации. Веструм приводит три характеристики хорошей информации:

1. Она дает ответы на вопросы, которые востребованы тем, кто их задает.
2. Она своевременна.
3. Она представлена таким образом, что может быть эффективно использована тем, кто ее получает.

Хорошая циркуляция информации имеет решающее значение для безопасной и эффективной работы в условиях высоких темпов и критических последствий, которые свойственны в том числе технологическим организациям. Веструм описывает характеристики организаций, которые подпадают под его три типа, в таблице 3.1.

Дополнительное открытие Веструма состояло в том, что определение типа организационной культуры предсказывает результаты работы. Мы включили это, в частности, потому, что мы так часто слышим, что культура важна в DevOps, и нам было интересно понять, может ли культура предсказать эффективность доставки программного обеспечения.

Таблица 3.1

Типология организационной культуры по Веструму

Патологические (ориентированные на власть)	Бюрократические (ориентированные на правила)	Производительные (ориентированные на результат)
Низкий уровень сотрудничества	Средний уровень сотрудничества	Высокий уровень сотрудничества
«Гонцов» с плохими новостями «пристреливают»	«Гонцов» игнорируют	«Гонцов» обучают
Уклонение от ответственности	Узкая область ответственности	Разделение рисков
Горизонтальные связи не допускаются	Горизонтальные связи допускаются	Горизонтальные связи поощряются
Ошибки ведут к поиску козла отпущения	Ошибки регулируются правилами	Ошибки ведут к исследованиям
Инновации подавляются	Инновации приводят к проблемам	Инновации внедряются

Измеряя культуру

Для того чтобы измерить культуру организаций, мы используем преимущества того факта, что указанные типы организаций формируют «точки на шкале» — «сплошную Веструма» (Веструм, 2014). Поэтому здесь отлично подходят вопросы по типу шкалы Ликерта. В психометрии шкалу Ликерта используют для оценки восприятия людей, предлагая им оценить, насколько они согласны или не согласны с утверждением. Когда люди отвечают на вопрос по шкале Ликерта, мы присваиваем ответу значение от 1 до 7, где 1 означает «категорически не согласен», а 7 означает «полностью согласен».

При таком подходе утверждение должно быть сформулировано предельно строго, чтобы люди могли решительно согласиться или не согласиться (или действительно чувствовать себя нейтрально) по этому поводу. На рисунке 3.1 вы можете увидеть часть опроса с утверждениями, которые мы создали из модели Веструма, вместе с вариантами ответов по шкале Ликерта.

Получив ответы на эти вопросы от нескольких человек (часто десятков или сотен), мы должны определить, является ли наша оценка организационной культуры достоверной и надежной со статистической точки зрения. То есть нам нужно выяснить, понимают ли вопросы одинаково все люди, принимающие участие в опросе, и действительно ли эти вопросы, взятые вместе, измеряют организационную культуру. Если анализ с использованием нескольких статистических тестов подтверждает эти свойства, мы называем то, что измерили, конструкцией (в этом случае нашей конструкцией будет

«организационная культура по Веструму»), и затем мы можем использовать эти показатели в дальнейших исследованиях.

	Категорически не согласен	Не согласен	Не согласен отчасти	Ни то, ни другое	Отчасти согласен	Согласен	Полностью согласен
Поиск информации ведется активно	○	○	○	○	○	○	○
Сотрудников не наказывают за сообщения об ошибках и другие плохие новости	○	○	○	○	○	○	○
Обязанности разделены	○	○	○	○	○	○	○
Кроссфункциональное сотрудничество поощряется и вознаграждается	○	○	○	○	○	○	○
Ошибки служат предметом изучения	○	○	○	○	○	○	○
Новые идеи приветствуются	○	○	○	○	○	○	○
Ошибки, как правило, рассматриваются как возможности для улучшения системы	○	○	○	○	○	○	○

Рис. 3.1. **Вопросы по шкале Ликерта для измерения культуры**

Анализ конструкций

Прежде чем проводить какой-либо анализ среди наших измерений (например, влияет ли организационная культура на эффективность доставки программного обеспечения), мы должны проанализировать данные и сами показатели. Надежные средства оценки в наших исследованиях мы называем конструкциями.

На этом первом этапе мы провели несколько анализов, чтобы убедиться, что наши методы были достоверными и надежными. Эти анализы включали тесты на дискриминантную валидность, конвергентную валидность и надежность.

- Дискриминантная валидность: проверка того, что элементы, которые не должны быть связаны, действительно не связаны (например, что элементы, которые, по нашему мнению, не отражают организационную культуру, на самом деле не связаны с ней).
- Конвергентная валидность: проверка того, что элементы, которые предполагаются связанными, действительно связаны (например, если предполагается, что показатели измеряют организационную культуру, они действительно измеряют ее).
- Надежность: проверка того, что утверждения прочитаны и одинаково интерпретированы всеми участниками опроса. Это также называется внутренней последовательностью.

Взятый вместе анализ достоверности (валидности) и надежности подтверждает наши методы и предшествует любому дополнительному анализу для проверки таких отношений, как корреляция или прогноз. Дополнительные сведения о достоверности и надежности см. в Главе 13. Дополнительную информацию о статистических испытаниях, используемых для подтверждения достоверности и надежности, можно найти в Приложении C.

Наше исследование последовательно выявило, что наша конструкция по модели Веструма — индикатор уровня организационной культуры, который определяет приоритет доверия и сотрудничества в команде, — является как достоверной, так и надежной[8].

Это означает, что вы тоже можете использовать эти вопросы в своих опросах. Чтобы вычислить балл для каждого ответа в опросе, возьмите числовое значение (1–7), соответствующее ответу на каждый вопрос, и вычислите среднее значение по всем вопросам. Затем вы можете выполнить статистический анализ всех ответов в целом.

Культура позволяет обрабатывать информацию с помощью трех механизмов. Во-первых, в организациях с производительной культурой люди сотрудничают более эффективно, и в них присутствует более высокий уровень доверия по всей организационной иерархии.

Во-вторых, «производительная культура подчеркивает миссию — акцент, который позволяет вовлеченным людям оставить в стороне свои личные проблемы, а также отраслевые проблемы, которые так очевидны в бюрократических организациях. Миссия первична. И в-третьих, созидательность поощряет равные условия, в которых иерархия играет не такую важную роль» (Веструм, 2014, с. 61).

Мы должны подчеркнуть, что бюрократия — это не обязательно плохо. Как отмечает Марк Шварц в книге The Art of Business Value, цель бюрократии — «обеспечить справедливость путем применения правил к управленческому поведению. Правила будут одинаковыми для всех случаев — так никто не получит преференциального или дискриминационного отношения. Кроме того, правила будут представлять собой лучшие продукты накопленных знаний организации: сформулированные бюрократами, которые являются экспертами в своих областях, правила будут насаждать эффективные структуры и процессы, гарантируя справедливость и устраняя произвол» (Шварц, 2016, с. 56).

Описание Веструмом культуры, ориентированной на правила, пожалуй, лучше всего рассматривать как культуру, в которой следование правилам считается более важным, чем достижение миссии. Мы работали с командами в федеральном правительстве США, которые без натяжки можно назвать производительными. И мы также работали со стартапами, которые были явно патологическими.

Что прогнозирует организационная культура Веструма

Теория Веструма утверждает, что организации с лучшим информационным потоком функционируют более эффективно. Согласно Веструму, этот тип организационной культуры имеет несколько важных предпосылок, что означает, что он является хорошим представителем характеристик, описанных этими предпосылками.

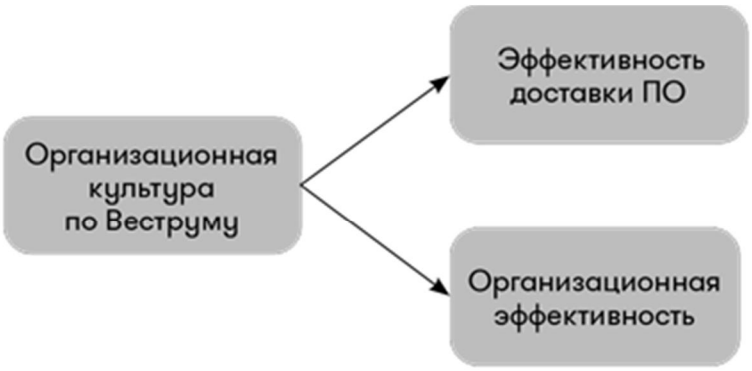


Рис. 3.2. **Результаты внедрения организационной культуры по Веструму**

Во-первых, хорошая культура требует доверия и сотрудничества между людьми по всей организации, поэтому она отражает уровень сотрудничества и доверия внутри организации.

Во-вторых, более высокая организационная культура может свидетельствовать о более качественном принятии решений. В команде с таким типом культуры не только больше информации доступно для принятия решений, но эти решения легче отменить, если они окажутся неверными, потому что команда, скорее всего, будет открытой и прозрачной, а не закрытой и иерархической.

Наконец, команды с такими культурными нормами, скорее всего, будут обладать лучшей атмосферой для людей, поскольку проблемы быстрее обнаруживаются и решаются.

Мы предположили, что культура будет предсказывать как эффективность доставки программного обеспечения, так и эффективность организации. Мы также предвидели, что это приведет к более высокому уровню удовлетворенности работой9. Обе эти гипотезы оказались верными. Эти взаимосвязи показаны на рис. 3.2.

Последствия теории Веструма для технологических организаций

Для современных организаций, которые надеются преуспеть перед лицом все более быстрых технологических и экономических изменений, важны как устойчивость, так и способность к инновациям через реагирование на эти изменения. Наше исследование применения теории

Веструма к технологиям показывает, что эти две характеристики связаны. Оно демонстрирует, что эта теория, изначально разработанная для прогнозирования результатов безопасности, также предсказывает как эффективность доставки ПО, так и организационную эффективность. Это имеет значение, потому что результаты безопасности — это результаты работы в медицинских учреждениях. Распространяя эту теорию на технологии, мы ожидали, что этот тип организационной культуры положительно повлияет на доставку программного обеспечения и эффективность организации. Это зеркально отражает исследование, проведенное Google с целью изучить, как создавать высокоэффективные команды.

Конструкция эффективности доставки

В Главе 2 мы говорили, что эффективность доставки сочетает в себе четыре показателя: время выполнения, частоту выпуска релизов, время восстановления службы и частоту отказов. При выполнении кластерного анализа все четыре метрики вместе значимым образом классифицируют и различают наших участников с высокими, средними и низкими показателями. То есть все четыре критерия хороши для категоризации команд. Однако, когда мы попытались превратить эти четыре метрики в конструкцию, мы столкнулись с проблемой: они не проходят все статистические тесты достоверности и надежности. Анализ показал, что только время выполнения, частота выпуска и время восстановления вместе образуют достоверную и надежную конструкцию. Таким образом, в остальной части книги, когда мы говорим об эффективности доставки программного обеспечения, она определяется только комбинацией этих трех показателей. Кроме того, когда показано, что эффективность доставки программного обеспечения коррелирует с какой-либо другой конструкцией, или когда мы говорим о прогнозах, связанных с эффективностью доставки программного обеспечения, мы говорим только о конструкции, определенной и измеренной таким образом.

Обратите внимание, однако, что частота сбоев при изменениях сильно коррелирует с конструкцией эффективности доставки программного обеспечения, что означает, что в большинстве случаев вещи, коррелирующие с конструкцией эффективности доставки программного обеспечения, также коррелируют с частотой отказов при изменениях.

Google хотела выяснить, есть ли какие-либо факторы, общие для ее наиболее эффективных команд. Они начали двухлетний исследовательский проект, чтобы изучить, что сделало команды Google эффективными, проведя «более 200 интервью… с сотрудниками и [имея перед глазами] более чем 250 характеристик более чем 180 активных команд Google» (Google, 2015). Они ожидали найти сочетание индивидуальных черт и навыков, которые были бы ключевыми компонентами высокоэффективных команд. Вместо этого они обнаружили, что «то, кто находится в команде, имеет меньшее значение, чем то, как члены команды взаимодействуют, структурируют свою работу

и оценивают свой вклад» (Google, 2015). Другими словами, все сводится к динамике команды.

Особенно показательно то, как организации справляются со сбоями или авариями. Патологические организации ищут козла отпущения: расследования направлены на то, чтобы найти человека или группу людей, ответственных за проблему, а затем наказать или обвинить их. Но в сложных адаптивных системах несчастные случаи почти никогда не происходят по вине одного человека, который ясно видел, что должно произойти, и не смог это предотвратить. Скорее, несчастные случаи, как правило, возникают в результате сложного взаимодействия факторов. Ошибка в сложных системах, как и другие типы поведения в таких системах, является эмерджентной, то есть возникшей из совокупности факторов (Перроу, 2011).

Таким образом, расследования несчастных случаев, которые останавливаются на человеческой ошибке, не просто плохи, но и опасны. Напротив, человеческая ошибка должна стать началом расследования. Наша цель должна состоять в том, чтобы выяснить, как мы можем улучшить поток информации, или найти лучшие инструменты, чтобы помочь предотвратить катастрофические сбои после явно заурядных рутинных действий.

Как мы меняем культуру?

Джон Шук, описывая свой опыт преобразования культуры команд на автомобильном заводе во Фремонте (Калифорния), заводе-основоположнике движения бережливого производства в США, написал: «…мой опыт научил меня, и это было действительно мощно, что изменить культуру — это не сначала изменить то, как люди думают, а вместо этого начать с изменения того, как люди себя ведут и что они делают» (Шук, 2010)10.

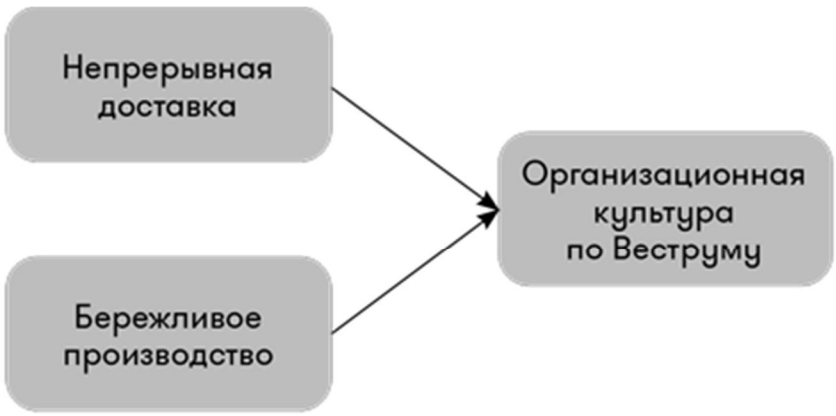


Рис. 3.3. **Движущие факторы организационной культуры по Веструму**

Таким образом, мы предполагаем, что, следуя теории, разработанной движениями Lean и Agile, внедрение практик этих движений может оказать влияние на культуру. Мы решили рассмотреть как технические,

так и управленческие практики, а также оценить их влияние на культуру. Наше исследование демонстрирует, что бережливое управление, как и ряд других технических практик, известных как непрерывная доставка (Хамбл и Фарлей, 2010), действительно влияет на культуру, как показано на рис. 3.3.

Вы можете проложить свой путь к лучшей культуре, применяя эти методы как в технологических организациях, так и в производственных. В следующей главе мы рассмотрим технические практики, а затем в Главах 7 и 8 обсудим управленческие практики.

Глава 4

Технические практики

На момент публикации Манифеста Agile (Agile Manifesto) в 2001 году экстремальное программирование (XP) было одним из самых популярных Agile-фреймворков11.

В отличие от Scrum XP предписывает ряд технических практик, таких как разработка на основе тестирования и непрерывная интеграция. Непрерывная доставка (Хамбл и Фарлей, 2010) также подчеркивает важность этих технических практик (в сочетании с комплексным управлением конфигурацией) как средства обеспечения более частых, более качественных и менее рискованных выпусков программного обеспечения.

Во многих внедрениях в рамках подхода Agile технические практики рассматриваются как второстепенные по сравнению с управленческими и командными, на которых делают акцент некоторые Agile-концепции. Наше исследование показывает, что техническая практика играет жизненно важную роль в достижении этих результатов.

В этой главе мы обсудим исследования, которые мы провели для того, чтобы измерить непрерывную доставку как возможность и оценить ее влияние на эффективность доставки ПО, организационную культуру и другие показатели результатов, такие как выгорание команды и «боль развертывания». Мы считаем, что практика непрерывной доставки действительно оказывает ощутимое влияние на эти результаты.

Что такое непрерывная доставка?

Непрерывная доставка — это набор возможностей, которые позволяют нам безопасно, быстро и устойчиво внедрять изменения всех видов: функции, изменения конфигурации, исправления ошибок, эксперименты, — в производство или непосредственно пользователям. В основе непрерывной доставки лежат пять ключевых принципов.

- Качество сборки. Третий из 14 пунктов Эдвардса Деминга для управления гласит: «Прекратите зависимость от экспертизы для достижения качества. Устраните необходимость проведения экспертизы на массовой основе, поставив на первое место повышение качества продукта» (Деминг, 2000). При непрерывной доставке мы инвестируем в создание культуры, поддерживаемой инструментами и людьми, в которой мы можем быстро обнаружить любые проблемы и устранить их, когда это еще обходится дешево.
- Работа небольшими партиями. Организации, как правило, планируют работу большими частями — будь то создание новых продуктов или услуг или инвестиции в организационные изменения. Разделяя работу на гораздо меньшие части, которые быстро обеспечивают измеримые результаты бизнеса для небольшой части нашего целевого рынка, мы получаем существенную обратную связь о работе, которую мы делаем, чтобы мы могли исправить курс. Даже при том, что работа в небольших партиях добавляет некоторые накладные расходы, она приносит огромные выгоды, позволяя нам избегать работы с нулевой или отрицательной ценностью для наших организаций.

  Ключевой целью непрерывной доставки является изменение экономики процесса доставки ПО, поэтому стоимость выпуска отдельных изменений очень низка.

- Компьютеры выполняют повторяющиеся задачи; люди решают проблемы. Одна из важных стратегий снижения затрат на выпуск изменений заключается в инвестировании в упрощение и автоматизацию повторяющейся работы, которая занимает много времени, такой как регрессионное тестирование и развертывание ПО. Таким образом мы освобождаем людей для более ценной работы по решению таких задач, как улучшение структуры наших систем и процессов в ответ на обратную связь.
- Неустанно проводить постоянные улучшения. Наиболее важной характеристикой высокоэффективных команд является то, что они никогда не бывают удовлетворены: они всегда стремятся стать лучше. В компаниях-лидерах улучшение является частью ежедневной работы каждого сотрудника.
- Каждый несет ответственность. Как мы узнали от Рона Веструма, в бюрократических организациях команды, как правило, сосредоточены на целях своего подразделения, а не на организационных целях. Таким образом, разработка фокусируется на пропускной способности, тестировании качества и операциях по обеспечению стабильности. Однако на самом деле все это результаты системного уровня и они могут быть достигнуты только при тесном сотрудничестве между всеми участниками процесса доставки программного обеспечения.

Ключевая цель руководства заключается в том, чтобы сделать состояние этих системных результатов прозрачным, работая с остальной частью организации для постановки измеримых, достижимых, привязанных к срокам целей для достижения этих результатов, а затем помочь своим командам работать над ними.

Чтобы осуществлять непрерывную доставку, необходимо создать следующие основы.

- Комплексное управление конфигурацией. Должна быть возможность обеспечить среду разработки и создавать, тестировать и развертывать наше программное обеспечение полностью автоматизированным способом исключительно из информации, хранящейся в системе управления версиями. Любые изменения в среде или программном обеспечении, которое на ней работает, должны применяться с использованием автоматизированного процесса из системы управления версиями. Это все еще оставляет место для ручных утверждений концепций, но, как только они утверждены, все изменения должны применяться автоматически.
- Непрерывная интеграция (НИ). Многие команды разработчиков программного обеспечения используются для разработки функций в ветках в течение нескольких дней или даже недель. Интеграция всех этих ветвей требует значительного времени и доработки. Следуя нашему принципу работы небольшими партиями и построения внутреннего качества продукта, высокоэффективные команды сохраняют короткие ветви (менее одного дня работы) и часто интегрируют их в магистраль. Каждое изменение запускает процесс сборки, который включает выполнение модульных тестов. Если в какой-либо части этого процесса происходит сбой, разработчики немедленно исправляют его.
- Непрерывное тестирование. Тестирование — это не то, что мы должны начинать только после того, как функция или релиз получает статус «разработка завершена» (dev complete). Поскольку тестирование жизненно важно, мы должны проводить его все время как неотъемлемую часть процесса разработки. Автоматические модульные и приемочные тесты должны выполняться на стадии каждой фиксации в системе контроля версий, чтобы дать разработчикам быструю обратную связь по их изменениям. Разработчики должны иметь возможность запускать все автоматические тесты на своих рабочих станциях для сортировки и исправления ошибок. Тестировщики должны проводить исследовательское тестирование непрерывно на стадии каждой новой сборки, чтобы выйти из НИ. Никто не должен утверждать, что работа сделана, пока все необходимые автоматические тесты не будут написаны и пройдены.

Реализация непрерывной доставки означает создание многочисленных циклов обратной связи для обеспечения того, чтобы высококачественное программное обеспечение доставлялось пользователям чаще и надежнее[12].

При правильной реализации процесс выпуска новых версий для пользователей должен быть рутинной деятельностью, которая может быть выполнена по требованию в любое время. Непрерывная доставка требует, чтобы разработчики и тестировщики, а также специалисты UX[13], менеджеры продуктов и сотрудники техподдержки эффективно сотрудничали на протяжении всего процесса доставки.

Влияние непрерывной доставки

В первых нескольких итерациях нашего исследования 2014–2016 годов мы смоделировали и измерили следующие возможности:

- использование системы управления версиями для кода приложения, конфигурации системы, конфигурации приложения, а также сценариев сборки и конфигурации;
- комплексную автоматизацию тестирования, надежную, легкую в исправлении ошибок и работающую постоянно;
- автоматизацию развертывания;
- непрерывную интеграцию;
- «сдвиг влево» по безопасности: включение групп безопасности и самой безопасности в процесс доставки программного обеспечения, а не в качестве нисходящей фазы;
- использование магистральной разработки в отличие от долгоживущих функциональных ветвей;
- эффективное управление тестовыми данными.

Большинство из этих возможностей измеряется в виде конструкций с использованием вопросов по шкале Ликерта[14]. Например, чтобы измерить возможности контроля версий, мы просим респондентов сообщить, в какой степени они согласны или не согласны со следующими утверждениями.

- Наш код приложения находится в системе контроля версий.
- Наши системные конфигурации находятся в системе контроля версий.
- Наши конфигурации приложений находятся в системе контроля версий.
- Наши скрипты для автоматизации сборки и конфигурации находятся в системе контроля версий.

Затем мы используем статистический анализ, чтобы определить, в какой степени эти возможности влияют на результаты, которые нам важны. Как

и ожидалось, в совокупности эти возможности оказывают сильное положительное влияние на эффективность доставки программного обеспечения. (Ниже в этой главе мы обсудим некоторые нюансы реализации этих практик.) Однако они также имеют и другие существенные преимущества: они помогают уменьшить «боль развертывания» и выгорание команды. Хотя мы слышали в организациях, с которыми мы работаем, интересные подтверждения преимуществ качества работы в течение многих лет, увидеть доказательства в данных было фантастическим. И это важно: мы ожидаем этого, потому что, когда команды практикуют непрерывную доставку (НД), развертывание в производство не является грандиозным событием уровня Большого взрыва — это то, что можно сделать в обычные рабочие часы в рамках стандартной повседневной работы. (Мы рассмотрим здоровье команды более подробно в Главе 9.) Интересно, что команды, хорошо справлявшиеся с непрерывной доставкой, сильнее идентифицировали себя с организацией, в которой они работали, — а это ключевой прогностический фактор организационной эффективности, который мы обсуждаем в Главе 10.

Как обсуждалось в Главе 3, мы предположили, что реализация НД повлияет на организационную культуру. Наш анализ показывает, что это действительно так. Если вы хотите улучшить свою культуру, вам поможет применение методов НД. Предоставляя разработчикам инструменты для обнаружения возникающих проблем, время и ресурсы для инвестирования в их развитие, а также полномочия для немедленного устранения проблем, мы создаем среду, в которой разработчики принимают ответственность за глобальные результаты, такие как качество и стабильность. Это оказывает положительное влияние на групповые взаимодействия и деятельность организационной среды и культуры членов команды.

В 2017 году мы расширили наш анализ и были более точны в том, как мы измеряли взаимосвязь между техническими возможностями, которые были важны для НД. Для этого мы создали конструкцию непрерывной доставки первого порядка. То есть мы измерили НД напрямую, что дало нам представление о способности команды достичь следующих результатов.

- Команды могут делать развертывание в производство (или конечным пользователям) по требованию на протяжении всего жизненного цикла доставки программного обеспечения.
- Быстрая обратная связь по качеству и развертываемости системы доступна всем в команде, и люди считают своим высшим приоритетом действия по этой обратной связи.

Наш анализ показал, что исходные возможности, измеренные в 2014–2016 годах, оказали сильное и статистически значимое влияние на эти результаты15. Мы также измерили две новые возможности, которые

также оказали сильное и статистически значимое влияние на непрерывную доставку:

- слабосвязанную, хорошо инкапсулированную архитектуру (подробнее это обсуждается в Главе 5);
- команды, которые могут выбрать свои собственные инструменты на основе того, что лучше для пользователей этих инструментов.

Мы покажем эту взаимосвязь на рис. 4.1.

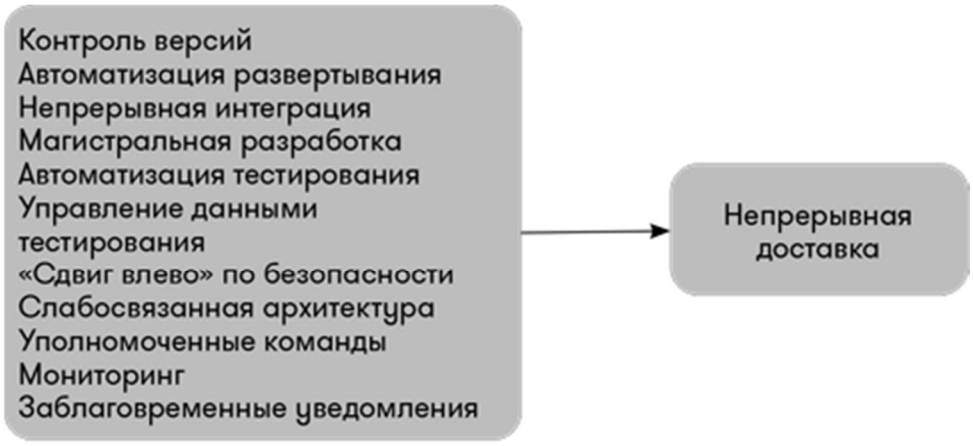


Рис. 4.1. **Движущие факторы непрерывной доставки**

Поскольку просто достижения непрерывной доставки ради самой непрерывной доставки недостаточно, мы хотели исследовать ее влияние на организации. Мы предположили, что это должно привести к повышению эффективности при доставке программного обеспечения, и предыдущие исследования показали, что это может даже улучшить культуру. Как и прежде, мы обнаружили, что команды, которые хорошо справлялись с непрерывной доставкой, достигали следующих результатов:

- сильная идентификация с организацией, в которой они работают (см. Главу 10);
- более высокие уровни эффективности доставки ПО (время выполнения, частота развертывания, время восстановления службы);
- более низкий процент отказов при изменениях;
- производительная, ориентированная на результат культура (см. Главу 3).

Эти взаимосвязи показаны на рис. 4.2.

Рис. 4.2. **Влияние непрерывной доставки**

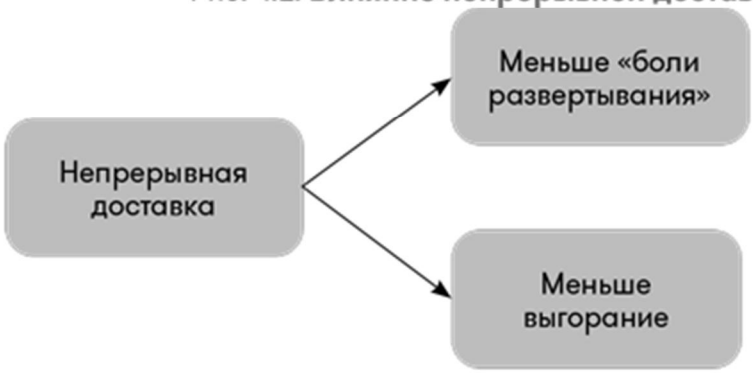


Рис. 4.3. **Непрерывная доставка делает работу более устойчивой**

Более того, наше исследование показало, что улучшения в НД принесли плюсы в том, как люди чувствовали себя на работе. Это означает, что инвестиции в технологии также являются инвестициями в людей и эти инвестиции сделают наш технологический процесс более устойчивым (рис. 4.3). Таким образом, НД помогает нам достичь одного из двенадцати принципов Манифеста Agile: «Процессы Agile способствуют устойчивому развитию. Организаторы, разработчики и пользователи должны иметь возможность поддерживать постоянство темпа бесконечно» (Бек и соавторы, 2001).

- Меньше «боли развертывания».
- Снижение командного выгорания (см. Главу 9).

Влияние непрерывной доставки на качество

Важнейший вопрос, который мы хотели бы решить: повышает ли непрерывная доставка качество? Для того чтобы ответить на этот вопрос, мы сначала должны найти способ измерения качества. Это сложно, потому что качество очень контекстуально и субъективно. Как говорит эксперт по качеству ПО Джерри Вайнберг, «качество — это ценность для какого-то человека» (Вайнберг, 1992, с. 7).

Мы уже знаем, что непрерывная доставка предсказывает более низкий процент сбоев при изменениях, что является важным показателем качества. Тем не менее мы также протестировали несколько дополнительных косвенных переменных для оценки качества:

- качество и производительность приложений — как они воспринимаются теми, кто работает над ними;
- процент времени, затраченного на доработку или незапланированную работу;
- процент времени, затраченного на работу с ошибками, выявленными конечными пользователями.

Наш анализ показал, что все показатели коррелировали с эффективностью доставки программного обеспечения. Однако самая сильная корреляция была замечена в процентах времени, затраченного на доработку или незапланированную работу, включая работу по прерыванию/исправлению, аварийные развертывания ПО и патчи, ответы на срочные запросы на аудиторскую документацию и т.д. Кроме того, непрерывная доставка прогнозирует более низкие уровни незапланированной работы и доработок статистически значимым образом. Мы обнаружили, что количество времени, затраченное на новую работу, незапланированную работу или доработку и на другие виды работ, значительно различается среди участников с высокими и низкими показателями эффективности. Мы покажем эти различия на рис. 4.4.

Участники с высокими показателями сообщили, что тратят 49% времени на новую работу и 21% на незапланированную работу или доработку. Напротив, участники с низкими показателями тратят 38% своего времени на новую работу и 27% на незапланированную работу или доработку.

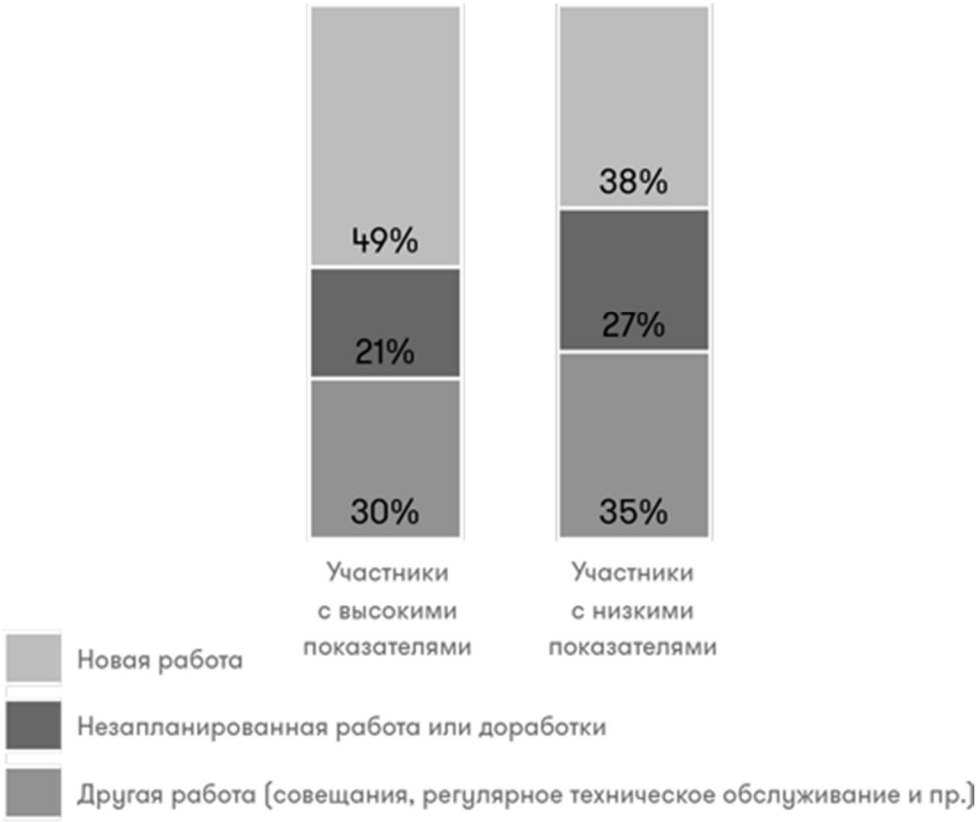


Рис. 4.4. **Новая работа vs незапланированная работа**

Незапланированная работа и доработка являются полезными индикаторами качества, потому что они представляют собой неспособность встроить качество в наши продукты. В книге The Visible Ops Handbook незапланированная работа описана как разница между тем, как если «обратить внимание на сигнал низкого уровня топлива в автомобиле или остаться без топлива на шоссе» (Бер и соавторы, 2004). В первом случае организация может устранить проблему плановым образом, без особой срочности или нарушения других запланированных работ. Во втором случае они вынуждены решать проблему в очень срочном порядке, часто задействуя все ресурсы, — представьте, что шесть инженеров должны бросить все и бежать по шоссе с полными газовыми баллонами, чтобы заправить застрявший грузовик.

Аналогично Джон Седдон, создатель метода Vanguard, подчеркивает важность снижения того, что он называет неудовлетворенным спросом — спросом на работу, вызванным неспособностью сделать все правильно с первого раза. Это одна из ключевых целей непрерывной доставки с ее акцентом на работу в небольших партиях с непрерывным тестированием в процессе.

Практика непрерывной доставки: что работает, а что нет

В нашем исследовании мы обнаружили девять ключевых возможностей, которые управляют непрерывной доставкой. Они перечислены ранее в этой главе. Некоторые из этих возможностей имеют интересные нюансы, которые мы обсудим в этом разделе — за исключением архитектуры и выбора инструментов, которые заслуживают отдельной главы (Глава 5). Непрерывная интеграция и автоматизация развертывания не рассматриваются далее в этой главе.

Контроль версий

Всестороннее использование контроля версий является относительно бесспорным. Мы спросили, сохраняют ли респонденты код приложения, конфигурацию системы, конфигурацию приложения и сценарии для автоматизации сборки и конфигурации в системе управления версиями. Эти факторы вместе предсказывают эффективность ИТ и формируют ключевой компонент непрерывной доставки. Наиболее интересно то, что сохранение конфигурации системы и приложения в системе управления версиями было более тесно связано с эффективностью доставки программного обеспечения, чем сохранение кода приложения в системе управления версиями. Конфигурация обычно считается второстепенной проблемой по сравнению с кодом приложения в управлении конфигурацией, но наше исследование показывает, что это ошибочное представление.

Автоматизация тестирования

Как уже обсуждалось выше, автоматизация тестирования является ключевой частью непрерывной доставки. Основываясь на нашем анализе, следующие методы предсказывают эффективность ИТ.

- Наличие надежных автоматических тестов: при их выполнении команды уверены, что их программное обеспечение готово к релизу. Кроме того, они уверены, что провалы тестов указывают на реальную ошибку. Слишком много тестовых наборов являются слоеными и ненадежными, производя ложные положительные и отрицательные результаты, поэтому имеет смысл инвестировать постоянные усилия в один надежный набор. Один из способов добиться этого — поместить ненадежные автоматические тесты в отдельный набор «на карантине», который выполняется независимо16. Или, конечно, вы можете просто удалить их. Если они управляются версиями (как и должно быть), вы всегда можете их вернуть.
- Разработчики в основном создают и поддерживают приемочные тесты, и они могут легко воспроизводить и фиксировать их на своих рабочих станциях разработки. Интересно отметить, что наличие автоматических тестов, созданных и поддерживаемых главным

образом командой обеспечения качества (QA — quality assurance) или аутсорсинговой стороной, не коррелирует с ИТ-эффективностью. Теория, лежащая в основе этого, заключается в том, что когда разработчики участвуют в создании и поддержании приемочных тестов, возникает два важных следствия. Во-первых, код становится проще тестировать, когда разработчики пишут тесты. Это одна из основных причин, почему разработка на основе тестирования (TDD — test-driven development) является важной практикой: она заставляет разработчиков создавать проекты, которые легче поддаются тестированию. Во-вторых, когда разработчики отвечают за автоматические тесты, они больше заботятся о них и будут вкладывать больше усилий в их поддержание и исправление.

Это не значит, что мы должны избавиться от тестеров. Тестировщики играют важную роль в жизненном цикле доставки программного обеспечения, выполняя ручное тестирование: исследовательское, приемочное и тестирование юзабилити — а также помогая разработчикам создавать и развивать наборы автоматических тестов.

После того как вы получили эти автоматические тесты, наш анализ показывает, что важно регулярно запускать их. Каждая фиксация кода должна инициировать сборку программного обеспечения и запуск набора быстрых автоматических тестов.

Разработчики должны получать обратную связь от более полного набора приемочных и эксплуатационных тестов. Более того, текущие сборки должны быть доступны тестировщикам для исследовательского тестирования.

Управление тестовыми данными

При создании автоматических тестов управление тестовыми данными может быть затруднено. По нашим данным, успешные команды имели достаточные тестовые данные для запуска своих полностью автоматизированных наборов тестов и могли получать тестовые данные для запуска автоматических тестов по требованию. Кроме того, тестовые данные не являются ограничением для автоматических тестов, которые они могут запускать.

Магистральная разработка

Наше исследование также показало, что магистральная разработка, в отличие от разработки на долгоживущих функциональных ветвях, коррелировало с более высокой эффективностью доставки. Успешные команды сохраняли в работе менее трех активных ветвей в любой момент времени, их ветви имели очень короткие периоды жизни (менее одного дня) до слияния в ствол и никогда не имели периодов

«замораживания кода» или стабилизации. Стоит еще раз подчеркнуть, что эти результаты не зависят от размера команды, размера организации или отрасли.

Даже после обнаружения того, что методы разработки на основе магистрали способствуют повышению эффективности доставки программного обеспечения, некоторые разработчики, которые привыкли к GitHub Flow17, сохраняют скептицизм. Этот рабочий процесс в значительной степени зависит от разработки с ветвями и только периодически сливается с магистралью.

Мы слышали, например, что стратегии ветвления эффективны, если команды разработчиков не поддерживают ветви слишком долго, — и мы согласны с тем, что работа над короткоживущими ветвями, которые объединяются в магистраль по крайней мере ежедневно, согласуется с общепринятой практикой непрерывной интеграции. Мы провели дополнительные исследования и обнаружили, что команды, использующие ветви, которые живут короткий промежуток времени (время интеграции меньше дня) в сочетании с короткими периодами слияния и интеграции (менее дня), с точки зрения эффективности доставки программного обеспечения лучше, чем команды, использующие долгоживущие ветви. Основываясь на доказательствах и нашем собственном опыте, мы предполагаем, что это происходит потому, что наличие нескольких долгоживущих ветвей препятствует как рефакторингу, так и внутрикомандной коммуникации. Однако следует отметить, что GitHub Flow подходит для проектов с открытым исходным кодом, участники которых не работают над проектом полный рабочий день. В этом случае имеют смысл ветви, живущие в течение длительного периода времени без слияния.

Информационная безопасность

Высокоэффективные команды более склонны включать информационную безопасность в процесс доставки. Их сотрудники по информационной безопасности обеспечивали обратную связь на каждом этапе жизненного цикла доставки программного обеспечения, от проектирования демоверсий до помощи с автоматизацией тестирования. Однако они сделали это таким образом, чтобы не замедлять процесс разработки, интегрируя проблемы безопасности в повседневную работу команд. На самом деле интеграция этих методов безопасности способствовала эффективности доставки программного обеспечения.

Внедрение непрерывной доставки

Наши исследования показывают, что технические практики непрерывной доставки оказывают огромное влияние на многие аспекты организации. Непрерывная доставка улучшает эффективность и качество доставки, а

также помогает улучшить культуру и уменьшить выгорание и проблемы развертывания ПО. Однако реализация этих методов часто требует переосмысления всего — от того, как команды работают, до того, как они взаимодействуют друг с другом, какие инструменты и процессы они используют. Это также требует значительных инвестиций в автоматизацию тестирования и развертывания в сочетании с неустанной работой по упрощению архитектуры систем на постоянной основе, чтобы гарантировать, что эта автоматизация не слишком дорога с точки зрения создания и обслуживания.

Таким образом, критическим препятствием для реализации непрерывной доставки является архитектура предприятия и самого приложения. Результаты нашего исследования этой важной темы мы обсудим в Главе 5.

Глава 5

Архитектура

Мы уже видели, что внедрение практик непрерывной доставки повышает эффективность доставки, влияет на культуру и уменьшает выгорание и «боль развертывания» ПО. Однако архитектура вашего программного обеспечения и сервисов, от которых оно зависит, может быть существенным препятствием для увеличения как скорости, так и стабильности процесса выпуска релизов и поставляемых систем.

Кроме того, DevOps и непрерывная доставка возникли в веб-системах, поэтому законно спросить, могут ли они быть применены к системам мейнфреймов, микропрограммному обеспечению или к средней корпоративной среде с большим количеством систем с нераспознаваемой структурой (Фуут и Йодер, 1997), состоящей из тысяч тесно связанных систем.

Мы решили выяснить влияние архитектурных решений и ограничений на эффективность доставки, а также то, что делает архитектуру эффективной. Мы обнаружили, что высокая эффективность возможна со всеми видами систем при условии, что системы и команды, которые их строят и поддерживают, слабо связаны.

Это ключевое архитектурное свойство позволяет командам легко тестировать и развертывать отдельные компоненты или службы, даже когда организация и количество систем, которыми она управляет, растут, то есть это позволяет организациям увеличивать свою производительность по мере их масштабирования.

Типы систем и эффективность доставки

Мы изучили большое количество типов систем, чтобы выяснить, существует ли корреляция между типом системы и эффективностью команды. Мы рассмотрели следующие типы систем как в качестве основной системы в разработке, так и в качестве интегрируемой службы:

- гринфилд (greenfield): новые системы, которые еще не были выпущены;
- системы взаимодействия (используются непосредственно конечными пользователями);
- системы записи (используются для хранения критически важной для бизнеса информации, когда жизненно необходимы согласованность и целостность данных);
- пользовательское программное обеспечение, разработанное другой компанией;
- пользовательское программное обеспечение, разработанное внутри компании;
- упакованное, коммерчески готовое программное обеспечение;
- встроенное программное обеспечение, которое работает на заводском аппаратном оборудовании (manufactured hardware device);
- программное обеспечение с компонентом, установленным пользователем (в том числе мобильные приложения);
- программное обеспечение вне мейнфреймов, которое работает на серверах другой компании;
- программное обеспечение вне мейнфреймов, которое работает на внутренних серверах;
- программное обеспечение для мейнфреймов.

Мы обнаружили, что участники исследования с низкими показателями с большей вероятностью говорили, что программное обеспечение, которое они создают, или набор сервисов, с которыми они должны взаимодействовать, — это пользовательское ПО, разработанное другой компанией (например, партнером на аутсорсинге). Они же с большей вероятностью работали над системами мейнфреймов. Интересно, что необходимость интеграции с системами мейнфреймов существенно не коррелировала с эффективностью.

В остальных случаях не было выявлено значимой корреляции между типом системы и эффективностью доставки. Мы были удивлены: мы ожидали, что команды, работающие над упакованным ПО, системами записи или встроенными системами, будут хуже, а команды, работающие над системами взаимодействия и системами гринфилд, — лучше. Данные показывают, что это не так.

Это делает еще более важным фокус на архитектурных характеристиках, обсуждаемых ниже, а не на деталях реализации вашей архитектуры. Достичь этих характеристик можно даже с помощью упакованного программного обеспечения и устаревших систем мейнфреймов — и,

наоборот, использование новейшей архитектуры микросервисов, развернутой на контейнерах, не гарантирует более высокой производительности, если вы игнорируете эти характеристики.

Как мы уже говорили в Главе 2, учитывая, что эффективность доставки программного обеспечения влияет на эффективность организации, важно инвестировать в собственные возможности для создания и развития основных, стратегических программных продуктов и услуг, которые обеспечивают ключевое отличие вашего бизнеса. Тот факт, что компании с низкой эффективностью чаще использовали или совмещали с существующим заказное программное обеспечение, разработанное сторонней компанией, лишь подчеркивает важность создания этой возможности внутри организации.

Сфокусируйтесь на развертываемости и тестируемости

Хотя в большинстве случаев тип системы, которую вы строите, не важен с точки зрения достижения высокой эффективности, есть две архитектурные характеристики, которые важны. Те, кто согласился со следующими утверждениями, с большей вероятностью попадали в группу с высокими показателями:

- мы можем выполнить большую часть нашего тестирования, не требуя интегрированной среды18;
- мы можем сделать и делаем развертывание или выпуск нашего приложения независимо от других приложений/сервисов, от которых оно зависит.

Представляется, что эти характеристики архитектурных решений, которые мы называем тестируемостью и развертываемостью, важны для создания высокой эффективности. Для достижения этих характеристик системы разработки слабо связаны, то есть могут быть изменены и проверены независимо друг от друга. В обзоре 2017 года мы провели анализ, чтобы проверить, насколько слабосвязанная, хорошо инкапсулированная архитектура влияет на эффективность ИТ. Мы обнаружили, что влияет. Действительно, самый большой вклад в непрерывную доставку в анализе 2017 года (даже больше, чем автоматизация тестирования и развертывания) состоит в том, могут ли команды:

- вносить масштабные изменения в разработку своей системы без разрешения кого-либо вне команды;
- вносить масштабные изменения в разработку своей системы, независимо от того, внесли ли изменения в собственные системы

другие команды, и не создавая при этом значительный объем работы для других команд;
- завершить свою работу без коммуникации и координации с людьми вне своей команды;
- осуществить развертывание и выпуск продукта или услуги по требованию независимо от других сервисов, от которых они зависят;
- провести большую часть тестирования по требованию, не требуя интегрированной среды;
- выполнить развертывание в обычное рабочее время с незначительным временем простоя.

В командах, получивших высокие оценки по архитектурным возможностям, требуется немного коммуникации между группами доставки для выполнения своей работы, а архитектура системы предназначена для того, чтобы команды могли тестировать, развертывать и изменять свои системы независимо от других команд. Другими словами, архитектура и команды слабо связаны. Для этого мы также должны обеспечить кроссфункциональную работу групп доставки, обладающих всеми навыками, необходимыми для проектирования, разработки, тестирования, развертывания и эксплуатации системы в одной и той же команде.

Эта связь между пропускной способностью связи и архитектурой систем была впервые затронута Мелвином Конвеем, который сказал: «Организации, которые разрабатывают системы… вынуждены производить конструкции, которые являются копиями коммуникационных структур этих организаций» (Конвей, 1968). Наше исследование поддерживает то, что иногда называют «обратным маневром Конвея»19, который гласит, что организации должны развивать свою команду и организационную структуру для достижения желаемой архитектуры. Цель состоит в том, чтобы ваша архитектура поддерживала способность команд выполнять свою работу — от проектирования до развертывания — без необходимости использования многоканальной связи между командами.

Архитектурные подходы, позволяющие использовать эту стратегию, включают использование ограниченных контекстов и API-интерфейсов как способов разделения больших областей на более мелкие слабосвязанные блоки, а также использование тестовых двойников и виртуализации для изолированного тестирования служб или компонентов. Предполагается, что сервис-ориентированная архитектура обеспечивает эти результаты, как и любая истинная архитектура микросервисов. Однако при реализации таких архитектур важно быть очень строгим в отношении этих результатов. К сожалению, в реальной жизни многие так называемые сервис-ориентированные архитектуры не позволяют тестировать и развертывать службы независимо друг от друга

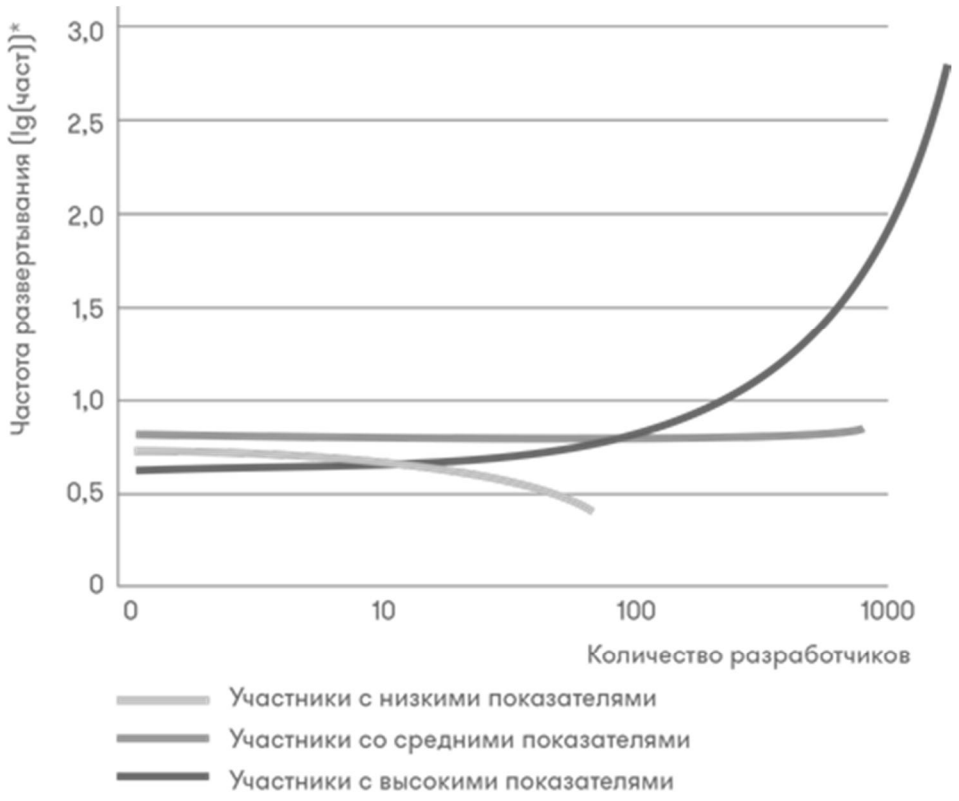
и таким образом не позволяют командам достичь более высокой производительности[20].

Конечно, DevOps — это прежде всего лучшее взаимодействие между командами, и мы не хотим сказать, что команды не должны работать вместе. Цель слабосвязанной архитектуры состоит в том, чтобы гарантировать, что каналы коммуникации не перегружены принятием мелких решений и мы можем вместо этого использовать эти каналы для обсуждения общих целей более высокого уровня и способов их достижения.

Слабосвязанная архитектура обеспечивает масштабирование

Если мы достигаем слабосвязанной, хорошо инкапсулированной архитектуры с соответствующей организационной структурой, происходят две важные вещи. Во-первых, мы можем добиться лучшей эффективности доставки, увеличивая как скорость, так и стабильность, одновременно уменьшая выгорание и проблемы развертывания. Во-вторых, мы можем существенно увеличить размер нашей инженерной организации и повысить производительность линейно (или даже больше, чем линейно) по мере того, как мы это делаем.

Чтобы измерить производительность, мы рассчитали следующий показатель из наших данных: количество развертываний в день на одного разработчика. Ортодоксальная точка зрения на масштабирование команд разработчиков ПО гласит, что при добавлении разработчиков в команду может повыситься общая производительность, а производительность отдельных разработчиков фактически снизится из-за «накладных расходов» на коммуникацию и интеграцию. Однако при взгляде на количество развертываний в день на одного разработчика для респондентов, которые осуществляют развертывание хотя бы один раз в день, мы можем увидеть следующие результаты на рис. 5.1.



* Чем выше, тем чаще: 1 = ежедневно.

Рис. 5.1. **Развертывания на одного разработчика в день**

По мере увеличения числа разработчиков мы обнаружили следующее.

- Участники из группы с низкими показателями эффективности сокращают частоту развертывания кода.
- Участники из группы со средними показателями развертывают код с постоянной частотой.
- Участники из группы с высокими показателями значительно увеличивают частоту развертывания кода.

Фокусируясь на факторах, которые прогнозируют высокую эффективность доставки: созидательной, ориентированной на цели культуре, модульной архитектуре, инженерных практиках, обеспечивающих непрерывную доставку, и эффективном лидерстве, — мы можем масштабировать развертывания на одного разработчика в день пропорционально количеству разработчиков (или больше). Это позволяет нашему бизнесу двигаться быстрее, когда мы берем больше людей, а не замедляться, как это обычно происходит.

Позволить командам выбирать свои собственные инструменты

Во многих организациях инженеры должны использовать инструменты и фреймворки из утвержденного списка. Этот подход обычно служит одной или нескольким из следующих целей:

- уменьшение сложности среды;
- обеспечение необходимых навыков для управления технологией на протяжении всего ее жизненного цикла;
- увеличение покупательной способности с поставщиками;
- обеспечение правильного лицензирования на все технологии.

Однако у этого недостатка гибкости есть и обратная сторона: она не позволяет командам выбирать технологии, наиболее подходящие для их конкретных потребностей, и экспериментировать с новыми подходами и парадигмами для решения своих проблем.

Наш анализ показывает, что выбор инструмента является важной частью технической работы. Когда команды могут решить, какие инструменты использовать, это способствует эффективности доставки программного обеспечения и, в свою очередь, организационной эффективности. Это неудивительно. Технические специалисты, которые разрабатывают и доставляют программное обеспечение и управляют сложными инфраструктурами, выбирают инструменты на основе того, что лучше всего подходит для завершения их работы и поддержки пользователей. Аналогичные результаты были получены и в других исследованиях технических специалистов (например, Форсгрен и соавторы, 2016), предполагающих, что преимущества делегирования выбора инструментов командам могут перевесить недостатки.

Тем не менее есть место и для стандартизации, особенно вокруг архитектуры и конфигурации инфраструктуры. Преимущества стандартизированной операционной платформы подробно обсуждаются в работе Хамбла (2017).

Другим примером служит описание Стивом Йегге перехода Amazon на сервис-ориентированную архитектуру (COA), в котором он отмечает: «Отладка проблем с чужим кодом становится намного сложнее и в принципе невозможна, если нет универсального стандартного способа запуска каждой службы в отладочной "песочнице"» (Йегге, 2011).

Еще один вывод нашего исследования заключается в том, что команды, которые встраивают системы безопасности в свою работу, также лучше справляются с непрерывной доставкой. Ключевым элементом этого является обеспечение того, чтобы команды информационной безопасности делали предварительно утвержденные и простые в использовании библиотеки, пакеты, наборы инструментов и процессы доступными для разработчиков и техподдержки.

Здесь нет никакого противоречия. Когда предоставленные инструменты действительно будут облегчать жизнь инженерам, которые их используют, они примут их добровольно. Это гораздо лучше, чем

заставлять их использовать инструменты, которые были выбраны для удобства других заинтересованных сторон. Фокус на удобстве использования и удовлетворенности клиентов так же важен при выборе или создании инструментов для внутренних клиентов, как и при создании продуктов для внешних клиентов, так что, позволяя вашим инженерам выбирать, использовать их или нет, вы гарантируете, что остаетесь честными в этом отношении.

Архитекторы должны фокусироваться на инженерах и результатах, а не на инструментах или технологиях

Дискуссии вокруг архитектуры часто фокусируются на инструментах и технологиях. Следует ли организации использовать микросервисы или бессерверные архитектуры? Должны ли они использовать Kubernetes или Mesos? На каком сервере, языке или платформе непрерывной интеграции они должны быть стандартизированы? Наше исследование показывает, что это не те вопросы, на которых следует фокусироваться.

Какие инструменты или технологии вы используете, не имеет значения, если люди, которые должны их использовать, ненавидят работать с ними или если они не достигают конечных результатов и не дают нужных нам моделей поведения. Важно, чтобы команды могли вносить изменения в свои продукты или услуги независимо от других команд или систем. Архитекторы должны тесно сотрудничать со своими пользователями-инженерами, создающими и эксплуатирующими системы, с помощью которых организация выполняет свою миссию, чтобы помочь им достичь лучших результатов и предоставить им инструменты и технологии, которые позволяют их достигать.

Глава 6

Интеграция информационной безопасности в жизненный цикл доставки

Возможно, название движения DevOps выбрано неудачно — оно игнорирует такие функции, как тестирование, управление продуктами и информационная безопасность. Первоначальной целью движения DevOps было — отчасти — объединить разработчиков и команды техподдержки для создания взаимовыгодных решений в погоне за целями системного уровня, вместо того чтобы перебрасывать работу через стену и указывать пальцами друг на друга, когда что-то пошло не

так. Однако этот тип поведения не ограничивается только разработкой и техподдержкой. Он возникает там, где различные функции в потоке создания ценности доставки программного обеспечения не работают эффективно вместе. Это особенно верно при обсуждении роли команд информационной безопасности. Информационная безопасность (infosec) является жизненно важной функцией в эпоху, когда угрозы повсеместны и постоянны. Тем не менее команды информационной безопасности часто плохо укомплектованы. Джеймс Уикетт, руководитель исследований в Signal Sciences, приводит соотношение: 1 специалист по информационной безопасности на 10 сотрудников инфраструктуры и 100 разработчиков в крупных компаниях (Уикетт, 2014). При этом они обычно участвуют только в конце жизненного цикла доставки программного обеспечения, когда уже часто мучительно больно и дорого вносить изменения, необходимые для повышения безопасности. Кроме того, многие разработчики не знают об общих рисках безопасности, таких как OWASP Top 1021, и как их предотвратить.

Наше исследование показывает, что внедрение безопасности в процесс разработки программного обеспечения не только повышает эффективность доставки, но и улучшает качество безопасности. Организации с высокой эффективностью доставки тратят значительно меньше времени на устранение проблем безопасности.

«Сдвиг влево» по безопасности

Мы обнаружили, что когда команды «сдвигаются влево» по информационной безопасности, то есть когда они встраивают ее в процесс доставки программного обеспечения, вместо того чтобы делать отдельной фазой, которая происходит после процесса разработки, это положительно влияет на их способность осуществлять непрерывную доставку. Что, в свою очередь, положительно сказывается на ее эффективности.

Что означает «сдвиг влево»? Во-первых, проверки безопасности проводятся для всех основных функций, и этот процесс выполняется таким образом, что он не замедляет процесс разработки. Как мы можем гарантировать, что такое внимание вопросам безопасности не снизит производительность разработки? Это то, на что направлен второй аспект этой возможности: информационная безопасность должна быть неотъемлемой частью всего жизненного цикла доставки программного обеспечения от разработки до эксплуатации. Это означает, что эксперты infosec должны вносить свой вклад в процесс разработки приложений, присутствовать и предоставлять обратную связь по демонстрациям программного обеспечения, а также обеспечивать тестирование функций безопасности в рамках автоматизированного набора тестов. Наконец, мы хотим, чтобы разработчикам было легко делать все правильно, когда дело доходит до infosec. Это может быть достигнуто путем обеспечения

простых в использовании, предварительно утвержденных библиотек, пакетов, наборов инструментов и процессов, доступных для разработчиков и служб техподдержки.

То, что мы видим здесь, — это переход от ситуации, когда команды infosec сами проводят проверку безопасности, к предоставлению разработчикам средств для создания встроенной безопасности. Это отражает две реальности.

Во-первых, гораздо легче убедиться, что люди, создающие программное обеспечение, делают все правильно, чем проверять почти завершенные системы и функции, чтобы найти значительные архитектурные проблемы и ошибки, которые требуют существенной переработки.

Во-вторых, команды информационной безопасности просто не имеют возможности проводить проверки безопасности при частых развертываниях. Во многих организациях безопасность и соответствие требованиям являются серьезным узким местом для перевода систем из статуса «разработка завершена» (dev complete) в жизнь. Привлечение специалистов infosec на протяжении всего процесса разработки также положительно влияет на коммуникационные и информационные потоки, а это и есть взаимовыгодная и основная цель DevOps.

Соблюдение требований в федеральном правительстве

Федеральные информационные системы подпадают под действие Федерального закона об управлении информационной безопасностью 2002 года (FISMA). FISMA требует, чтобы федеральные агентства следовали системе управления рисками[22], утвержденной NIST[23]. Система управления рисками включает в себя несколько этапов, таких как подготовка плана безопасности системы, который документирует, как был реализован соответствующий контроль информационной безопасности (325 элементов контроля для систем умеренного воздействия), а затем оценка и отчет по ее результатам[24], который документирует одобрение этой реализации. Этот процесс может занять от нескольких месяцев до более чем года и часто только начинается после стадии «разработка завершена» (dev complete).

В целях сокращения времени и затрат на доставку федеральных информационных систем небольшая группа государственных гражданских служащих в отделе 18F создала платформу в виде службы под названием cloud.gov на основе открытой версии Cloud Foundry компании Pivotal, размещенной на Amazon Web Services. Большинство элементов контроля в системах, размещенных на cloud.gov, а именно — 269 из 325 необходимых для информационной системы умеренного воздействия, приняты на уровне платформы. Системы, размещенные на cloud.gov, могут пройти путь от dev complete до запуска в жизнь в течение нескольких недель, а не месяцев. Это значительно сокращает объем работ и, следовательно, затрат, необходимых для реализации требований системы управления рисками.

Подробнее можно прочитать здесь: https://18f.gsa.gov/2017/02/02/cloud-gov-is-now-fedramp-authorized

Когда встраивание безопасности в программное обеспечение является частью повседневной работы разработчиков, а команды infosec предоставляют инструменты, обучение и поддержку, чтобы облегчить разработчикам выполнение правильных действий, эффективность доставки повышается. Кроме того, это положительно сказывается на безопасности. Мы обнаружили, что участники опроса из группы с высокими показателями эффективности тратят на 50% меньше времени на устранение проблем безопасности, чем участники с низкими показателями. Другими словами, встраивая безопасность в свою повседневную работу вместо решения проблем безопасности в конце разработки, они тратили значительно меньше времени на решение задач безопасности.

Движение за надежность

Для расширения движения DevOps с целью охватить проблемы информационной безопасности были предложены и другие названия. Одно из них — DevSecOps (придуманное такими известными людьми в отрасли, как Топо Пал из Capital One и Шеннон Литц из Intuit). Еще один вариант названия — «Надежный DevOps», придуманный Джошем Корманом и Джеймсом Уикеттом. «Надежный DevOps» — это комбинация из DevOps и «Манифеста Надежности» (Rugged Manifesto):

- Я надежен, и, что более важно, мой код надежен.
- Я признаю, что программное обеспечение стало основой современного мира.
- Я осознаю огромную ответственность, которая приходит с этой фундаментальной ролью.
- Я осознаю, что не знаю, как будет использоваться мой код, и признаю, что он будет использован не так, как предполагалось при его создании, и гораздо дольше, чем было рассчитано.
- Я осознаю, что мой код будет атакован талантливыми и настойчивыми противниками, которые угрожают нашей физической, экономической и национальной безопасности.
- Я осознаю все это и выбираю быть надежным.
- Я надежен, потому что я отказываюсь быть источником уязвимости или слабости.
- Я надежен, потому что я ручаюсь, что мой код будет поддерживать свою миссию.
- Я надежен, потому что мой код может столкнуться с этими проблемами и выдержать их натиск.
- Я надежен не потому, что это легко, а потому, что это необходимо и я готов принять этот вызов (Корман и соавторы, 2012).

Чтобы движение за надежность преуспело и в соответствии с принципами DevOps, быть надежным — это ответственность каждого.

Глава 7

Управленческие практики при работе с ПО

Теория и практика управления в контексте доставки программного обеспечения претерпела значительные изменения за последние десятилетия вместе с несколькими основными парадигмами. В течение многих лет доминировала парадигма управления проектами и программами, и ее можно найти в таких структурах, как Институт управления проектами (Project Management Institute) и PRINCE2. После выпуска Манифеста Agile в 2001 году быстро набрали обороты методы Agile.

Между тем идеи бережливого движения в производстве стали применяться к программному обеспечению. Это движение происходит от подхода Toyota к производству, первоначально предназначенного для решения задачи создания широкого спектра различных автомобилей для относительно небольшого японского рынка. Стремление Toyota к неустанному совершенствованию позволило компании производить автомобили быстрее, дешевле и с более высоким качеством, чем у конкурентов. Такие компании, как Toyota и Honda, глубоко проникли в американскую автопромышленность, которая выжила только благодаря внедрению их идей и методов. Философия бережливого производства была впервые адаптирована для разработки программного обеспечения Мэри и Томом Поппендайками в их серии книг Lean Software Development («Бережливая разработка ПО»).

В этой главе мы обсудим практики управления, почерпнутые из движения бережливого производства, и то, как они влияют на эффективность доставки программного обеспечения.

Практики бережливого управления

В нашем исследовании мы смоделировали бережливое управление и его применение к доставке программного обеспечения с помощью трех компонентов (рис. 7.1 вместе с упрощенным управлением изменениями, рассмотренным далее в этой главе).

1. Ограничение незавершенного производства (НЗП) и использование этих ограничений для улучшения процессов и увеличения производительности.

2. Создание и поддержка визуальных дисплеев, показывающих ключевые показатели качества и производительности, а также текущий статус работы (включая ошибки), доступ к этим дисплеям для инженеров и руководителей и согласование данных показателей с операционными целями.
3. Использование на ежедневной основе данных из инструментов мониторинга инфраструктуры и эффективности приложений для принятия бизнес-решений.

Использование ограничений НЗП и визуальных дисплеев хорошо известно в сообществе Lean. Они применяются для того, чтобы избежать перегрузок команды (что может привести к более длительным срокам выполнения) и не создавать препятствий для рабочего потока. Самое интересное, что ограничения НЗП сами по себе не сильно влияют на эффективность доставки. Мы наблюдаем сильный эффект только тогда, когда они объединены с использованием визуальных дисплеев и имеют цикл обратной связи от инструментов мониторинга производства до команд доставки или бизнеса. Когда команды используют эти инструменты вместе, мы видим гораздо более сильное положительное влияние на эффективность доставки ПО.

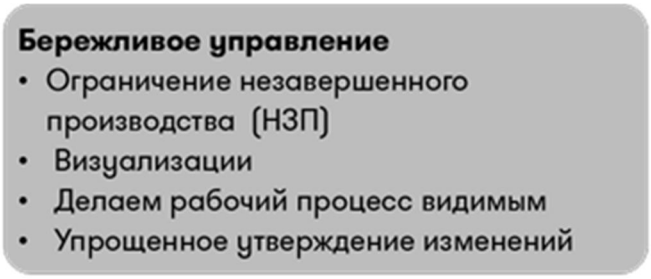


Рис. 7.1. **Компоненты бережливого управления**

Рис. 7.2. **Влияние практик бережливого менеджмента**

Также стоит немного подробнее остановиться на том, что именно мы измеряем. В случае НЗП мы не просто спрашиваем команды, хорошо ли они ограничивают НЗП и имеют ли налаженные процессы для этого. Мы

также спрашиваем, делают ли их ограничения НЗП более заметными препятствия для более высокого рабочего потока и устраняют ли команды эти препятствия путем улучшения процесса, что приводило бы к повышению потока. Ограничения НЗП бесполезны, если они не приводят к улучшениям, увеличивающим рабочий поток.

В случае визуальных дисплеев мы спрашиваем, используются ли они или панели мониторинга для обмена информацией и используют ли команды такие инструменты, как метод канбан или раскадровки, для организации своей работы. Мы также спрашиваем, доступна ли информация о качестве и продуктивности, отображаются ли сбои или ошибки публично с помощью визуальных дисплеев и насколько легко эта информация доступна. Основными принципами при этом являются типы отображаемой информации, как широко она используется и насколько легко к ней получить доступ. Ключевые моменты — наглядность и высококачественная связь, которую она обеспечивает.

Мы предположили, что в сочетании эти методы повышают эффективность доставки, и это действительно так. На самом деле они также оказывают положительное влияние на культуру и производительность команды. Как показано на рис. 7.2, эти практики бережливого управления одновременно уменьшают выгорание (которое мы обсудим в Главе 9) и приводят к более производительной культуре (как описано в модели Веструма в Главе 3).

Внедрение упрощенного процесса управления изменениями

У каждой организации есть какой-то собственный процесс для внесения изменений в свою производственную среду. В стартапах этот процесс управления изменениями может быть настолько простым, как попросить другого разработчика проверить ваш код перед запуском изменения в жизнь. В крупных организациях мы часто видим процессы управления изменениями, которые занимают дни или недели, требуя, чтобы каждое изменение было рассмотрено консультативным советом по изменениям (CAB — change advisory board), внешним по отношению к команде, в дополнение к экспертизам на уровне команды, таким как формальный процесс проверки кода.

Мы хотели изучить влияние процессов утверждения изменений на эффективность доставки программного обеспечения. С этой целью мы задали вопрос о четырех возможных сценариях.

1. Все изменения в производстве должны быть утверждены внешним органом (менеджером или CAB).
2. Только изменения с высоким риском, такие как изменения базы данных, требуют утверждения.

3. Мы полагаемся на внутреннюю экспертную оценку при управлении изменениями.
4. У нас нет процесса утверждения изменений.

Результаты оказались удивительными. Мы обнаружили, что утверждение только изменений с высоким риском не коррелирует с эффективностью доставки программного обеспечения. Команды, которые сообщили об отсутствии процесса утверждения или использовали внутреннюю экспертную оценку, достигали более высокой эффективности доставки ПО. Наконец, команды, которым требовалось утверждение со стороны внешнего органа, продемонстрировали более низкую эффективность.

Далее мы исследовали случай утверждения внешним органом, чтобы посмотреть, коррелирует ли эта практика со стабильностью. Мы обнаружили, что внешние утверждения отрицательно коррелировали со временем выполнения, частотой развертывания и временем восстановления и не имели корреляции с частотой сбоев изменений. Короче говоря, утверждение внешним органом (менеджером или CAB) просто не работает на повышение стабильности производственных систем, измеряемой временем восстановления обслуживания и частотой сбоев изменений. Однако это, безусловно, замедляет работу, что на самом деле хуже, чем вообще не иметь процесса утверждения изменений.

Наша рекомендация, основанная на этих результатах, заключается в использовании упрощенного процесса утверждения изменений, основанного на экспертной оценке, такой как парное программирование или проверка кода внутри команды, в сочетании с конвейером развертывания для обнаружения и отклонения плохих изменений. Этот процесс можно использовать для всех видов изменений, включая изменения кода, инфраструктуры и базы данных.

А как насчет разделения ответственности?

В регулируемых отраслях разделение ответственности часто требуется либо буквально в формулировках регламента (например, в случае PCI DSS), либо аудиторами. Однако реализация этого контроля не требует CAB или отдельной рабочей группы. Существуют два механизма, которые могут быть эффективно использованы для того, чтобы соответствовать букве и духу этого контроля.

Во-первых, когда какое-либо изменение зафиксировано, кто-то, кто не участвовал в создании изменения, должен просмотреть его либо до, либо сразу после фиксации в системе управления версиями. Это может быть кто-то из той же команды. Этот человек должен утвердить изменение, записав свое одобрение в систему записи, такую как GitHub (путем утверждения запроса на принятие изменений) или инструмент конвейера развертывания (утвердив ручной этап сразу после фиксации изменения).

Во-вторых, изменения должны применяться только к производству с использованием полностью автоматизированного процесса, который является частью конвейера развертывания25. То есть никакие изменения не могут быть внесены в производство, если они не были зафиксированы в системе управлении версиями, проверены стандартным процессом сборки и тестирования, а затем развернуты с помощью автоматизированного процесса, запускаемого через конвейер развертывания. В результате применения конвейера развертывания аудиторы будут иметь полную запись о том, какие изменения были применены к каким средам, откуда они берутся в системе управления версиями, какие тесты и проверки были для них выполнены и кто и когда их одобрил. Таким образом, конвейер развертывания особенно ценен в контексте строго регулируемых отраслей или отраслей, для которых безопасность критически важна.

Логически понятно, почему утверждение со стороны внешних органов является проблематичным. Ведь системы программного обеспечения сложны. Каждый разработчик сделал, казалось бы, безобидное изменение, которое подвесило часть системы. Какова вероятность того, что внешний орган, не очень хорошо знакомый с внутренними компонентами системы, может просмотреть десятки тысяч строк изменения кода, внесенные, возможно, сотнями инженеров, и точно определить их влияние на сложную производственную систему? Эта идея является своего рода театром управления рисками: мы устанавливаем флажки, чтобы когда что-то пойдет не так, мы могли сказать, что по крайней мере мы следили за процессом. В лучшем случае этот процесс приведет только к временным задержкам и перебрасыванию ответственности.

Мы считаем, что есть ситуации, когда люди вне команды могут осуществить эффективное управление рисками изменений. Однако это скорее роль руководства, чем собственно проверка изменений. Такие команды должны контролировать эффективность доставки и помогать командам улучшать ее, внедряя практики, которые повышают стабильность, качество и скорость, такие как непрерывная доставка и методы бережливого управления, описанные в этой книге.

Глава 8

Разработка продукта

Бренд Agile более или менее выиграл методологические войны. Однако многое из того, что было реализовано, является, скажем так, искусственным Agile: люди следуют некоторым из общих практик, не затрагивая более широкую организационную культуру и процессы. Например, в крупных компаниях до сих пор нередко на составление бюджета, анализ и сбор требований до начала работы уходят месяцы;

работа сгруппирована в большие проекты с нечастыми релизами, а обработка обратной связи от клиентов рассматривается в десятую очередь. Напротив, и концепция бережливой разработки продукта, и движение бережливого запуска делают акцент на тестировании дизайна и бизнес-модели вашего продукта, проводя частые исследования мнения пользователей с самого начала жизненного цикла продукта.

Книга Эрика Райса The Lean Startup (Райс, 2011) вызвала всплеск интереса к облегченным подходам к изучению новых бизнес-моделей и продуктовых идей в условиях неопределенности. Работа Райса представляет собой синтез идей бережливого движения, дизайн-мышления и работы предпринимателя Стива Бланка (Бланк, 2013), который подчеркивает важность принятия экспериментального подхода к разработке продукта. Этот подход, основанный на наших исследованиях, включает в себя создание и проверку прототипов с самого начала, работу небольшими партиями, а также изменение или «поворот» продуктов и вслед за ними бизнес-моделей на ранней стадии и достаточно часто.

Мы хотели проверить, оказывают ли эти методы непосредственное влияние на эффективность организации, измеряемую с точки зрения производительности, доли рынка и прибыльности.

Методы разработки бережливого продукта

Мы рассмотрели четыре возможности, которые составляют нашу модель бережливого подхода к разработке продукта (см. также рис. 8.1).

1. Степень, с которой команды разделяют продукты и функции на небольшие партии, которые могут быть завершены менее чем за неделю и выпущены частыми релизами, включая использование минимально жизнеспособных продуктов (MVP — minimum viable product).
2. Есть ли у команд хорошее понимание всего потока работы от бизнеса до клиентов и есть ли у них прозрачность в этом потоке, включая статус продуктов и функций.
3. Занимаются ли организации активным и регулярным поиском обратной связи от клиентов и включают ли они эту обратную связь в разработку своих продуктов.
4. Есть ли у группы разработчиков полномочия создавать и изменять спецификации в рамках процесса разработки без необходимости утверждения.

Анализ показал, что эти факторы статистически значимы при прогнозировании более высокой эффективности доставки программного обеспечения и организационной эффективности, а также повышения организационной культуры и снижения выгорания.

Проводя наше исследование в течение нескольких лет, мы также обнаружили, что эффективность доставки программного обеспечения предсказывает методы бережливого управления продуктом. Это взаимное соотношение, представленное в литературе, формирует то, что известно как замкнутый круг благоприятных событий. Повышение эффективности доставки программного обеспечения улучшит вашу способность работать в небольших партиях и включать обратную связь с клиентами на протяжении всего пути.

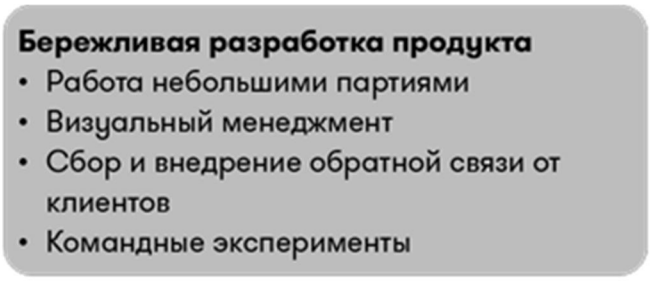


Рис. 8.1. **Компоненты бережливого продукт-менеджмента**

Работа небольшими партиями

Ключ к работе небольшими партиями состоит в том, чтобы работа была разложена на функции, которые поддаются быстрой разработке, вместо сложных функций, разработанных на ветвях и редко выпускаемых. Эта идея может быть применена как на уровне функции, так и на уровне продукта. MVP — это прототип продукта с достаточным количеством функций, обеспечивающий обоснованное изучение продукта и его бизнес-модели. Работа в малых партиях позволяет сократить время выполнения заказа и ускорить цикл обратной связи.

В софтверных компаниях возможность работы и доставки небольшими партиями особенно важна, поскольку она позволяет быстро собирать отзывы пользователей с помощью таких методов, как A/B-тестирование. Стоит отметить, что экспериментальный подход к разработке продукта тесно связан с техническими практиками, которые способствуют непрерывной доставке.

Получение обратной связи от клиентов включает в себя несколько методов: регулярный сбор показателей удовлетворенности клиентов, активный поиск информации о мнении клиентов относительно качества продуктов и функций и использование этой обратной связи для информирования о дизайне продуктов и функций. Важно также и то, в какой степени команды действительно имеют полномочия реагировать на эту обратную связь.

Командные эксперименты

Многие группы разработчиков, претендующих на то, что они внедрили Agile, тем не менее обязаны следовать требованиям, созданным различными командами. Это ограничение может создать реальные

проблемы и привести к созданию продуктов, которые на самом деле не радуют и не привлекают клиентов и не приносят ожидаемых результатов в бизнесе.

Одним из важных пунктов Agile-разработки является сбор мнения клиентов на протяжении всего процесса разработки, в том числе на ранних стадиях. Это позволяет команде разработчиков собирать важную информацию, которая затем информирует о следующих этапах разработки. Но если команде разработчиков не разрешается без официального одобрения какого-либо внешнего органа изменять требования или спецификации в ответ на то, что они обнаруживают, их способность к инновациям резко тормозится.

Наш анализ показал, что способность команд пробовать новые идеи и создавать и обновлять спецификации в процессе разработки, не нуждаясь в одобрении людей вне команды, является важным фактором в прогнозировании эффективности организации, измеряемой с точки зрения прибыльности, производительности и доли рынка.

Мы не предлагаем вам освободить своих разработчиков и позволить им работать над любыми идеями, которые им нравятся. Чтобы быть эффективным, экспериментирование должно сочетаться с другими возможностями, которые мы здесь измеряем: работа небольшими партиями, создание видимого для всех потока работы над всем процессом доставки и включение обратной связи с клиентами в разработку продуктов. Это гарантирует, что ваши команды делают хорошо обоснованный, информированный выбор в отношении проектирования, разработки и доставки работы, а также ее изменения на основе обратной связи. Это также гарантирует, что принимаемые ими обоснованные решения доводятся до сведения всей организации, что увеличивает вероятность того, что идеи и функции, которые они создают, доставят удовольствие клиентам и добавят ценность организации.

Эффективный продукт-менеджмент повышает эффективность

Мы вели анализ возможностей бережливого управления продуктами в течение двух лет, начиная с 2016–2017 годов. В нашей первой модели мы увидели, что методы бережливого управления продуктами положительно влияют на эффективность доставки программного обеспечения, стимулируют производительную культуру и уменьшают выгорание.

В следующем году мы перевернули модель и подтвердили, что эффективность доставки программного обеспечения определяет методы бережливого управления продуктом. Улучшение возможностей доставки ПО позволяет работать небольшими партиями и проводить

исследования пользовательского мнения, что приводит к созданию лучших продуктов. Если мы объединяем модели, это становится обоюдной моделью или, проще говоря, замкнутым кругом благоприятных событий. Мы также обнаружили, что методы бережливого управления продуктами предсказывают эффективность организации, измеряемую с точки зрения производительности, прибыльности и доли рынка. Этот замкнутый круг, состоящий из повышенной эффективности доставки и методов бережливого продукт-менеджмента, обеспечивает достижение лучших результатов для вашей организации (см. рис. 8.2).

Рис. 8.2. **Влияние бережливого продукт-менеджмента**

В софтверных организациях способность работать и доставлять небольшие партии ПО особенно важна, поскольку она позволяет командам интегрировать исследования опыта пользователей в разработку и доставку продукта. Кроме того, способность применять экспериментальный подход к разработке продукта тесно связана с техническими практиками, которые способствуют непрерывной доставке.

Глава 9

Как обеспечить устойчивость работы

Чтобы гарантировать, что эффективность доставки программного обеспечения не достигается с помощью грубой силы или за счет психического здоровья вашей команды, наш проект исследовал как выгорание в командах, так и то, насколько болезненным является процесс развертывания. Мы измерили их, потому что мы знаем, что они являются важными проблемами в технологической отрасли, которые

приводят к болезням, истощению и потере производительности на миллионы долларов.

«Боль развертывания»

Страх и тревога, которые испытывают инженеры и технический персонал, когда они запускают код в производство, могут многое рассказать нам об эффективности доставки программного обеспечения команды. Мы называем это «болью развертывания», и ее важно измерить, потому что она подчеркивает трения и разногласия, которые существуют между разработкой и тестированием программного обеспечения и работой, выполняемой для поддержания и сохранения его работоспособности. Именно здесь разработка встречается с техподдержкой, и именно здесь существует наибольший потенциал для различий: в среде, в процессе и методологии, в мышлении и даже в словах, которые команды используют для описания своей работы.

Наш опыт в этой области и наше взаимодействие на протяжении многих лет с профессионалами, создающими и развертывающими программное обеспечение, постоянно отмечали важность и значимость «боли развертывания». Из-за этого мы хотели исследовать проблемы развертывания, чтобы увидеть, можно ли их измерить и, что более важно, влияют ли на них практики DevOps. Мы обнаружили, что там, где развертывание кода наиболее болезненно, вы найдете самую низкую эффективность доставки программного обеспечения, эффективность организации и культуру.

Преимущества непрерывной доставки в Microsoft

Microsoft engineering — это один из примеров того, как инженерные команды чувствуют преимущества непрерывной доставки. Тьяго Алмейда является старшим инженером по разработке программного обеспечения в Microsoft. Он управляет облачными вычислениями, открытым исходным кодом и DevOps-практиками в команде Azure. Он рассказал о дополнительных преимуществах практики непрерывной доставки для своей команды, сказав следующее: «Вы можете подумать, что все выгоды от этого [получают] ваши клиенты, но на самом деле они [выгоды] есть даже внутри вашей компании…»26 Перед внедрением технических практик и дисциплины непрерывной доставки в команде Bing инженеры сообщили, что показатель удовлетворенности балансом работа/жизнь составляет всего 38%. После внедрения этих технических методов он подскочил до 75%. Разница поразительна. Это означает, что технические специалисты лучше справлялись со своими профессиональными обязанностями в рабочее время, им не приходилось выполнять процессы развертывания вручную и они были в состоянии выдерживать рабочее напряжение.

Хотя «боль развертывания» может свидетельствовать о том, что разработка и доставка программного обеспечения не являются

устойчивыми в вашей организации, также вызывает беспокойство, когда группы разработки и тестирования не имеют представления о том, что такое развертывание. Если у ваших команд нет прозрачности в развертываниях кода, то есть если вы спросите свои команды, что такое развертывание программного обеспечения, и ответ будет: «Я не знаю… Я никогда об этом не думал!» — это еще одно предупреждение о том, что эффективность доставки программного обеспечения может быть низкой, потому что если разработчики или тестировщики не знают о процессе развертывания, вероятно, существуют барьеры, скрывающие от них работу. И барьеры, которые скрывают работу по развертыванию от разработчиков, редко бывают полезными, потому что они изолируют разработчиков от последствий их работы.

Часто разработчики и особенно специалисты техподдержки спрашивают нас: «Что можно сделать, чтобы облегчить "боль развертывания" и улучшить работу технического персонала?» Чтобы ответить на этот вопрос, мы включили параметр «боль развертывания» в наши исследования в 2015, 2016 и 2017 годах. Основываясь на нашем собственном опыте разработки и доставки программного обеспечения, а также разговорах с людьми, работающими с системами, мы создали новый параметр измерения, чтобы уловить, как люди чувствуют себя при развертывании кода. Измерение «боли развертывания» оказалось относительно простым: мы спросили респондентов, были ли развертывания опасны, разрушительны в их работе, или, напротив, были легкими и безболезненными.

Наше исследование показывает, что улучшение ключевых технических возможностей снижает «боль развертывания»: команды, которые реализуют комплексную автоматизацию тестирования и развертывания; используют непрерывную интеграцию, включая магистральную разработку; «сдвигаются влево» по безопасности; эффективно управляют тестовыми данными; используют слабосвязанные архитектуры; могут работать независимо; используют контроль версий всего, что требуется для воспроизведения производственных сред, — уменьшают «боль развертывания».

Другими словами, технические практики, которые улучшают нашу способность доставлять программное обеспечение как быстро, так и стабильно, также уменьшают стресс и беспокойство, связанные с запуском кода в производство. Эти технические практики изложены в Главах 4 и 5. Статистический анализ также выявил высокую корреляцию между «болью развертывания» и ключевыми результатами: чем более болезненными являются развертывания кода, тем хуже эффективность ИТ, организационная эффективность и организационная культура.

Насколько болезненны ваши развертывания кода?

Если вы хотите знать, как работает ваша команда, просто спросите ее, насколько болезненны развертывания и что конкретно вызывает эту боль.

В частности, имейте в виду, что если развертывание должно выполняться в нерабочее время, это признак архитектурных проблем, которые должны быть решены. Вполне возможно — при наличии достаточных инвестиций — построить сложные, крупномасштабные распределенные системы, которые позволяют полностью автоматизировать развертывание с нулевым временем простоя.

В основном большинство проблем развертывания вызвано сложным, хрупким процессом развертывания. Как правило, это является результатом трех факторов. Во-первых, программное обеспечение часто пишется без учета возможности дальнейшего развертывания. Общим симптомом здесь является то, что требуются сложные, организованные развертывания, поскольку программное обеспечение предполагает, что его среда и зависимости будут настроены под него совершенно особым образом, и не допускает никаких отклонений от этих ожиданий. Это дает мало полезной информации администраторам о том, что идет не так и почему ПО работает неправильно. (Эти характеристики также говорят о плохом проектировании для распределенных систем.)

Во-вторых, вероятность неудачного развертывания существенно возрастает, когда в процессе развертывания необходимо вручную внести изменения в производственную среду. Ручные изменения могут легко привести к ошибкам, вызванным вводом текста, ошибками копирования/вставки или плохой или устаревшей документацией. Кроме того, среды, конфигурация которых управляется вручную, часто существенно отличаются друг от друга (проблема, известная как «дрейф конфигурации»), что приводит к значительным объемам работы во время развертывания, когда операторы отлаживают систему, чтобы понять различия в конфигурации, внося вручную дополнительные изменения, которые потенциально могут добавить еще проблем.

Наконец, в рамках сложных развертываний часто требуется несколько итераций передачи полномочий между командами, особенно в разрозненных организациях, где администраторы баз данных, сетевые администраторы, системные администраторы, infosec, тестирование/QA и разработчики работают в отдельных командах.

Чтобы уменьшить проблемы при развертывании, мы должны:

- обеспечить создание систем, которые предназначены для легкого развертывания в нескольких средах, могут обнаруживать и допускать сбои в своих средах и независимо обновлять различные компоненты системы;
- обеспечить возможность автоматического воспроизведения состояния производственных систем (за исключением

производственных данных) из информации в системе управления версиями;
- встроить интеллектуальные решения в приложение и платформу, чтобы процесс развертывания был насколько возможно простым.

Приложения, разработанные для формата «платформа как сервис», такие как Heroku, Pivotal Cloud Foundry, Red Hat OpenShift, Google Cloud Platform, Amazon Web Services или Microsoft Azure, обычно можно развернуть с помощью одной команды27.

Теперь, когда мы обсудили «боль развертывания» и рассмотрели некоторые стратегии противодействия ей, давайте перейдем к выгоранию. «Боль развертывания», если ее оставить без внимания, может привести к выгоранию.

Выгорание

Выгорание — это физическое, умственное или эмоциональное истощение, вызванное переутомлением или стрессом, но это больше, чем просто переутомление или стресс. Выгорание приводит к тому, что вещи, которые мы когда-то любили в нашей работе и жизни, начинают казаться незначительными и скучными. Оно часто проявляется как чувство беспомощности и связано с патологическими культурами и непродуктивной, бесполезной работой.

Последствия выгорания огромны как для отдельных людей, так и для их команд и организаций. Исследования показывают, что стрессовая работа может быть так же вредна для физического здоровья, как пассивное курение (Гох и соавторы, 2015) и ожирение (Чандола и соавторы, 2006). Симптомы выгорания включают в себя чувство усталости, цинизма или неэффективности; мало или совсем нет чувства выполненного долга, а чувства по поводу работы негативно влияют на другие аспекты вашей жизни. В крайних случаях выгорание может привести к семейным проблемам, тяжелой клинической депрессии и даже самоубийству.

Стресс на работе также влияет на работодателей, обходясь экономике США в $300 млрд в год за счет больничных, долгосрочной нетрудоспособности и чрезмерной текучести кадров (Маслах, 2014). Таким образом, работодатели равно обязаны заботиться о своих сотрудниках и обеспечивать условия, в которых персонал не сгорал бы на работе.

Выгорание можно предотвратить или обратить вспять, и DevOps может в этом помочь. Организации могут устранить условия, которые приводят к выгоранию, создавая благоприятную рабочую среду, убеждая людей, что их работа имеет значение, и обеспечивая понимание сотрудниками того, как их собственная работа связана со стратегическими целями компании.

Как и любая другая быстро развивающаяся сфера работы с серьезными последствиями, программное обеспечение и технологии страдают от выгорания сотрудников. Технологические руководители, как и многие другие менеджеры с благими намерениями, часто пытаются исправить человека, игнорируя рабочую среду, хотя изменение окружающей среды гораздо более важно для долгосрочного успеха. Руководители, которые хотят предотвратить выгорание сотрудников, должны сосредоточить свое внимание и усилия на следующих действиях:

- создать уважительную, поддерживающую рабочую среду, в которой на неудачах учатся, а не используют их для обвинений;
- транслировать сильное чувство цели;
- инвестировать в развитие сотрудников;
- спрашивать сотрудников, что мешает им достичь своих целей, а затем исправлять это;
- предоставлять сотрудникам время, пространство и ресурсы для экспериментов и обучения.

И последнее, но не менее важное: сотрудникам должны быть предоставлены полномочия принимать решения, которые влияют на их работу и их рабочие места, особенно в тех областях, где они несут ответственность за результаты.

Общие проблемы, которые могут привести к выгоранию

Кристина Маслах, профессор психологии Калифорнийского университета в Беркли и пионер в области исследований профессионального выгорания, обнаружила шесть организационных факторов риска, которые влекут за собой выгорание (Лейтер и Маслах, 2008)28.

1. Перегрузка работой: требования работы превышают человеческие пределы.
2. Отсутствие контроля: неспособность влиять на решения, которые затрагивают вашу работу.
3. Недостаточное вознаграждение: недостаточное финансовое, институциональное или социальное вознаграждение.
4. Распад сообщества: неблагоприятная рабочая среда.
5. Отсутствие справедливости: отсутствие справедливости в процессе принятия решений.
6. Конфликты ценностей: несоответствие организационных и личных ценностей.

Маслах обнаружила, что большинство организаций пытаются исправить человека и игнорируют рабочую среду, хотя ее исследование показывает, что исправление среды имеет более высокую вероятность

успеха. Все вышеперечисленные факторы риска — это то, что менеджмент и организации могут изменить. Мы также отсылаем читателя к Главе 11 для получения дополнительной информации о важности и влиянии лидерства и управления в DevOps.

Чтобы измерить выгорание, мы спросили респондентов:

- чувствовали ли они себя выгоревшими или истощенными. Многие из нас знают, что такое выгорание, и мы часто опустошены им;
- чувствовали ли они безразличие или цинизм по отношению к своей работе или чувствовали ли они себя неэффективными. Классическим признаком выгорания является безразличие и цинизм, а также ощущение того, что ваша работа больше не эффективна или не приносит пользы;
- оказывала ли их работа негативное влияние на их жизнь. Когда ваша работа начинает негативно влиять на вашу жизнь вне работы, часто наступает выгорание.

Наше исследование обнаружило, что улучшение технических практик (например, тех, которые способствуют непрерывной доставке) и бережливых практик (например, в области бережливого менеджмента и бережливого управления продуктами) уменьшает чувство выгорания среди наших респондентов.

Как уменьшить выгорание или бороться с ним

Наши собственные исследования показывают, какие организационные факторы сильнее всего коррелируют с высоким уровнем выгорания, и подсказывают, где искать решения. Вот пять главных факторов.

1. Организационная культура. Сильное чувство выгорания обнаруживаеся в организациях с патологической, ориентированной на власть культурой. Менеджеры в конечном счете несут ответственность за создание благоприятной и уважительной рабочей среды, и они могут сделать это, создав среду без обвинений, стремясь учиться на неудачах и сообщая общее чувство цели. Менеджеры должны также следить за другими сопутствующими факторами и помнить, что человеческая ошибка никогда не является основной причиной сбоев в системах.
2. «Боль развертывания». Сложные, болезненные развертывания, которые должны выполняться в нерабочее время, способствуют высокому стрессу и ощущению отсутствия контроля29. При правильной практике развертывания не должны быть болезненными событиями. Руководители и лидеры должны спросить свои команды, насколько болезненно происходит их развертывание, и исправить то, что причиняет наибольшую боль.

3. Эффективность лидеров. Обязанности руководителя группы включают ограничение работы в процессе и устранение препятствий, чтобы команда могла выполнять свою работу. Неудивительно, что респонденты с эффективными руководителями команд сообщили о более низком уровне выгорания.
4. Организационные инвестиции в DevOps. Организации, которые инвестируют в развитие навыков и возможностей своих команд, получают лучшие результаты. Инвестиции в обучение и предоставление людям необходимой поддержки и ресурсов (включая время) для приобретения новых навыков имеют решающее значение для успешного внедрения DevOps.
5. Организационная эффективность. Наши данные показывают, что бережливое управление и методы непрерывной доставки помогают повысить эффективность доставки программного обеспечения, что, в свою очередь, повышает организационную эффективность. В основе бережливого менеджмента лежит предоставление сотрудникам необходимого времени и ресурсов для улучшения собственной работы. Это означает создание рабочей среды, которая поддерживает эксперименты, трансформирует ошибки в обучение и позволяет сотрудникам принимать решения, влияющие на их работу. Это также означает создание пространства для сотрудников, позволяющего делать новую, творческую, добавляющую ценность работу в течение рабочей недели, а не просто ожидание того, что они посвятят этому дополнительное время после окончания рабочего дня. Хорошим примером может служить политика Google «20% времени», в рамках которой компания позволяет сотрудникам 20% своей рабочей недели работать над новыми проектами. Или, например, программа IBM «THINK Friday», когда по пятницам во второй половине дня нет никаких встреч, а сотрудники поощряются к работе над новыми интересными им проектами, на которые у них обычно нет времени.

Следует отметить важность выравнивания ценностей и роль этого фактора в борьбе с выгоранием. Когда организационные и индивидуальные ценности не совпадают, вы с большей вероятностью увидите выгорание у сотрудников, особенно в такой требовательной и рискованной работе, как ИТ. Мы видели это слишком часто, и последствия этого явления широко распространены и печальны.

Мы считаем, что противоположное состояние является более перспективным и действенным, то есть когда организационные и индивидуальные ценности совпадают, последствия выгорания могут быть сокращены и даже устранены. Например, если человек ценит защиту окружающей среды, а организация сбрасывает отходы в близлежащие реки и тратит деньги на лоббирование своих представителей в правительстве, чтобы позволить этому продолжаться, налицо противоречие ценностей. Этот человек, вероятно, будет гораздо

счастливее, работая в организации с сильной приверженностью корпоративной социальной ответственности в рамках «зеленых» инициатив. Это область потенциального влияния на сотрудников, которой организации пренебрегают на свой страх и риск. Выравнивая организационные ценности с индивидуальными, можно уменьшить выгорание сотрудников. Представьте себе, как это влияет на удовлетворенность, производительность и лояльность сотрудников. Потенциальная ценность для организаций и экономики ошеломляет.

Важно отметить, что организационные ценности, которые мы здесь рассматриваем, являются реальными, актуальными, живыми организационными ценностями, ощущаемыми сотрудниками. Если организационные ценности, которые ощущают сотрудники, отличаются от официальных ценностей организации — заявлений о миссии, напечатанных на листках бумаги или даже на плакатах, — это будут именно те повседневные жизненные ценности, которые имеют значение. Если есть несоответствие ценностей либо между сотрудником и организацией, либо между заявленными ценностями организации и ее фактическими ценностями, выгорание будет проблемой. Когда есть выравнивание, сотрудники будут процветать.

Таким образом, наше исследование показало, что технические и бережливые методы управления способствовали снижению как выгорания, так и «боли развертывания». Обобщенно это можно увидеть на рис. 9.1. Эти выводы имеют серьезные последствия для технологических организаций: инвестиции в технологии не только улучшают разработку и доставку программного обеспечения, но и улучшают рабочую жизнь наших специалистов.

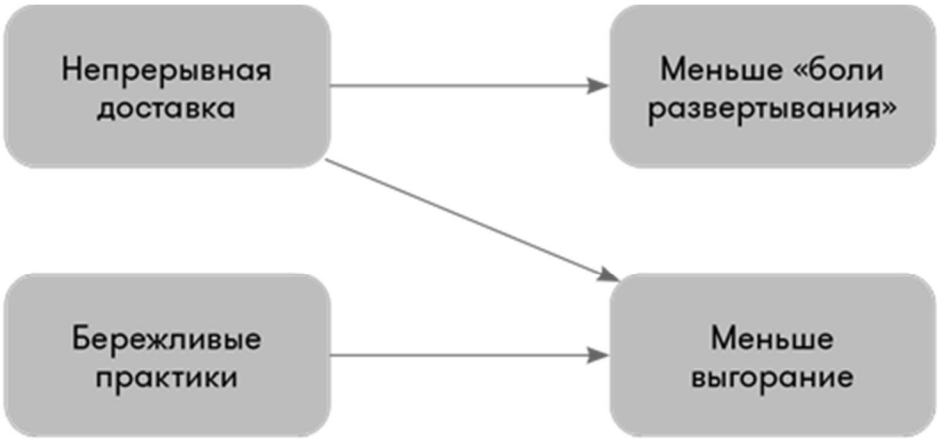


Рис. 9.1. **Влияние технических и бережливых практик на рабочую жизнь**

Мы обсудили важные компоненты организационной культуры и пути ее улучшения и оценки. Теперь мы обратимся к деталям удовлетворенности и идентификации сотрудников с компанией и к тому, что это означает для технологических преобразований.

Глава 10

Удовлетворенность сотрудников, их идентификация с организацией и вовлеченность

Люди находятся в центре каждой технологической трансформации. Под давлением рынка, вынуждающего как можно быстрее поставлять новые технологии и решения, важность привлечения, удержания и вовлечения персонала больше, чем когда бы то ни было. Каждый хороший менеджер знает это, но все еще нет информации о том, как измерить эти результаты, и о том, что на них влияет, особенно в контексте технологических преобразований.

Мы хотели включить в наше исследование людей, которых затронуло внедрение практик DevOps, чтобы посмотреть, что может улучшить их работу и могут ли эти улучшения повлиять на организацию. Наше исследование показало, что вовлеченность и удовлетворенность сотрудников являются показателями лояльности и приверженности сотрудников компании. Они способны снизить выгорание и повысить ключевые организационные результаты, такие как прибыльность, продуктивность и доля рынка. Мы также покажем вам, как измерить эти главные человеческие факторы, чтобы вы могли реализовать их в своих собственных командах — будь вы лидером, руководителем или заинтересованным специалистом.

В этой главе мы обсудим лояльность сотрудников (измеряемую через eNPS и идентификацию сотрудников с компанией) и удовлетворенность работой, а затем завершим обсуждение темой разнообразия.

Лояльность сотрудников

Чтобы понять вовлеченность сотрудников в контексте технологических преобразований и DevOps, мы рассмотрели ее через призму широко используемого стандарта лояльности клиентов — Net Promoter Score (NPS).

В высокоэффективных компаниях лояльность сотрудников, измеряемая индексом чистой лояльности (eNPS), выше. Наши исследования показали, что сотрудники высокоэффективных организаций в 2,2 раза чаще рекомендуют свою компанию как отличное место для работы, а другие исследования также показали, что это коррелирует с лучшими результатами бизнеса (Аззарелло и соавторы, 2012).

Измерение индекса лояльности (NPS)

NPS рассчитывается на основе одного простого вопроса: насколько вероятно, что вы порекомендуете нашу компанию/продукт/услугу другу или коллеге?

NPS оценивается по шкале от 0 до 10 и классифицируется так.

- Клиенты, которые дают оценку 9 или 10, считаются «промоутерами». Промоутеры создают наибольшую ценность для компании, потому что они, как правило, покупают больше, остаются дольше, генерируют позитивное «сарафанное радио», а приобрести и удержать их стоит меньше.
- Те, кто дает оценку 7 или 8, считаются «пассивами». Пассивы — это удовлетворенные, но гораздо менее восторженные клиенты. Они менее склонны предоставлять рекомендации и с большей вероятностью переметнутся к конкурентам, если появится что-то лучшее.
- Те, кто дает оценку от 0 до 6, считаются «недоброжелателями». Недоброжелателей дороже приобретать и удерживать, они быстрее «дезертируют» и могут повредить бизнесу через негативное «сарафанное радио».

В нашем исследовании мы задали два вопроса, чтобы оценить eNPS.

1. Вы бы порекомендовали свою ОРГАНИЗАЦИЮ в качестве места работы другу или коллеге?
2. Вы бы порекомендовали свою КОМАНДУ в качестве места работы другу или коллеге?

Мы сравнили долю промоутеров (тех, кто дал оценку 9 или 10) в высокоэффективной группе участников с низкоэффективной группой. Мы обнаружили, что сотрудники в высокоэффективных командах в 2,2 раза чаще порекомендовали бы свою организацию другу как отличное место работы и в 1,8 раза чаще порекомендовали бы другу свою команду.

Это важный вывод, так как исследования показали, что «компании с высокой вовлеченностью работников увеличили доходы в 2,5 раза по сравнению с компаниями с низким уровнем вовлеченности. И [публично торгуемые] акции компаний с рабочей средой с высоким уровнем доверия увеличили свои рыночные индексы в три раза с 1997 по 2011 год» (Аззарелло и соавторы, 2012).

Вовлеченность сотрудников — это не просто показатель хорошего самочувствия. Она определяет результаты бизнеса. Мы обнаружили, что индекс чистой лояльности сотрудников коррелирует со следующими конструкциями:

- степень, в которой организация собирает отзывы клиентов и использует их для информирования команд разработки продуктов и функций;

- способность команд визуализировать и понимать поток продуктов или функций через разработку и далее весь путь до клиента;
- степень, в которой сотрудники идентифицируют себя с ценностями и целями своей организации, и усилия, которые они готовы приложить, чтобы сделать организацию успешной.

Как мы показали в Главе 8, когда сотрудники видят связь между выполняемой ими работой и ее позитивным воздействием на клиентов, они сильнее идентифицируют себя с целями компании, что приводит к улучшению доставки программного обеспечения и организационной эффективности.

Что такое NPS

Хотя это может показаться упрощенной оценкой, исследования показали, что NPS коррелирует с ростом компаний во многих отраслях (Райхельд, 2003). Аналогично оценке NPS самой компании оценка eNPS используется для измерения лояльности сотрудников.

Существует связь между лояльностью сотрудников и их работой: лояльные сотрудники являются наиболее вовлеченными и полностью выкладываются, часто делая дополнительную работу, чтобы обеспечить лучший опыт для клиента, что, в свою очередь, повышает эффективность компании.

NPS рассчитывается путем вычитания процента недоброжелателей из процента промоутеров. Например, если 40% сотрудников являются недоброжелателями и только 20% — промоутерами, то NPS составляет 20%.

Изменение организационной культуры и идентификация

Люди — это самый главный актив любой организации, но, к сожалению, часто к ним относятся как к расходным материалам. Когда лидеры инвестируют в своих сотрудников и позволяют им делать свою лучшую работу, сотрудники сильнее идентифицируют себя с организацией и готовы приложить дополнительные усилия, чтобы помочь ей добиться успеха. В свою очередь, организации получают более высокий уровень эффективности и производительности, что приводит к лучшим результатам для бизнеса. Эти данные исследования показаны на рис. 10.1.

Рис. 10.1. **Влияние технических и бережливых практик на идентификацию**

Эффективные методы управления в сочетании с техническими подходами, такими как непрерывная доставка, не только влияют на эффективность, но и оказывают измеримое влияние на организационную культуру. Продолжая наше исследование, мы добавили новый показатель: уровень идентификации респондентов с организациями, в которых они работают. Чтобы его измерить, мы спросили людей, в какой степени они согласны со следующими утверждениями (адаптировано из работы Канканхалли и соавторов, 2005).

- Я рад, что выбрал работу в этой организации, а не в другой компании.
- Я рассказываю об этой организации друзьям как об отличном месте работы.
- Я готов приложить гораздо больше усилий, чем обычно ожидается, чтобы помочь моей организации добиться успеха.
- Я считаю, что мои ценности и ценности организации очень схожи.
- В целом люди, занятые в моей организации, работают в направлении одной цели.
- Я чувствую, что моя организация заботится обо мне.

Мы использовали шкалу Ликерта для измерения согласия или несогласия с этими утверждениями. Пункты удовлетворяли всем статистическим условиям для измерения конструкции (в данном случае идентификации); поэтому для измерения идентификации в ваших собственных командах вы можете усреднить оценки пяти пунктов, взятые вместе, в одну оценку для идентификации человека. (См. Главу 13 для обсуждения психометрии и скрытых конструкций.)

Наша ключевая гипотеза при постановке этих вопросов состояла в том, что команды, реализующие практики непрерывной доставки и применяющие экспериментальный подход к разработке продукта, будут создавать лучшие продукты, а также будут чувствовать себя более связанными с остальной частью своей организации. Это, в свою очередь, создает позитивный замкнутый круг: создавая более высокие уровни эффективности доставки программного обеспечения, мы увеличиваем

скорость, с которой команды могут проверять свои идеи, тем самым обеспечивая более высокие уровни удовлетворенности работой и организационной эффективности.

Другим ключевым моментом является то, что идентификация включает в себя согласование ценностей с целями команды и организации. Вспомните из предыдущей главы, что одним из основных факторов выгорания является несоответствие личных и организационных ценностей. Это говорит нам о том, что чувство собственной идентификации с компанией может помочь уменьшить выгорание путем согласования личных и организационных ценностей. Поэтому инвестиции в практики непрерывной доставки и бережливого управления, которые способствуют укреплению чувства принадлежности, могут отлично помочь уменьшить выгорание. Еще раз, это создает позитивный замкнутый круг создания ценности в бизнесе, где инвестиции в технологии и процессы, которые делают работу лучше для сотрудников, необходимы для обеспечения ценности для клиентов и бизнеса.

Это контрастирует с тем, как многие компании все еще работают: требования спускаются командам разработчиков, которые затем должны доставлять большие части работы в пакетном режиме. В этой модели сотрудники ощущают слабый контроль над продуктами и опытом клиентов, которые они создают, а также слабую связь с организациями, в которых они работают. Это чрезвычайно демотивирует команды и приводит к тому, что сотрудники чувствуют себя эмоционально оторванными от своей работы, и, соответственно, к худшим организационным результатам.

Степень, с которой люди отождествляли себя со своей организацией, предопределяла производительную, ориентированную на результат культуру, а также эффективность организации, измеряемую с точки зрения производительности, доли рынка и прибыльности. Это не должно нас удивлять. Если люди являются самым главным активом компании (а многие корпоративные лидеры заявляют, что это именно так), то наличие сотрудников, которые сильно идентифицируют себя с компанией, должно стать конкурентным преимуществом. Однажды лидер одной из компаний из списка Fortune 500 спросил Эйдриана Кокрофта, главного архитектора облака Netflix, откуда он берет своих удивительных людей. Кокрофт ответил: «Я нанял их у вас!» (личное общение). Наш вывод ясен: в сегодняшнем быстро развивающемся и конкурентном мире лучшее, что вы можете сделать для своих продуктов, своей компании и своих сотрудников, — это создать культуру экспериментов и обучения и инвестировать в технические и управленческие возможности, которые этому способствуют. Как было показано в Главе 3, здоровая организационная культура способствует привлечению и удержанию сотрудников, и лучшие, самые инновационные компании извлекают из этого выгоду.

Как удовлетворенность работой влияет на организационную эффективность?

Мы уже упоминали о позитивном замкнутом круге ранее в отношении эффективности доставки программного обеспечения, и здесь мы опять видим его в действии: люди, которые чувствуют поддержку своих работодателей, у которых есть инструменты и ресурсы для выполнения своей работы и которые чувствуют, что их мнение ценится, работают лучше всего. Лучшая работа приводит к более высокой эффективности доставки программного обеспечения, что, в свою очередь, повышает уровень организационной эффективности. Эти результаты показаны на рис. 10.2.

Рис. 10.2. **Влияние технических и бережливых практик на удовлетворенность работой**

Этот замкнутый круг непрерывного совершенствования и обучения является отличительной чертой успешных компаний, позволяющей им внедрять инновации, опережать конкурентов — и побеждать.

Как DevOps способствует удовлетворенности работой?

Хотя DevOps — это, прежде всего, культура, важно отметить, что удовлетворенность работой сильно зависит от наличия правильных инструментов и ресурсов для ее выполнения. На самом деле наша мера удовлетворенности работой определяется несколькими ключевыми условиями: если вы удовлетворены своей работой, если вам предоставлены инструменты и ресурсы для выполнения своей работы и если в вашей работе оптимально задействованы ваши навыки и способности. Важно перечислить эти моменты, потому что вместе взятые они делают удовлетворение от работы поистине впечатляющим.

Инструменты являются важным компонентом практики DevOps, и многие из этих инструментов подразумевают автоматизацию. Кроме того, мы обнаружили, что хорошая техническая практика DevOps предопределяет

удовлетворенность работой. Автоматизация имеет значение, потому что она дает на откуп компьютерам то, в чем они хороши, — механические задачи, которые не требуют мышления и которые на самом деле выполняются лучше, когда вы не слишком много думаете о них. Поскольку человек плохо справляется с такого рода задачами, передача их компьютерам позволяет людям сосредоточиться на вещах, в которых они хороши: взвешивании доказательств, обдумывании проблем и принятии решений. Способность применять свои суждения и опыт к сложным проблемам — это львиная доля того, что делает людей удовлетворенными своей работой.

Рассматривая показатели, которые сильно коррелируют с удовлетворенностью работой, мы видим некоторые общие черты. Такие методы, как проактивный мониторинг и автоматизация тестирования и развертывания, автоматизируют служебные задачи и требуют от людей принятия решений на основе цикла обратной связи. Вместо того чтобы управлять задачами, люди получают возможность принимать решения, используя свои навыки, опыт и мнение.

Разнообразие в ИТ: что выявило наше исследование

Разнообразие имеет значение. Исследования показывают, что команды с большим разнообразием в отношении пола сотрудников или различных меньшинств умнее (Рок и Грант, 2016), добиваются лучшей командной эффективности (Deloitte, 2013) и достигают лучших результатов в бизнесе (Хант и соавторы, 2015). Наши исследования показывают, что совсем немного команд могут похвастаться таким разнообразием. Мы рекомендуем командам, стремящимся достичь высоких результатов, сделать все возможное для привлечения и удержания большего числа женщин и представителей различных меньшинств, а также работать над улучшением разнообразия и в других сферах, таких как люди с ограниченными возможностями.

Важно также отметить, что разнообразия недостаточно. Команды и организации также должны быть инклюзивными. Инклюзивная организация — это та, в которой «все члены организации чувствуют себя желанными и ценными за то, кто они и что "приносят к столу". Все заинтересованные стороны разделяют сильное чувство принадлежности и реализуют общую цель» (Смит и Линдсей, 2014, с. 1). Инклюзивность необходима, чтобы разнообразие пустило корни.

Женщины в DevOps

Мы начали поднимать гендерный вопрос в 2015 году, что вызвало оживленную дискуссию в социальных сетях на тему женщин в технологиях. Мы услышали все: от искренней поддержки многих женщин

и мужчин в сообществе DevOps до вопросов о том, зачем нужно гендерное разнообразие в ИТ. Из общего числа респондентов 5% идентифицировали себя как женщин в 2015 году, 6% в 2016 году и 6,5% в 2017 году. Эти цифры оказались намного ниже, чем мы ожидали, учитывая, что женщины составили около 7% в 2011 году (SAGE, 2012), снизившись с 13% в 2008 году (SAGE, 2008) в области системного администрирования и 27% в области управления компьютерами и информацией (Диаз и Кинг, 2013). Мы надеялись найти более обнадеживающее число женщин, работающих в технических командах.

Среди респондентов:

- 33% сообщили, что работают в командах, в которых нет женщин;
- 56% сообщили, что работают в командах, в которых менее 10% женщин;
- 81% сообщили, что работают в командах, в которых менее 25% женщин.

Мы начали наше исследование вокруг бинарной гендерной системы, потому что это позволило нам сравнить наши результаты с существующими исследованиями. В будущем мы надеемся расширить нашу работу в области небинарной гендерной системы. Мы можем представить основную гендерную статистку для исследования 2017 года (см. рис. 10.3).

Рис. 10.3. **Гендерная демография в исследовании 2017 году**

Недостаточно представленные меньшинства в DevOps

Мы также спросили респондентов, относят ли они себя к недостаточно представленным меньшинствам (см. рис. 10.4).

- 77% ответили: «Нет, я не отношу себя к недостаточно представленной группе».
- 12% ответили: «Да, я отношу себя к недостаточно представленной группе».
- 11% ответили, что они предпочли бы не отвечать или не определились.

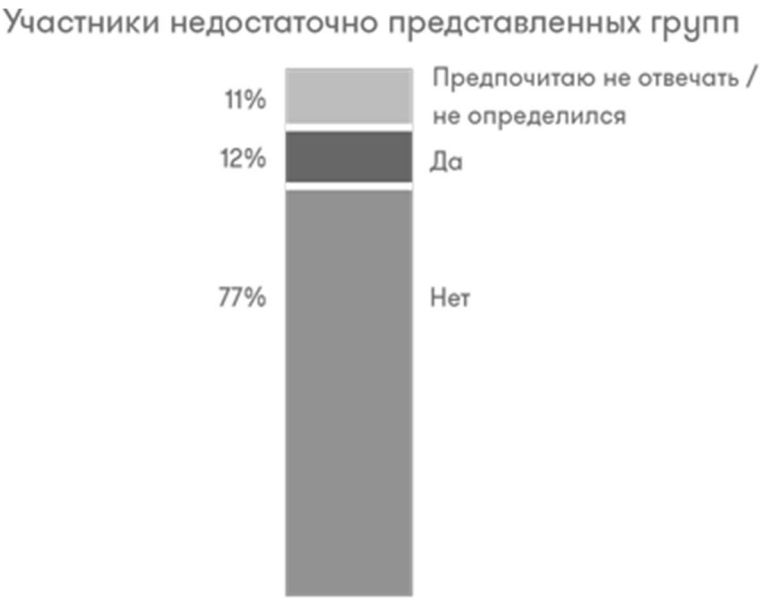


Рис. 10.4. **Демография недостаточно представленных меньшинств в исследовании 2017 году**

Поскольку данные собирались по всему миру, эта самоидентификация была достаточно специфичной. Например, США выделяют и определяют несколько этнических групп и национальностей как меньшинства (афроамериканцы, латиноамериканцы, выходцы с тихоокеанских островов и т.д.), которые не существуют или не имеют смысла в других странах.

Мы еще не расширили наши исследования на людей с ограниченными возможностями, но надеемся сделать это в будущем.

Что другие исследования говорят нам о разнообразии

Большинство исследований в области разнообразия рассматривают бинарную гендерную систему, поэтому давайте начнем с этого. О чем говорят современные исследования? Существует множество исследований, связывающих присутствие женщин на руководящих должностях с более высокими финансовыми показателями (Макгрегор, 2014), показателями фондового рынка (Covert July, 2014) и доходностью хедж-фондов (Covert January, 2014). Кроме того, в своем исследовании Анита Вулли и Томас Мэлоун измерили групповой интеллект и обнаружили, что команды с большим количеством женщин, как правило, попадают в категорию выше среднего по шкале коллективного интеллекта (Вулли и Мэлоун, 2011). Несмотря на эти очевидные

преимущества у организаций не получается нанимать и удерживать женщин в технических областях.

Поскольку нет никаких существенных различий между мужчинами и женщинами с точки зрения способностей или предрасположенности в STEM (наука, технологии, инженерное дело и математика30, Лесли и соавторы, 2015), что же удерживает женщин и представителей различных меньшинств от сферы технологий?31 Ответом может быть не более чем распространенное убеждение, что некоторые люди от природы более подходят для технической работы, потому что обладают врожденным блестящим интеллектом (Лесли и соавторы, 2015).

Именно это распространенное убеждение проникает в нашу культуру, создавая среду, в которой женщинам все труднее оставаться (Снайдер, 2014). Женщины уходят из технологий на 45% чаще, чем мужчины (Quora, 2017), и прогноз для меньшинств, вероятно, аналогичен. Женщины и представители меньшинств сообщают о притеснениях, микроагрессии и неравной оплате труда (например, Mundy, 2017). Это все то, за чем мы можем активно следить и улучшать как лидеры и коллеги.

Что мы можем сделать

Все мы должны уделять приоритетное внимание разнообразию и поощрять инклюзивную среду. Это хорошо и для команды, и для бизнеса. Вот некоторые ресурсы, которые помогут вам начать работу.

- Институт Аниты Борг располагает отличными инструментами для продвижения женщин в области технологий, включая конференцию Грейс Хоппер (Grace Hopper Conference). Не без проблем, но это мероприятие дало многим женщинам воодушевляющий опыт участия в технической конференции, которая привлекла 18 000 женщин только в 2017 году32.
- Geek Feminism — отличный онлайн-ресурс для поддержки женщин в сообществах гиков33.
- Project Include — фантастический ресурс для поддержки разнообразия по нескольким направлениям, полностью онлайн и с открытым исходным кодом34.

Глава 11

Лидеры и руководители

На протяжении многих лет наше исследование изучало влияние различных технических и бережливых методов управления на эффективность доставки программного обеспечения, а также культуру

команды. Однако в первые годы проекта мы напрямую не изучали влияние лидерства на практику DevOps.

В этой главе будут представлены наши выводы о роли лидеров и руководителей в технологических преобразованиях, а также изложены некоторые шаги, которые лидеры могут предпринять для улучшения культуры в своих собственных командах.

Трансформационное лидерство

Не знаете, насколько важно технологическое лидерство? Подумайте вот об этом: к 2020 году половина ИТ-директоров, которые не трансформировали возможности своих команд, будут вытеснены из лидеров цифровых команд своих организаций (исследование компании Gartner).

Это потому, что лидерство действительно оказывает мощное влияние на результаты. Быть лидером не означает, что у вас есть люди, отчитывающиеся перед вами на организационных диаграммах. Лидерство заключается в том, чтобы вдохновлять и мотивировать тех, кто вас окружает. Хороший лидер влияет на способность команды доставлять код, создавать хорошие системы и применять принципы бережливого производства к тому, как команда управляет своей работой и разрабатывает продукты. Все это оказывает ощутимое влияние на прибыльность, производительность и долю рынка организации. Это также оказывает влияние на удовлетворенность клиентов, эффективность и способность достигать организационных целей — некоммерческих целей, которые важны как для коммерческих, так и для некоммерческих организаций. Тем не менее все эти воздействия на организационные и некоммерческие цели являются косвенными, реализуемыми через технические и бережливые практики, которые лидеры поддерживают в своих командах.

На наш взгляд, роль лидерства в технологическом преобразовании была одной из наиболее упущенных тем в DevOps несмотря на то, что трансформационное лидерство имеет большое значение для:

- установления и поддержки созидательных и высоконадежных культурных норм;
- создания технологий, обеспечивающих производительность разработчиков, сокращение времени развертывания кода и поддержку более надежных инфраструктур;
- поддержания экспериментов и инноваций в команде, а также более быстрого создания и внедрения улучшенных продуктов;
- работы над преодолением организационной разобщенности для достижения стратегического выравнивания ценностей.

К сожалению, в сообществе DevOps мы несколько раз были виновны в клевете на руководство — например, когда менеджеры среднего звена или консервативные несогласные мешают командам вносить изменения, необходимые для улучшения доставки программного обеспечения и организационной эффективности.

И все же один из самых распространенных вопросов, который мы слышим: «Как нам заполучить лидеров в команду, чтобы мы могли произвести необходимые изменения?» Мы все признаем, что активное лидерство жизненно необходимо для успешных преобразований DevOps. Лидеры обладают полномочиями и бюджетом для проведения крупномасштабных изменений, которые часто необходимы для обеспечения «прикрытия с воздуха» в ходе преобразований и для изменения стимулов целых групп технических специалистов — будь то в области разработки, контроля качества, техподдержки или информационной безопасности. Лидеры — это те, кто задает тон организации и укрепляет желаемые культурные нормы.

Чтобы изучить трансформационное лидерство, мы использовали модель, которая включает в себя пять измерений (Рафферти и Гриффин, 2004). Согласно этой модели, выделяется пять характеристик трансформационного лидера.

1. Видение. Имеет четкое представление о том, куда идет организация и где она должна быть через пять лет.
2. Вдохновляющее общение. Общается таким образом, что вдохновляет и мотивирует даже в неопределенной или меняющейся среде.
3. Интеллектуальная стимуляция. Заставляет своих последователей думать о проблемах по-новому.
4. Поддерживающее лидерство. Демонстрирует заботу и внимание к личным потребностям и чувствам своих последователей.
5. Личное признание. Хвалит и признает достижение целей и улучшение качества работы; лично хвалит других, когда они делают выдающуюся работу.

Что такое трансформационное лидерство?

Трансформационное лидерство означает, что лидеры вдохновляют и мотивируют своих последователей на достижение более высоких результатов, апеллируя к их ценностям и чувству цели, способствуя тем самым широкомасштабным организационным изменениям. Такие лидеры побуждают свои команды работать над достижением общей цели через свое видение, ценности, общение, собственный пример и очевидную заботу о личных потребностях своих последователей.

Было замечено, что есть сходство между «служебным» лидерством и трансформационным лидерством, но они отличаются фокусом внимания лидера. «Служебные» лидеры сосредотачиваются на развитии и

эффективности своих последователей, в то время как трансформационные лидеры сфокусированы на том, чтобы добиться от последователей идентификации себя с организацией и участия в поддержке организационных целей.

Мы также выбрали трансформационное лидерство в качестве модели для нашего исследования, потому что оно более прогнозируемо с точки зрения результатов работы в других контекстах, а мы были заинтересованы в понимании того, как улучшить эффективность в области технологий.

Мы измерили трансформационное лидерство, используя вопросы опросника, адаптированные из работы Рафферти и Гриффина (2004)35.

Мой лидер или руководитель:

- видение:
- имеет четкое понимание, куда мы идем;
- имеет четкое представление о том, где он/она хочет, чтобы наша команда была через пять лет;
- имеет четкое представление о том, куда идет организация;
- вдохновляющее общение:
- говорит вещи, которые заставляют сотрудников гордиться тем, что они являются частью этой организации;
- позитивно отзывается о работе подразделения;
- поощряет людей видеть изменяющуюся среду как ситуацию, полную возможностей;
- интеллектуальная стимуляция:
- заставляет меня думать о старых проблемах по-новому;
- транслирует идеи, которые заставили меня переосмыслить некоторые вещи, в которых я никогда ранее не сомневался;
- заставляет меня переосмыслить некоторые из моих основных предположений о моей работе;
- поддерживающее лидерство:
- учитывает мои личные чувства, прежде чем действовать;
- ведет себя таким образом, чтобы заботиться о моих личных потребностях;
- видит, что интересы сотрудников учитываются должным образом;
- личное признание:
- поощряет, когда я делаю работу лучше, чем на среднем уровне;
- признает улучшение качества моей работы;
- хвалит лично меня, когда я делаю выдающуюся работу.

Наш анализ показал, что эти характеристики трансформационного лидерства сильно коррелируют с эффективностью доставки программного обеспечения. Фактически, мы наблюдали значительные различия в характеристиках лидерства среди высоко-, средне- и

низкоэффективных команд. Высокоэффективные команды сообщили, что у них есть лидеры с самыми высокими показателями во всех измерениях: видение, вдохновляющая коммуникация, интеллектуальная стимуляция, поддерживающее лидерство и личное признание. Напротив, низкоэффективные команды сообщили о самых низких уровнях этих лидерских характеристик. Все эти различия были статистически значимыми. Сделав еще один шаг вперед в нашем анализе, мы обнаружим, что команды с наименее трансформирующими лидерами с гораздо меньшей вероятностью достигают высоких результатов. В частности, среди команд, которые сообщают о лидерстве в нижней трети шкалы лидерства, только половина, скорее всего, будут высокоэффективными. Это подтверждает наш общий опыт: хотя мы часто слышим истории об успехах DevOps и технологических преобразований, исходящих от рядовых сотрудников, гораздо легче добиться успеха, когда у вас есть поддержка руководства.

Мы также обнаружили, что трансформационное лидерство в значительной степени коррелирует с eNPS. Мы видим трансформационных лидеров там, где сотрудники счастливы, лояльны и вовлечены. Хотя наше исследование не включало показатели трансформационного лидерства и организационной культуры в том же году, другие исследования показали, что сильные трансформационные лидеры создают и поддерживают здоровую командную и организационную культуру (Рафферти и Гриффин, 2004).

Влияние трансформационного лидера проявляется через его поддержку работы своих команд, будь то в технических практиках или возможностях управления продуктами. Положительное (или отрицательное) влияние лидерства распространяется на эффективность доставки программного обеспечения и организационную эффективность. Покажем это на рис. 11.1.

Другими словами, мы нашли доказательства того, что лидеры в одиночку не могут достичь высоких результатов DevOps. Мы рассмотрели результаты работы команд с самыми сильными трансформационными лидерами — теми, которые получили 10% лучших характеристик трансформационного лидерства. Можно было бы предположить, что эффективность этих команд выше средней. Однако они с той же или даже меньшей вероятностью достигали высоких результатов по сравнению со всей совокупностью команд, представленных в результатах опроса.

Это имеет смысл, потому что лидеры не могут достичь целей в одиночку. Они нуждаются в том, чтобы их команды выполняли работу на подходящей архитектуре, с использованием хороших технических практик и принципов бережливого производства, а также всеми другими факторами, которые мы изучали на протяжении многих лет.

Рис. 11.1. **Влияние трансформационного лидерства на технические возможности и возможности бережливого производства**

Подводя итог, мы обнаружили, что лидерство помогает создавать отличные команды, отличные технологии и отличные организации, но косвенно лидерство позволяет командам перестраивать свои системы и внедрять необходимые методы непрерывной доставки и бережливого управления.

Трансформационное лидерство задействует практики, которые коррелируют с высокой эффективностью, а также поддерживает эффективное общение и сотрудничество между членами команды в достижении организационных целей. Такое лидерство также обеспечивает основу для культуры, в которой непрерывное экспериментирование и обучение являются частью повседневной работы каждого сотрудника.

Таким образом, поведение трансформационных лидеров усиливает и активизирует ценности, процессы и практики, выявленные в ходе нашего исследования. Это не отдельная модель поведения или новый набор практик — это лишь то, что усиливает эффективность технических и организационных практик, которые мы изучали в течение нескольких лет.

Роль руководителей

Мы видим, что лидеры играют решающую роль в любом технологическом преобразовании. Когда эти лидеры являются руководителями, они могут еще сильнее влиять на изменения.

Руководители — это те, кто несет ответственность за людей, а часто и за бюджеты и ресурсы в организациях. В лучшем случае руководители также являются лидерами и принимают на себя характеристики трансформационного лидерства, описанные выше.

Руководители, в частности, играют важную роль в соединении стратегических целей бизнеса с работой своих команд. Они могут многое сделать для повышения эффективности своей команды, создавая рабочую среду, в которой сотрудники чувствуют себя в безопасности,

инвестируя в развитие возможностей своих сотрудников и устраняя препятствия в их работе.

Мы также обнаружили, что инвестиции в DevOps сильно коррелируют с эффективностью доставки программного обеспечения. Когда дело доходит до культуры, руководители могут улучшить ситуацию, включив конкретные методы DevOps в свои команды и заметно инвестируя в DevOps и в профессиональное развитие своих сотрудников.

Руководители также могут способствовать значительному улучшению эффективности доставки программного обеспечения, принимая меры, чтобы сделать развертывание менее болезненным. И последнее, но не менее важное: руководители должны сделать показатели эффективности прозрачными и приложить усилия, чтобы привести их в соответствие с целями организации, а также делегировать больше полномочий своим сотрудникам. Знание — это сила, и вы должны дать силу тем, кто обладает знанием.

Вы можете спросить себя: как могут выглядеть инвестиции в инициативы DevOps и мои команды? Есть несколько способов, с помощью которых технологические лидеры могут инвестировать в свои команды.

- Обеспечьте доступность существующих ресурсов для всех сотрудников организации. Создайте пространство и возможности для обучения и совершенствования.
- Создайте специальный бюджет на обучение и убедитесь, что люди знают о нем. Кроме того, дайте вашим сотрудникам свободу выбора обучения, которое их интересует. Этот бюджет на обучение может включать в себя выделенное в течение дня время для использования тех ресурсов, которые уже существуют в организации.
- Поощряйте сотрудников участвовать в технических конференциях по крайней мере один раз в год и делиться со всей командой тем, что они узнали.
- Введите внутренние «дни взлома» (hack days), когда кроссфункциональные команды могут собраться вместе, чтобы работать над проектом.
- Поощряйте команды к организации внутренних «дней яка» (yak days), когда команды собираются вместе, чтобы работать над техническим долгом. Это большие события, потому что технический долг редко находится в приоритете.
- Регулярно проводите внутренние мини-конференции DevOps. Мы уже видели, как организации достигают успеха, используя классический формат DevOpsDays, который сочетает в себе заранее подготовленные дискуссии с «открытыми пространствами», где участники самоорганизуются, чтобы предложить и провести свои собственные сессии.

- Предоставьте сотрудникам специальное время, например, 20% времени или несколько дней после выпуска релиза, для экспериментов с новыми инструментами и технологиями. Выделите бюджет и инфраструктуру для специальных проектов.

Советы по улучшению культуры и поддержке ваших команд

Поскольку реальная ценность лидеров или руководителей проявляется в том, как они усиливают работу своих команд, возможно, самая ценная работа, которую они могут сделать, — это взращивание и поддержка сильной организационной культуры среди тех, кому они служат: их команд. Это позволяет специалистам, которые работают с ними и для них, работать с максимальной эффективностью, создавая ценность для организации.

В этом разделе мы перечислим несколько простых способов, с помощью которых менеджеры, руководители команд и даже специалисты-практики могут поддерживать культуру в своих командах. Наше исследование показывает, что три вещи сильно коррелируют с эффективностью доставки программного обеспечения и способствуют сильной командной культуре: кроссфункциональное сотрудничество, климат для обучения и инструменты.

Активируйте кроссфункциональное сотрудничество следующими способами.

- Выстраивайте доверительные отношения с вашими коллегами в других командах. Построение доверия между командами — это самое важное, что вы можете сделать. Доверие строится на сдержанных обещаниях, открытом общении и предсказуемом поведении даже в стрессовых ситуациях. Ваши команды смогут работать более эффективно, а эти отношения будут сигнализировать организации, что кроссфункциональное сотрудничество ценится.
- Поощряйте специалистов к перемещению между отделами. Админы или инженеры могут обнаружить, что на позиции в другом отделе они приобрели интересные им навыки. Такое горизонтальное перемещение может быть невероятно ценным для обеих команд. Практикующие специалисты приносят ценную информацию о процессах и проблемах в свою новую команду, а члены их предыдущей команды получают в лице этих людей естественную точку входа, когда дело доходит до сотрудничества.
- Активно ищите, поощряйте и вознаграждайте работу, которая облегчает сотрудничество. Убедитесь, что успех поддается

повторению, и обратите внимание на скрытые факторы, облегчающие сотрудничество.

Используйте упражнения по тестированию аварийного восстановления для построения отношений в команде

Многие крупные технологические компании проводят тесты аварийного восстановления или «игровые дни», в которых сбои моделируются или фактически создаются в соответствии с заранее подготовленным планом и команды должны работать вместе для поддержания или восстановления необходимого уровня обслуживания.

Крипа Кришнан, директор по облачным операциям в Google, руководит командой, которая планирует и выполняет подобные упражнения. Она сообщает: «Для того чтобы успешно преодолевать такие тесты, организация сначала должна принять системные и технологические сбои как средство обучения… Мы разрабатываем тесты, требующие от инженеров из нескольких групп, которые обычно не работают вместе, взаимодействовать друг с другом. Таким образом, если когда-либо произойдет настоящая крупномасштабная катастрофа, у этих людей уже будут прочные рабочие отношения» (ACMQueue, 2012).

Помогите создать климат обучения.

- Создайте бюджет на обучение и продвигайте его внутри команды. Подчеркните, насколько организация ценит климат обучения, вкладывая ресурсы в возможности формального образования.
- Обеспечьте вашей команде ресурсы для участия в неформальном обучении и пространство для изучения идей. Обучение часто происходит вне формального образования. Некоторые компании, такие как 3M и Google, выделили часть времени (15% и 20% соответственно) для целенаправленного свободного мышления и изучения сторонних проектов.
- Сделайте ошибки безопасными. Если неудача наказуема, люди не будут пробовать новые вещи. Рассмотрение неудач как возможностей для обучения и исследование ошибок для выработки способов улучшения процессов и систем без необходимости обвинений помогают людям чувствовать себя комфортно, принимая риски (разумные), и способствуют созданию культуры инноваций.
- Создайте возможности и пространства для обмена информацией. Независимо от того, проводите ли вы еженедельные короткие беседы, или выделяете ресурсы для ежемесячного «учебного» обеда, установите регулярные возможности для сотрудников поделиться своими знаниями.

- Поощряйте обмен и инновации, устраивая демонстрационные дни и форумы. Это позволяет командам делиться друг с другом тем, что они создали, отметить свою работу и учиться друг у друга.

Эффективно используйте инструменты.

- Убедитесь, что ваша команда сама может выбрать инструменты. Если нет веской причины не делать этого, специалисты-практики должны выбирать свои собственные инструменты. Если они смогут построить инфраструктуру и приложения так, как они хотят, то они с гораздо большей вероятностью будут вкладываться в свою работу. Это подтверждено данными: один из основных факторов, влияющих на удовлетворенность работой, заключается в том, чувствуют ли сотрудники, что у них есть инструменты и ресурсы для выполнения своей работы (см. Главу 10). Мы также рассматриваем это в наших данных как один из предопределяющих факторов непрерывной доставки: команды, которые уполномочены выбирать свои собственные инструменты, повышают эффективность доставки ПО (см. Главу 5). Если ваша организация должна стандартизировать инструменты, убедитесь, что закупки и финансы действуют в интересах команд, а не наоборот.
- Сделайте мониторинг приоритетом. Усовершенствуйте свою инфраструктуру и систему мониторинга приложений, а также убедитесь, что вы собираете информацию о нужных сервисах и используете эту информацию с пользой. Наглядность и прозрачность, которые обеспечиваются эффективным мониторингом, бесценны. Проактивный мониторинг был тесно связан с эффективностью и удовлетворенностью работой в нашем исследовании, и это является ключевой частью прочной технической основы (см. Главы 7 и 10).

Хотя многие истории успеха DevOps подчеркивают фантастические усилия вовлеченных технических команд, наш опыт и наши исследования показывают, что технологические преобразования выигрывают от действительно вовлеченных трансформационных лидеров, которые способны поддерживать и усиливать работу своих команд. Эта поддержка обеспечивает ценность для бизнеса, поэтому организациям было бы разумно рассматривать развитие лидерства как инвестиции в свои команды, свои технологии и свои продукты.

ЧАСТЬ II

ИССЛЕДОВАНИЯ

Чтобы обосновать то, что было изложено в Части I, нам пришлось выйти за пределы кейсов и историй и перейти к строгим методам исследования. Это дало нам возможность выявить практики, которые являются наиболее явными предикторами успеха для всех организаций любого размера в любой отрасли.

В первой части книги мы обсудили результаты данной исследовательской программы и описали, почему технологии на сегодняшний день являются ключевым двигателем ценности и отличительным фактором для всех организаций.

Теперь мы представим научные обоснования результатов исследования, изложенных в Части I.

Глава 12

Научный подход, стоящий за этой книгой

Каждый день наши новостные ленты пестрят стратегиями, призванными облегчить нашу жизнь, сделать нас счастливее и помочь покорить мир. Также мы слышим истории о том, как какие-то компании и организации используют различные стратегии для трансформации своих технологий и завоевывают рынок. Но как нам понять, какие предпринимаемые нами действия соответствуют изменениям, которые мы наблюдаем в своей среде, и какие действия будут способствовать этим изменениям? Как раз тут самое время для строгого первичного исследования. Но что мы подразумеваем под терминами «строгое» и «первичное»?

Первичные и вторичные исследования

Исследования делятся на два больших класса: первичные и вторичные. Ключевое различие заключается в том, кто занимается сбором данных. Вторичное исследование использует данные, которые были собраны другими лицами. Примеры вторичных исследований, вероятно, вам уже знакомы, — это отчеты о книгах или исследованиях, которые мы выполняли в школе или университете: мы собирали существующую информацию, обобщали ее и (будем надеяться) добавляли наши собственные идеи к тому, что уже было найдено. Типичные примеры также включают в себя тематические кейсы и некоторые отчеты о рыночных исследованиях.

Отчеты о вторичных исследованиях могут быть ценными, особенно если существующие данные сложно найти, резюме отличается глубиной или отчеты представляются с регулярными интервалами. Вторичные исследования, как правило, проводятся быстрее и стоят дешевле, но

может оказаться, что данные не совсем подходят для исследовательской группы, так как они связаны с уже существующими данными.

В противоположность этому первичные исследования включают сбор новых данных исследовательской группой. Примером первичных исследований является перепись населения в Соединенных Штатах.

Каждые десять лет исследовательская группа собирает новые данные для представления демографической статистики и статистики численности населения по стране. Ценность первичных исследований состоит в том, что они могут сообщать о ранее неизвестной информации и предоставлять данные, которые недоступны в имеющихся базах данных. Первичные исследования дают исследователям больше возможностей и контроля над вопросами, с которыми они обращаются, хотя, как правило, они являются более дорогими и трудоемкими. Эта книга и «Отчеты о состоянии DevOps» основаны на первичных исследованиях.

Количественные и качественные исследования

Исследования могут быть качественными или количественными. Качественное исследование представляет собой любое исследование, данные которого не представлены в числовой форме. Они могут включать интервью, записи в блогах, сообщения в Twitter, подробные данные журналов регистрации событий и наблюдения этнографов. Многие считают, что исследование методом опросов является качественным, потому что не связано с компьютерными системами, однако это совсем не обязательно: все зависит от типа задаваемых вопросов. Качественные данные являются очень описательными и могут дать исследователям возможность обнаружить больше открытий и новых моделей поведения, особенно в сложных или новых областях. Тем не менее анализ таких данных зачастую сложнее и дороже; а усилия по анализу качественных данных с использованием автоматизированных средств часто заключаются в кодировании данных в числовую форму, что делает их количественными.

Количественные исследования — это любые исследования с использованием данных, которые включают в себя числа. Это могут быть системные данные (в числовом формате) или данные фондовых рынков. Системные данные — это любые данные, полученные из наших инструментов, и одним из примеров являются данные журнала регистрации событий. Они также могут включать в себя данные опроса, если в ходе опроса задаются вопросы, которые фиксируют ответы в числовом формате — предпочтительно на некой шкале. Представленное в данной книге исследование является количественным, так как его

данные были собраны с использованием такого инструмента опроса, как шкала Ликерта.

Что такое шкала Ликерта?

Шкала Ликерта записывает ответы и присваивает им числовое значение. Например, ответу «категорически не согласен» присваивается значение 1, нейтральному ответу — 4, а ответу «полностью согласен» — 7. Этим обеспечивается последовательный подход к оценке всех предметов исследования и численная база для использования исследователями в своем анализе.

Типы анализа

Количественные исследования позволяют проводить статистический анализ данных. Согласно концепции, представленной доктором Джеффри Ликом в Высшей школе общественного здравоохранения Блумберга при Университете Джона Хопкинса36 (Лик, 2013), существует шесть типов анализа данных (приводятся ниже в порядке возрастания сложности). Сложность обусловлена знаниями, которые требуются от аналитиков данных, затратами на проведение анализа, а также временем, необходимым для его выполнения. Вот эти уровни анализа.

1. Описательный.
2. Исследовательский.
3. Дедуктивно-прогностический.
4. Прогностический.
5. Казуальный.
6. Механистический.

Представленный в настоящей книге анализ относится к первым трем категориям концепции доктора Лика. Мы также приводим описание дополнительного типа анализа, классификацию, которая не полностью вписывается в вышеуказанную структуру.

Описательный анализ

Описательный анализ используют в отчетах о переписи населения. Данные обобщаются и вносятся в отчет, то есть описываются. Данный тип анализа требует наименьших усилий и зачастую выполняется в качестве первого шага в анализе данных, чтобы помочь исследовательской группе разобраться с имеющимся у них набором данных (и в более широком смысле их выборкой и, по возможности, популяцией). В некоторых случаях отчет останавливается на описательном анализе, как в случае с отчетами о переписи населения.

Что такое популяция и выборка и почему они важны?

Когда речь идет о статистике и анализе данных, термины «популяция» и «выборка» имеют особое значение. Популяция представляет собой всю группу объектов, которые интересуют вас в исследовании; это могут быть все люди, которые проходят через технологическую трансформацию, все инженеры надежности систем в организации или даже каждая запись в журнале регистрации событий за определенный период времени. Выборка — это часть популяции, которая тщательно определена и отобрана. Это набор данных, на основе которого исследователи проводят свой анализ. Выборка используется, когда популяция слишком велика или труднодоступна для исследования. Точные и правильно подобранные методы выборки важны для того, чтобы убедиться, что выводы, сделанные на основе анализа выборки, верны для всей популяции.

Наиболее распространенным примером описательного анализа является государственная перепись, в ходе которой обобщаются и представляются статистические данные о населении. Другие примеры включают большинство отчетов поставщиков и аналитиков, которые собирают данные и представляют сводную итоговую статистику о состоянии использования инструментов в какой-либо отрасли или об уровне образования и сертификации у технических специалистов. Процент компаний, которые начали свое путешествие по пути Agile или DevOps, как сообщает Forrester (Клавенс и соавторы, 2017), отчет IDC (International Data Corporation) о средней стоимости простоев (Эллиот, 2014) и обзор заработной платы аналитиков данных от компании O'Reilly (Кинг и Магулас, 2016) относятся именно к этой категории.

Эти отчеты очень полезны в качестве индикатора текущего состояния отрасли, показывающего, где в настоящее время находятся референтные группы (такие как население или отрасли), где они когда-то были и куда указывают тенденции. Однако описательные выводы хороши настолько, насколько хороши основной проект исследования и методы сбора данных. Любые отчеты, которые ставят своей целью представление базовой популяции, должны тщательно производить выборку для этой популяции и учитывать любые ограничения. Обсуждение этой проблематики выходит за рамки данной книги.

Примером описательного анализа в данной книге является демографическая информация об участниках нашего опроса и организациях, в которых они работают: из каких они стран, размер организаций, отрасль, а также их должности и пол (см. Главу 10).

Исследовательский анализ

Исследовательский анализ — это следующий уровень статистического анализа. Это широкая категория, которая ищет взаимосвязи между данными и может включать визуальные отображения для

идентификации шаблонов в данных. На данном этапе также могут быть обнаружены резко отклоняющиеся значения, хотя исследователям следует проявлять бдительность и убедиться в том, что такие отклоняющиеся значения, по сути, являются именно отклоняющимися значениями, а не полноправными участниками группы.

Исследовательский анализ — крайне интересная и увлекательная часть исследовательского процесса. Для тех, кто мыслит разнопланово, он зачастую представляет собой этап, на котором генерируются и предлагаются новые идеи, новые гипотезы и новые исследовательские проекты. Здесь мы открываем, как связаны переменные в наших данных, и ищем возможные новые связи и отношения. Однако это не должно стать завершением процесса для команды, которая желает сделать заявления о прогнозе или причинной связи.

Многие люди слышали фразу «корреляция не подразумевает причинной связи», но что она значит? Анализ, проведенный на стадии исследования, включает в себя изучение корреляции, но не причинно-следственной связи. Корреляция рассматривает, насколько согласованно (или несогласованно) изменяются две переменные, однако она не говорит нам, предсказывает или вызывает ли изменение одной переменной изменение другой переменной. Корреляционный анализ говорит нам лишь о том, изменяются ли две переменные совместно или противоположно; он не говорит нам, почему это происходит или что вызывает такое изменение. Согласованное изменение двух переменных всегда может быть связано с третьей переменной или просто случайностью.

Фантастический и интересный набор примеров, которые выделяют высокие корреляции, основанные на случайных связях, можно найти на сайте Spurious Correlations («ложные корреляции». — Прим. ред.)37. Его автор Тайлер Виджен рассчитал примеры высококоррелированных переменных, которые, как подсказывает нам здравый смысл, не являются прогностическими и, уж конечно, не являются причинными. Например, он демонстрирует (рис. 12.1), что потребление сыра на душу населения сильно коррелирует с числом людей, которые умерли, запутавшись в простынях (с корреляцией 94,71% или r = 0,9471; см. следующую сноску). Конечно, потребление сыра не является причиной удушения простынями. (А если и является, то какого именно сорта сыра?) Также трудно себе представить удушение простынями, вызывающее потребление сыра, если только его не подают в обязательном порядке на похоронах и поминках в масштабах целой страны. (И, опять же, сыр какого сорта? Что рисует нам мрачные маркетинговые возможности.) И все же, когда мы отправляемся «выуживать данные», наше сознание заполняет историю, поскольку наши наборы данных взаимосвязаны и часто именно этим и значимы. Именно поэтому так важно помнить, что корреляция — это всего лишь

этап исследования: мы можем сообщить о корреляциях, а затем переходить к более сложным типам анализа.

Есть несколько примеров корреляций, о которых идет речь в нашем исследовании и в этой книге, поскольку мы осознаем важность и ценность понимания того, каким образом взаимосвязаны вещи в нашей среде. Во всех случаях мы говорили о корреляциях Пирсона[38], представляющих собой тип корреляции, наиболее часто используемый в современном бизнес-контексте.

Рис. 12.1. **Ложная корреляция: потребление сыра и удушение простынями на душу населения**

Дедуктивно-прогностический анализ

Третий уровень анализа, дедуктивный, на сегодняшний день является одним из наиболее распространенных в области исследования бизнеса и технологий. Его также называют дедуктивно-прогностическим. Он помогает нам понять влияние HR-политики, организационного поведения и мотивации, а также то, как технологии влияют на такие результаты, как удовлетворенность пользователей, эффективность команды и производительность организации. Дедуктивное проектирование применяется тогда, когда чисто экспериментальное проектирование невозможно, а предпочтение отдается полевым экспериментам, например, в бизнесе, когда сбор данных осуществляется в сложных организациях, а не в стерильных лабораторных условиях, и компании не будут жертвовать своей прибылью, чтобы вписаться в контрольные группы, определенные исследовательской командой.

Чтобы избежать проблем с «выуживанием данных» и нахождением ложных корреляций, гипотезы должны быть основаны на теории. Данный тип анализа является первым шагом в научном методе.

Многие из нас знакомы с научным методом: выдвинуть гипотезу, а затем ее проверить. На этом уровне анализа гипотеза должна основываться на развитой и хорошо обоснованной теории.

Всякий раз, когда мы в этой книге говорим о результатах, оказывающих влияние или являющихся движущей силой, наш проект исследований использовал именно этот третий тип анализа. Хотя некоторые предполагают, что использование теоретически обоснованного проекта исследования ведет к предвзятости утверждений, именно так и творится наука.

Хотя постойте-ка — почти так. Наука не делается просто путем подтверждения того, что ищет исследовательская группа. Наука заключается в выдвижении гипотез, разработке исследований для их проверки, сборе данных, а затем проверке заявленных гипотез. Чем больше доказательств мы находим в поддержку гипотезы, тем больше у нас уверенности в ней. Этот процесс также помогает избежать опасностей, возникающих в ходе «выуживания данных», — ложных корреляций, которые могут случайно существовать, но не имеют реальной причины или объяснения, кроме случайности.

Примерами гипотез, проверенных с помощью дедуктивного анализа в нашем проекте, являются методы непрерывной доставки и архитектурные практики, повышающие эффективность доставки программного обеспечения, доставка программного обеспечения, положительно влияющая на эффективность организации, и организационная культура, оказывающая позитивное влияние как на доставку программного обеспечения, так и на эффективность организации. В этих случаях из статистических методов использовались либо множественная линейная регрессия, либо регрессия частичных наименьших квадратов. Более подробно эти методы описаны в Приложении C.

Прогностический, казуальный и механистический анализ

Последние уровни анализа мы не включили в свое исследование, потому что у нас не было данных, необходимых для работы такого рода. Мы кратко изложим их здесь для полноты картины и чтобы удовлетворить ваше любопытство.

- Прогностический анализ используется для предсказания или прогнозирования будущих событий на основе предыдущих событий. Типичные примеры включают в себя прогнозирование затрат или коммунальных платежей в коммерческой деятельности. Прогнозировать очень трудно, особенно когда вы пытаетесь заглянуть в отдаленное будущее. Этот тип анализа обычно требует исторических данных.

- Казуальный анализ считается золотым стандартом, но он сложнее прогностического анализа и является наиболее сложным для большинства технологических и бизнес-ситуаций. Этот тип анализа обычно требует проведения рандомизированных исследований. Распространенным видом казуального анализа в бизнесе является A/B-тестирование при разработке прототипов или веб-сайтов, когда можно собрать и проанализировать случайные данные.
- Механистический анализ требует наибольших усилий и редко применяется в бизнесе. В данном виде анализа практикующие его специалисты вычисляют конкретные изменения, которые необходимо внести в переменные, чтобы вызвать конкретное поведение, которое будет наблюдаться при определенных условиях. Он чаще всего используется в естественных или технических науках и не подходит для сложных систем.

Классификационный анализ

Еще одним видом анализа является классификация, или кластерный анализ. В зависимости от контекста, проекта исследования и методов анализа классификация может рассматриваться как исследовательский, прогностический или даже казуальный анализ. В данной книге мы пользуемся классификацией, когда говорим о наших командах доставки программного обеспечения с высокой, средней и низкой эффективностью. Это может быть знакомо вам в других контекстах, когда вы слышите о профилях клиентов или анализе потребительской корзины. На высшем уровне процесс работает следующим образом: в алгоритм кластеризации вводятся классификационные переменные и выделяются значимые группы.

В нашем исследовании мы применили данный статистический метод, используя переменные темпа и стабильности, чтобы постараться понять и определить, были ли различия в том, как команды разрабатывали и доставляли программное обеспечение, и в чем эти различия выражались. И вот что мы сделали: четыре наши переменных показателя эффективности технологии — частоту развертывания, время выполнения изменений, среднее время восстановления и процент сбоев при изменениях — мы поместили в алгоритм кластеризации и наблюдали, какие появились группы. Мы видим отчетливые, статистически значимые различия, где участники с высокой эффективностью лучше по всем четырем показателям, участники с низкой эффективностью хуже по всем четырем показателям, а имеющие среднюю эффективность значительно лучше отстающих, но существенно хуже лидеров. Подробнее см. в Главе 2.

Что такое кластеризация?

Для кабинетных (или профессиональных) статистиков может быть интересно, что мы использовали иерархическую кластеризацию. Мы

выбрали ее вместо кластеризации по методу k-средних по нескольким причинам. Во-первых, у нас не было никаких теоретических или иных идей о том, сколько групп ожидать, до анализа. Во-вторых, иерархическая кластеризация позволила нам исследовать отношения типа «родитель — ребенок» в возникающих кластерах, что дало нам лучшую интерпретируемость. Наконец, у нас не было огромного набора данных, поэтому вычислительная мощность и скорость не были проблемой.

Исследование в данной книге

Представленное в данной книге исследование охватывает четырехлетний период времени и было проведено самими авторами. Поскольку данное исследование является первичным, оно идеально подходит для решения имеющихся у нас исследовательских задач — в частности, какие возможности определяют эффективность доставки программного обеспечения и организационную эффективность. Настоящий проект был основан на количественных данных опроса, что позволяет нам произвести статистический анализ для проверки наших гипотез и обнаружить факторы, которые повышают эффективность доставки программного обеспечения.

В следующих главах мы обсудим шаги, которые мы предприняли для того, чтобы данные, которые мы собрали в ходе опросов, были достоверными и надежными. Затем мы разберемся, почему опросы могут быть предпочтительным источником данных для измерения — как в таком исследовательском проекте, как наш, так и в ваших собственных системах.

Глава 13

Введение в психометрию

Два наиболее распространенных вопроса, которые мы получаем о нашем исследовании: почему мы используем опросы (его мы подробно рассмотрим в следующей главе) и уверены ли мы в том, что можем доверять данным, собранным с помощью опросов (в отличие от данных, сгенерированных системой). Все это подпитывается сомнениями в качестве наших базовых данных и, следовательно, в достоверности наших результатов. Скептицизм в отношении достоверных данных справедлив, поэтому давайте начнем отсюда: насколько вы можете доверять данным, которые были собраны в ходе опроса. Большая часть опасений происходит от типов опросов, с которыми приходится сталкиваться многим из нас, а именно пуш-опросов (также известных как пропагандистские опросы), быстрых опросов и опросов, написанных теми, кто не имеет надлежащей подготовки в области исследований.

Пуш-опросы — это опросы с четко обозначенной и очевидной повесткой дня, на вопросы таких опросов сложно отвечать честно, если только вы заранее не согласны с точкой зрения «исследователя». Примеры таких опросов часто встречаются в политике. Например, в феврале 2017 года президент Трамп выпустил свой Опрос о подконтрольности ведущих СМИ (Mainstream Media Accountability Survey), и общественность мгновенно отреагировала с тревогой. Всего несколько фрагментов из опроса подчеркивают сомнения в вопросах и их способности собирать данные ясно и беспристрастно.

1. «Считаете ли вы, что ведущие СМИ выпустили несправедливые сообщения о нашем движении?» Это был первый вопрос, и довольно тонкий, но он задает тон остальной части опроса. Пользуясь конструкцией «наше движение», он предлагает респонденту занять позицию «мы против них». «Ведущие СМИ» также является отрицательно заряженным термином в данном политическом цикле.
2. «Знали ли вы, что данный опрос был выпущен, чтобы показать, что большинство американцев на самом деле поддержали указ президента Трампа о временном ограничении?» Такой вопрос является ярким примером пуш-опроса, когда вопрос скорее пытается навязать респонденту информацию, чем интересуется его мнением или восприятием происходящего. Вопрос также использует психологическую тактику, предполагающую, что «большинство американцев» поддерживают указ о временном ограничении, и апеллирующую к желанию читателя принадлежать к данной группе.
3. «Согласны ли вы со стратегией президента Трампа пробиться сквозь создаваемую СМИ шумиху и напрямую донести наше послание до людей?» Используется резкий, поляризующий язык, характеризующий все СМИ как «шум» — негативную коннотацию в данном политическом климате.

На этом примере мы видим, почему люди так скептически относятся к опросам. Если это ваш единственный контакт с ними, тогда, конечно, им нельзя доверять! Никакие данные по любому из этих вопросов не могут с достоверностью сообщить о восприятии или мнении респондента.

Даже без такого очевидного примера, как пуш-опрос, некачественные опросы встречаются повсеместно. Чаще всего они являются результатом действий благонамеренных, но неподготовленных авторов опросов, которые надеются получить некоторое представление о мнениях своих клиентов или сотрудников. Их типичными слабыми местами являются:

- наводящие вопросы. Вопросы должны позволять респонденту отвечать без навязывания какой-либо точки зрения. Например,

формулировка «Как бы вы описали рост Наполеона?» лучше, чем «Наполеон был низкого роста?»;

- провокационные вопросы. Вопросы не должны вынуждать респондентов давать неверные для них ответы. К примеру, «Где вы сдавали свой сертификационный экзамен?» не предусматривает возможности того, что респондент не сдавал сертификационный экзамен;
- несколько вопросов в одном. Вопрос должен спрашивать только об одной вещи. Например, «Ваши клиенты и центр управления сетями уведомляют вас о сбоях?» не конкретизирует ту часть вопроса, на которую отвечал респондент. Клиенты? Центр управления сетями? И то и другое? Или ни то, ни другое?
- непонятный язык. В вопросах должен использоваться язык, знакомый вашим респондентам, и при необходимости должны быть предоставлены разъяснения и примеры.

Потенциальным слабым местом большинства опросов, используемых в бизнесе, является то, что для сбора данных используется только один вопрос. Иногда их называют быстрыми опросами, и они довольно часто используются в маркетинговых исследованиях и бизнес-исследованиях. Они могут быть полезны, если они основаны на хорошо составленных и правильно понятых вопросах. Однако важно, чтобы из такого вида опросов были сделаны только узкоспециальные выводы. Примером правильного быстрого опроса является индекс лояльности клиентов (NPS). Он был тщательно разработан и изучен, хорошо понят, а его использование и области применения детально задокументированы. Хотя существуют более качественные статистические показатели удовлетворенности пользователей и сотрудников, например, те, которые используют большее количество вопросов (например, Ист и соавторы, 2008), зачастую легче получить от вашей аудитории единственный показатель. Кроме того, преимущество NPS состоит в том, что он превратился в отраслевой стандарт и, следовательно, его легко использовать для сравнения между командами и компаниями.

Доверие к данным со скрытыми конструкциями

Учитывая все вышеизложенное, как мы можем доверять данным, полученным на основе данных опроса? Как мы можем быть уверены в том, что никто своей ложью не исказит результаты? Наши исследования используют скрытые конструкции и статистический анализ для представления достоверных данных или, по крайней мере, обеспечивают разумную уверенность в том, что данные говорят нам именно то, что мы слышим.

Скрытая конструкция представляет собой способ измерения чего-либо, что не может быть измерено напрямую. Мы можем спросить о температуре в помещении или времени отклика веб-сайта — это то, что можно изменить напрямую.

Хорошим примером того, что нельзя измерить напрямую, является организационная культура. Мы не можем измерить «температуру» команды или организационной культуры. Нам придется измерить культуру путем измерения ее составных частей (называемых явными переменными), и мы измеряем эти составные части с помощью опросов. То есть, когда вы описываете кому-то организационную культуру команды, вы, вероятно, упоминаете ряд характеристик. Они и являются составными частями организационной культуры. Мы будем измерять каждую (в качестве явных переменных), и вместе они представят организационную культуру команды (скрытую конструкцию). И использование опросов для сбора этих данных является целесообразным, поскольку культура — это жизненный опыт тех, кто работает в команде.

При работе со скрытыми конструкциями либо чем-то еще, что мы хотим измерить в ходе исследования, важно начать с четкого определения и понимания того, что именно мы хотим измерить. В данном случае нам нужно решить, что мы подразумеваем под организационной культурой. Как мы обсуждали в Главе 3, организационная культура, которая нас интересовала, способствовала росту доверия и оптимизации потока информации. Мы ссылались на типологию, предложенную доктором Роном Веструмом (2004), показанную в таблице 13.1.

Таблица 13.1

Типология организационной культуры по Веструму

Патологические (ориентированные на власть)	Бюрократические (ориентированные на правила)	Производительные (ориентированные на результат)
Низкий уровень сотрудничества	Средний уровень сотрудничества	Высокий уровень сотрудничества
«Гонцов» с плохими новостями «пристреливают»	«Гонцов» игнорируют	«Гонцов» обучают
Уклонение от ответственности	Узкая область ответственности	Разделение рисков
Горизонтальные связи не допускаются	Горизонтальные связи допускаются	Горизонтальные связи поощряются
Ошибки ведут к поиску козла отпущения	Ошибки регулируются правилами	Ошибки ведут к исследованиям
Инновации подавляются	Инновации приводят к проблемам	Инновации внедряются

Как только мы выявили конструкцию, мы составляем вопросы для опроса. Очевидно, что концепция организационной культуры,

предложенная Веструмом, не может быть охвачена только одним вопросом; организационная культура является многогранной идеей. Спрашивая «Какова ваша организационная культура?», вы рискуете, что этот вопрос будет понят разными людьми по-разному. Пользуясь скрытыми конструкциями, мы можем задать один вопрос для каждого аспекта исходной концепции.

Если мы определим конструкцию и правильно пропишем ее элементы, она будет работать концептуально, как диаграмма Венна, при этом каждый вопрос такого опроса будет охватывать связанный с ним аспект основной концепции.

После сбора данных мы можем использовать статистические методы для проверки того, что показатели действительно отражают основополагающую концепцию. Как только это будет сделано, мы сможем объединить эти показатели, чтобы получить единое значение.

В данном примере комбинация вопросов для каждого аспекта организационной культуры становится нашим показателем для всей концепции. Усредняя наши показания по каждому пункту, мы получаем своего рода «температуру организационной культуры». Преимущество скрытых конструкций в том, что, используя несколько показателей (называемых явными переменными, — частей скрытой переменной, которые могут быть измерены) для охвата основной концепции, вы помогаете оградить себя от неверных оценок и злого умысла. Каким образом? Для этого есть несколько способов, которые применимы для системных данных с целью измерения эффективности вашей системы.

1. Скрытые конструкции помогают нам тщательно продумывать, что именно мы хотим измерить и каким образом мы определяем наши конструкции.
2. Они дают нам несколько точек зрения на поведение и эффективность системы, которую мы наблюдаем, помогая нам устранить ложные данные.
3. Они затрудняют искажение наших результатов вследствие использования одного источника неверных данных (по недоразумению или злому умыслу).

Скрытые конструкции помогают нам тщательно продумывать то, что мы измеряем

Первый способ, с помощью которого скрытые конструкции помогают нам избежать недостоверных данных, заключается в том, чтобы помочь нам тщательно продумать, что мы хотим измерить и как мы определяем наши конструкции. Время, потраченное на продумывание этого процесса, может помочь нам избежать неверных измерений. Отступите на шаг назад и подумайте о том, что вы пытаетесь измерить и как вы

будете это измерять. Давайте вернемся к нашему примеру измерения культуры.

Часто мы слышим, что культура важна при технологических трансформациях, поэтому мы хотим ее измерить. Может быть, нам следует просто спросить наших сотрудников и коллег: «Хороша ли ваша культура?» или «Нравится ли вам культура вашей команды?» А если бы они ответили «да» (или «нет»), что бы это вообще значило? О чем именно это нам говорит?

Что мы подразумеваем под культурой и как ее интерпретировал респондент в первом вопросе? О какой культуре идет речь: о культуре вашей команды или о культуре вашей организации? Если на самом деле мы говорим о культуре на рабочем месте, то какие аспекты этой культуры мы имеем в виду? Или нас действительно больше интересует ваша национальная принадлежность и культура? Предположим, что все поняли часть вопроса о культуре, но что здесь имеется в виду под «хорошо»? Означает ли это доверие? Удовольствие? Или же что-то совершенно иное? Возможно ли вообще, чтобы культура была исключительно хорошей или исключительно плохой?

Второй вопрос немного лучше, потому что мы указываем на то, что спрашиваем о культуре на уровне команды. Однако мы по-прежнему не даем читателю никакого представления о том, что мы подразумеваем под «культурой», поэтому мы можем получить данные, отражающие очень разные представления о том, что же такое командная культура. Еще одна проблема здесь заключается в том, что мы спрашиваем, нравится ли человеку его командная культура. А что значит, что «культура нравится»?

Такой пример может показаться утрированным, но мы видим, что люди делают такие ошибки все время (хотя и не вы, дорогой читатель). Сделав шаг назад, чтобы тщательно продумать, что вы хотите измерить, и действительно определив, что мы подразумеваем под культурой, мы можем получить более надежные данные. Когда мы слышим, что культура важна в технологических преобразованиях, мы имеем в виду культуру, которая имеет высокую степень доверия, способствует информационному потоку, выстраивает мосты между командами, поощряет инновации и разделяет риски. Имея в виду данное определение командной и организационной культуры, мы можем понять, почему типология, представленная доктором Веструмом, так хорошо подходит для нашего исследования.

Скрытые конструкции дают нам несколько точек зрения на наши данные

Второй способ, с помощью которого скрытые конструкции помогают нам избежать недостоверных данных, — это обеспечение нескольких представлений о поведении и эффективности системы, которую мы

наблюдаем. Это позволяет нам идентифицировать любые ложные показатели, которые остались бы незамеченными, если бы они были единственным параметром, которым мы бы пытались охватить поведение системы.

Давайте вернемся к вопросу измерения организационной культуры. Чтобы начать измерять данную конструкцию, мы сначала предложили несколько аспектов организационной культуры, основанных на определении доктора Веструма. Исходя из этих аспектов, мы написали несколько элементов[39].

Позже в данной главе мы более подробно поговорим о написании хороших элементов опроса и проверке их качества. Как только мы соберем данные, мы сможем провести ряд статистических тестов, чтобы убедиться в том, что эти элементы действительно измеряют одну и ту же базовую идею — скрытую конструкцию. Эти тесты проверяют следующее.

- Дискриминантная валидность: тесты проводятся, чтобы убедиться, что элементы, которые не должны быть связаны, на самом деле не связаны (к примеру, убедитесь, что элементы, которые, по нашему мнению, не затрагивают организационную культуру, на самом деле с организационной культурой не связаны).
- Конвергентная валидность: тесты проводятся, чтобы убедиться, что элементы, которые должны быть связаны, действительно связаны (к примеру, если предполагается, что элементы должны измерять организационную культуру, то они действительно измеряют организационную культуру).

В дополнение к тестированию валидности в целях наших исследований проводятся также испытания надежности. Это гарантирует то, что пункты опроса одинаково читаются и интерпретируются всеми, кто принимает в нем участие. Это также называется внутренней согласованностью.

Взятые вместе, статистические тесты на достоверность (валидность) и надежность подтверждают наши измерения. Они предшествуют любому анализу.

В случае организационной культуры Веструма у нас есть семь элементов, отражающих организационную культуру команды.

В моей команде…

- ведется активный поиск информации;
- «гонцов» с сообщениями о неудаче и другими плохими новостями не наказывают;
- ответственность распределена;
- кроссфункциональное сотрудничество поощряется и вознаграждается;
- ошибки ведут к исследованиям;

- новые идеи приветствуются;
- сбои рассматриваются в первую очередь как возможности для улучшения системы.

Используя шкалу от «1 = категорически не согласен» до «7 = полностью согласен», команды могут быстро и легко измерить свою организационную культуру.

Эти элементы были проверены и признаны статистически достоверными и надежными. То есть они измеряют то, для чего они предназначены, и люди обычно читают и интерпретируют их неизменным образом. Вы также заметите, что мы запрашивали данные элементы для команды, а не для организации. Мы приняли это решение при создании элементов опроса как отступление от первоначальной концепции Веструма, так как организации могут быть очень крупными и иметь очаги различных организационных культур. Кроме того, люди точнее отвечают за свою команду, чем за свою организацию. Это помогает нам собирать более точные показатели.

Скрытые конструкции помогают нам защититься от ложных данных

Здесь необходимо кое-что прояснить. Скрытые конструкции, которые периодически перепроверяются статистическими методами и демонстрируют хорошие психометрические свойства, помогают нам защититься от ложных данных.

Как так? Давайте мы объясним.

В предыдущем разделе мы говорили о достоверности и надежности — статистических тестах, которые мы можем сделать, чтобы убедиться, что элементы опроса, которые измеряют скрытую конструкцию, принадлежат друг другу. Когда наши конструкции проходят все эти статистические тесты, мы говорим, что они обладают хорошими психометрическими свойствами. Хорошей практикой будет периодически пересматривать их, чтобы убедиться, что ничего не изменилось, особенно если вы подозреваете изменения в системе или окружающей среде.

В примере с организационной культурой все эти элементы являются надежными показателями конструкции. Вот еще один пример конструкции, где тесты выявили возможности для улучшения нашего измерения. В данном случае мы были заинтересованы в изучении уведомлений об ошибках. Такими элементами были следующие.

- Уведомления об ошибках мы получаем в первую очередь из сообщений клиентов.
- Уведомления об ошибках мы получаем в первую очередь из центра управления сетями.

- Мы получаем предупреждения об ошибках из систем протоколирования и мониторинга.
- Мы отслеживаем работоспособность системы на основании предупреждений о превышении пороговых значений (например, если загрузка процессора превышает 90%).
- Мы отслеживаем работоспособность системы на основе предупреждений о скорости изменений (например, загрузка процессора за последние 10 минут увеличилась на 25%).

В предварительном проекте опроса мы провели пилотное тестирование конструкции с привлечением порядка 20 технических специалистов и элементами опроса, представленными вместе (то есть они измеряли одну и ту же базовую конструкцию). Однако когда мы завершили наш окончательный, более масштабный сбор данных, мы провели тесты для подтверждения конструкции. В этих заключительных тестах мы обнаружили, что эти элементы на самом деле измеряли два разных параметра. То есть, когда мы провели наши статистические тесты, они не подтвердили единую конструкцию, а вместо этого выявили две конструкции. Первые два элемента измеряют одну конструкцию, которая, как представляется, охватывает «уведомления, поступающие извне автоматизированных процессов»:

- уведомления об ошибках мы получаем в первую очередь из сообщений клиентов;
- уведомления об ошибках мы получаем в первую очередь из центра управления сетями.

Второй набор элементов измеряет другую конструкцию — «уведомления, поступающие из систем», или «упреждающие уведомления о сбоях»:

- мы получаем предупреждения об ошибках из систем протоколирования и мониторинга;
- мы отслеживаем работоспособность системы на основании предупреждений о превышении пороговых значений (например, если загрузка процессора превышает 90%);
- мы отслеживаем работоспособность системы на основе предупреждений о скорости изменений (например, загрузка процессора за последние 10 минут увеличилась на 25%).

Если бы мы только спросили наших респондентов, отслеживают ли они сбои, с помощью одного вопроса из данного опроса, мы бы не узнали о важности учета того, откуда эти уведомления поступают. Кроме того, если один из этих источников уведомлений изменяет свое поведение, наши статистические тесты его поймают и предупредят нас. Аналогичная концепция применима в отношении системных данных. Мы можем использовать несколько показателей наших систем для отслеживания поведения системы, и эти показатели могут пройти наши проверки

достоверности. Тем не менее мы должны продолжать проводить периодические проверки этих показателей, поскольку они могут измениться. Наше исследование показало, что эта вторая конструкция — упреждающее уведомление о сбоях — является технической возможностью, которая предсказывает эффективность доставки программного обеспечения.

Как скрытые конструкции могут использоваться для системных данных

Некоторые из этих идей о скрытых конструкциях также распространяются и на системные данные: они помогают нам избежать ложных данных, используя ряд показателей для поиска похожих моделей поведения, и они помогают нам продумать то, что мы на самом деле пытаемся сделать. Предположим, что мы хотим измерить производительность системы. Мы можем просто собрать данные о времени отклика определенного аспекта системы. Чтобы найти похожие шаблоны в данных, мы можем собрать несколько фрагментов данных из нашей системы, которые могут помочь нам понять ее время отклика. Если подумать о том, что мы действительно пытаемся измерить, — о производительности, мы можем рассмотреть различные аспекты производительности и то, как еще она может быть отражена в системных метриках. Мы можем понять, что нас интересует концептуальный показатель производительности системы, который сложно измерить напрямую и который лучше фиксируется с помощью нескольких связанных показателей.

Здесь важно отметить следующее: все показатели являются опосредованными. То есть они представляют для нас какую-то идею, даже если мы сознательно не признаем ее. Это так же справедливо в отношении системных данных, как и в отношении данных опроса. К примеру, мы можем использовать время отклика в качестве косвенного показателя производительности нашей системы.

Если только одна из точек данных используется в качестве барометра и эта одна точка ошибочна или может стать таковой, мы этого не знаем. Например, изменение исходного кода, собирающего метрики, может повлиять на один показатель; и если были собраны данные только об одном показателе, вероятность того, что мы зафиксируем такое изменение, невелика. Однако, если мы соберем несколько метрик, у нас будет больше шансов обнаружить это изменение в поведении. Скрытые конструкции дают нам механизм защиты от неудачных показателей или злого умысла. Это справедливо как для опросов, так и для системных данных.

Глава 14

Зачем использовать опрос

Теперь, когда мы знаем, что данным нашего опроса можно доверять, то есть у нас есть разумная уверенность в том, что данные наших хорошо разработанных и надежно проверенных конструкций, созданных на основе психометрического опроса, говорят нам то, что мы о них думаем, зачем нам использовать опрос? И зачем вообще кому бы то ни было использовать опрос?

Команды, желающие понять эффективность своего процесса доставки программного обеспечения, часто начинают с инструментирования процесса доставки ПО и цепочки инструментов для получения данных (в этой книге собранные таким образом данные мы называем системными данными). Действительно, сегодня несколько представленных на рынке инструментов предлагают анализ такой метрики, как срок выполнения. Так почему же кому-то захочется собирать данные из опросов, а не просто из своего набора инструментов?

Существует несколько причин для использования данных опроса. В этой главе мы кратко представим некоторые из них.

1. Опросы позволяют быстро собирать и анализировать данные.
2. Измерение полного программного стека (full stack) системными данными является сложной задачей.
3. Измерение исключительно системными данными является сложной задачей.
4. Данным опроса можно доверять.
5. Некоторые вещи можно измерить только с помощью опросов.

Опросы позволяют быстро собирать и анализировать данные

Зачастую самой веской причиной использования опросов является скорость и простота сбора данных. Это особенно актуально для новых или разовых усилий по сбору данных или же для сбора данных, который распространяется или пересекает организационные границы. Данные для исследования в этой книге собирались четыре раза.

Каждый раз на сбор данных у нас уходило от четырех до шести недель, сбор осуществлялся со всего мира и от тысяч респондентов, представляющих тысячи организаций. Представьте себе сложность (на самом деле невозможность) получения системных данных от такого количества команд за тот же период времени. Даже само получение юридических разрешений было бы невозможным, не говоря уже о спецификации и передаче данных.

Но давайте предположим, что у нас получилось собрать системные данные от нескольких тысяч респондентов со всего мира за четыре недели. Следующим шагом будет очистка данных и их анализ. Анализ данных для «Отчетов о состоянии DevOps» обычно занимает 3–4 недели. Многие из вас, вероятно, работали с системными данными; еще больше из вас, вероятно, имели удовольствие (а скорее всего, боль) от объединения и сортировки электронных таблиц Excel. Представьте себе получение приблизительных системных данных (или, возможно, таблиц планирования капиталовложений) от нескольких тысяч команд со всего мира. Представьте себе задачу по очистке, систематизации и последующему анализу таких данных и будьте готовы предоставить результаты для отчетности в течение трех недель.

К основной задаче очистки данных и выполнения анализа добавляется сложная проблема, которая может поставить под сомнение всю вашу работу и, вероятно, стать самым большим ограничением: сами данные, а если более конкретно, то исходное значение самих данных.

Вероятно, вы видели это и в своих собственных организациях: разные команды могут ссылаться на абсолютно разные (или немного разные) показатели под одним и тем же названием. Приведем два примера: «время выполнения» (которое мы определяем как время от коммита до кода в развертываемом состоянии) и «время производственного цикла» (которое некоторые определяют как время от начала разработки до кода в развертываемом состоянии). Однако эти два термина часто используются как взаимозаменяемые и довольно часто с ними возникает путаница, несмотря на то что они измеряют разные параметры.

Что же произойдет, если одна команда пользуется термином «время производственного цикла», а другая — «время выполнения», но обе они измеряют при этом одно и то же? Или если обе команды называют некий процесс «временем выполнения», но измеряют при этом два разных параметра? А потом мы собрали данные и теперь пытаемся провести анализ… Но мы не знаем наверняка, какие из этих переменных какие. Это создает существенные проблемы измерения и анализа.

Тщательно сформулированные и проработанные опросы, которые были проверены, помогают решить эту проблему. Все респонденты теперь работают с одними и теми же элементами, терминами и определениями. Не важно, как они называют что-либо в своей организации, важно то, о чем их спрашивают в данном опросе. Значение имеет то, о чем их спрашивают, и поэтому намного более важными являются качество и ясность элементов опроса. Но, как только завершается работа по составлению опроса, работа по очистке и подготовке данных для анализа выполняется быстрее и проще.

В ходе тщательного исследования дополнительные анализы (например, общий метод проверки смещений) проводятся в целях того, чтобы убедиться, что сам опрос не внес искажений в результаты, и ответы

проверяются на предмет смещений в ответах более ранних и более поздних респондентов (см. Приложение C).

Измерение полного программного стека системными данными является сложной задачей

Даже если ваша система сообщает хорошие и полезные данные (предположение, которое, как мы знаем по опыту, довольно часто бывает неверным и, как правило, должно удостоверяться опытным путем), такие данные редко бывают исчерпывающими. То есть вы действительно можете быть уверены в том, что он на 100% измеряет поведение системы, которая вас интересует?

Давайте проиллюстрируем это на примере. Один из авторов часть своей карьеры проработала в качестве инженера по производительности в компании IBM, работая над корпоративными дисковыми системами хранения данных. Роль ее команды заключалась в диагностике и оптимизации производительности таких устройств, включая операции чтения, записи, кэширования и перестроения RAID-массива40 в условиях различной рабочей нагрузки. После работы над несколькими инициативами блок работал хорошо и у команды в доказательство этого были метрические показатели со всех уровней системы. Периодически команда все еще получала от клиентов отчеты о том, что блок работал медленно. Команда каждый раз проводила расследование, но первые один-два отчета были отклонены командой, так как у них было подтверждение того, что производительность была хорошей: это демонстрировали все системные журналы!

Однако по мере того, как команда начала получать все больше сообщений о низкой производительности, потребовался более серьезный анализ. Несомненно, и у клиентов, и у филд-инженеров могут быть мотивы лгать, например, для получения скидок исходя из нарушения условий соглашений о качестве предоставления услуг. Но в отчетах клиентов и отчетах филд-инженеров прослеживалась своя закономерность — все ссылались на одинаковую медлительность. Хотя эти данные, полученные от людей, не имели такой же степени точности, как системные журналы (например, минутная точность в сообщаемом времени отклика по сравнению с миллисекундной точностью из файлов системных журналов), это дало команде достаточно данных, чтобы знать, где искать. Они подсказали алгоритмы и подали сигналы, за которыми нужно было следовать в своей работе.

Так что же это было? Оказалось, что сам блок работал очень хорошо. Команда оборудовала каждый уровень стека и охватила все, что можно было охватить… в блоке. Чего команда не охватила, так это интерфейс. То, как клиенты взаимодействовали с блоком, вело к существенному

снижению производительности. Команда быстро организовала небольшую группу для управления этой новой областью, и вскоре вся система работала на пике производительности.

Не задавая людям вопросов о производительности системы, команда не поняла бы, что происходит. Уделяя время для проведения периодических оценок, включающих в себя точку зрения технических специалистов, которые занимаются созданием и доставкой вашей технологии, вы сможете обнаружить узкие места и другие ограничения в вашей системе. Опросив всех членов своей команды, вы можете помочь избежать проблем, связанных с наличием ряда излишне позитивных или излишне негативных ответов41.

Измерение исключительно системными данными является сложной задачей

Связанная с этим причина использования опросов заключается в неспособности зафиксировать все, что происходит, посредством системных данных, потому что ваши системы знают только о том, что происходит в границах системы. И наоборот, люди могут наблюдать все, что происходит в системе и вокруг нее, и сообщают об этом. Давайте проиллюстрируем это на примере.

Наше исследование продемонстрировало, что использование контроля версий является ключевой возможностью в эффективности доставки программного обеспечения. Если мы хотим знать, в какой степени команда использует систему контроля версий для всех производственных артефактов, мы можем спросить команду. Участники команды могут рассказать это нам, так как они видят всю рабочую картину. Однако, если мы хотим измерить это через систему, у нас есть существенные ограничения. Система в состоянии сказать нам только о том, что она видит, то есть сколько файлов или репозиториев регистрируется в системе контроля версий. Но это абстрактное число без контекста смысла не имеет.

В идеале мы хотели бы знать процент файлов или репозиториев, которые находятся в управлении версиями, однако этого система сообщить нам не сможет: это потребует подсчета зарегистрированных и незарегистрированных файлов, а система не знает, сколько файлов не находится в системе контроля версий. Система лишь обеспечивает наглядное представление находящихся в ней объектов, и в этом случае использование систем контроля версий — это то, что не поддается точному измерению системными журналами и измерительными приборами.

Люди также не будут обладать совершенными знаниями или абсолютным доступом в системы, однако, если вы полностью

игнорируете восприятие и опыт профессионалов, работающих над вашими системами, вы теряете важный взгляд на них.

Данным опроса можно доверять

Нас часто спрашивают, почему можно доверять каким-либо данным, полученным в результате опросов, и, соответственно, результатам опросов. Это можно проиллюстрировать мыслительным упражнением, которое мы иногда используем, обращаясь к группам технических специалистов и задавая им вопросы о работе. Задайте себе (или кому-то из своих знакомых, работающих в области разработки и доставки программного обеспечения) эти вопросы.

1. Доверяете ли вы данным опроса? Безусловно, этот первый вопрос получает крайне мало поддержки. Большинство представителей нашей целевой аудитории, к сожалению, склонны ожидать от людей худшего и полагают, что респонденты в опросах будут лгать, или же ожидают, что авторы и разработчики опроса попытаются исказить вопросы, чтобы получить желаемые результаты, — тема, которую мы рассматривали ранее.
2. Доверяете ли вы своей системе или данным журнала регистрации событий? Этот вопрос часто получает больше поддержки и одобрения. Мы чувствуем себя комфортно с данными, которые поступают из наших систем, поскольку мы уверены, что такие данные не были подтасованы. Итак, давайте перейдем к третьему вопросу.
3. Вы когда-либо получали из вашей системы недостоверные данные? Исходя из нашего опыта, практически каждый получал неверные данные из системных файлов. И хотя многие верят, что системные данные не подвергаются искажению, системы создаются людьми (а следовательно, и данные, поступающие из таких систем), а люди совершают ошибки. Или, если мы действительно предполагаем, что в наших системах могут существовать «злоумышленники», достаточно лишь одного такого «злоумышленника», чтобы внедрить код, который заставит систему выдавать нам ошибочные данные.

Злоумышленники и системные данные

Сюжет ставшего культовой классикой фильма «Офисное пространство» (Office Space) построен на этой теме: злоумышленники вносят изменения в финансовое программное обеспечение, которое депонирует небольшие суммы денег (известные как «ошибка округления») на некий личный счет. Об этой ошибке округления в финансовых отчетах не сообщается. Это яркий пример недостоверных системных данных.

Если мы так хорошо знакомы с недостоверными данными в наших системах, почему мы так им доверяем и при этом все же скептически

относимся к данным опроса? Возможно, это связано с тем, что, будучи инженерами и техническими специалистами, мы понимаем, как работают наши системы. Мы полагаем, что сможем обнаружить ошибки в данных, поступающих из этих систем, а затем будем знать, как их исправить.

И наоборот, работа с данными опроса кажется непривычной, особенно для тех, кто не изучал создание опросов и психометрические методы. Но обзор концепций, представленный в Части II, призван продемонстрировать, что есть шаги, которые можно предпринимать, чтобы сделать данные опроса более надежными. Они включают использование тщательно выработанных показателей, скрытых конструкций и статистических методов для подтверждения достоверности и надежности измерений.

Сравните два наших случая: системные данные и данные опроса. В случае с системными данными один или несколько человек могут изменить данные в файлах системного журнала. Это может быть высокомотивированный злоумышленник с корневым доступом (или высоким уровнем системного доступа) или это может быть разработчик, допустивший ошибку, которая не была обнаружена в ходе проверки или тестирования. Они оказывают значительное влияние на качество данных, так как у вас, вероятно, есть только одна или несколько точек данных, на которые бизнес обращает внимание. В этом случае ваши необработанные данные являются неверными, а вы можете не замечать этого месяцами, годами или вообще всегда.

В случае с данными опроса несколько высокомотивированных «злоумышленников» могут лгать, отвечая на вопросы, и их ответы могут исказить результаты всей группы. Их влияние на данные зависит от размера опрашиваемой группы. В исследовании, проведенном для данной книги, у нас было более 23 000 респондентов, чьи ответы мы объединили вместе. Чтобы внести заметные различия, потребовалось бы несколько сотен человек, которые бы скоординированно и организованно лгали, то есть они должны были бы лгать о каждом элементе в скрытой конструкции в одинаковой степени и в одинаковом направлении. В этом случае использование опроса действительно защищает нас от злоумышленников. Для обеспечения сбора достоверных данных предпринимаются дополнительные шаги; например, все ответы являются анонимными, что помогает респондентам чувствовать себя в безопасности и делиться честными отзывами.

Вот почему мы можем доверять данным нашего опроса или, по крайней мере, у нас есть разумная уверенность в том, что данные говорят нам то, о чем мы думаем: мы пользуемся скрытыми конструкциями, а также тщательно и продуманно прописываем критерии нашего опроса, избегая использования любых пропагандистских элементов; мы выполняем ряд статистических тестов, чтобы подтвердить, что наши показатели соответствуют психометрическим стандартам достоверности и

надежности; и у нас есть большой набор данных от респондентов со всего мира, что служит защитой от ошибок или злоумышленников.

Некоторые вещи можно измерить только с помощью опросов

Некоторые вещи можно измерить только с помощью опросов. Когда мы хотим спросить о восприятии, чувствах и мнениях, опрос зачастую является единственным способом. Мы еще раз укажем на наш предыдущий пример с организационной культурой.

У людей часто будет возникать желание полагаться на объективные данные, чтобы определить что-то вроде организационной культуры. Объективные данные не подвержены влиянию со стороны чувств или эмоций; напротив, субъективные данные отражают восприятие или чувства человека по отношению к ситуации. В случае организационной культуры команды часто обращаются к объективным показателям, потому что они хотят быстрее собрать данные (например, из систем управления персоналом), и все равно еще остается беспокойство по поводу того, что люди лгут о своих чувствах. Проблема с использованием переменных из систем управления персоналом для определения культуры состоит в том, что такие переменные редко являются прямым отображением. Например, распространенным показателем «хорошей» организационной культуры является удержание сотрудников, и, наоборот, показателем «плохой» организационной культуры является текучесть кадров.

С этим определением связано несколько проблем, потому что существует много факторов, влияющих на то, остается ли кто-то или нет в команде или организации. Например:

- если сотрудник получает предложение от другой компании с существенным повышением зарплаты и уходит, в этом случае текучесть кадров может не иметь ничего общего с культурой;
- если супруг или партнер сотрудника получает предложение о работе, которое требует переезда, и ваш сотрудник тоже решает переехать, в таком случае текучесть кадров, вероятно, не будет иметь ничего общего с культурой;
- если сотрудник решает начать карьеру в другой области или продолжить обучение, это может не иметь ничего общего с культурой, а скорее связано с его личным выбором. На самом деле один из авторов знает случай, когда сотрудник работал в очень вдохновляющей, поддерживающей компании и в отличной команде. И именно эта великолепная командная атмосфера побудила его следовать за своей мечтой и изменить свою карьеру, чтобы ставить новые цели и добиваться их. В этом случае сильная культура привела к текучести кадров, а не наоборот;

- эти показатели могут быть подтасованы. Если руководитель сотрудника узнает, что он активно ищет работу, руководитель может его уволить, чтобы тот не портил статистику по текучести кадров. И наоборот, если менеджеров поощряют за удержание членов команды, они могут препятствовать переходам сотрудников из своих команд, сохраняя людей, даже если их командная культура плоха.

Текучесть может быть полезным показателем, если мы тщательно продумываем то, что мы собираемся измерять42. Однако в представленных выше примерах мы видим, что текучесть и удержание кадров не очень много говорят нам о нашей команде или организационной культуре, или если они и говорят что-то, то это может быть не тем, о чем мы думали. Если мы хотим понять, как люди относятся к риску, обмену информацией и коммуникации через границы, мы должны спросить их об этом. Да, вы можете использовать другие системные индикаторы, чтобы увидеть, как происходит что-то из перечисленного. Например, вы можете наблюдать за сетевым трафиком, чтобы понять, кто из членов команды чаще других общается между собой, и в течение продолжительного времени вы можете наблюдать тенденции к увеличению или снижению командного общения. Вы даже можете провести семантический анализ, чтобы узнать, имеют ли слова в электронных письмах или чатах в целом негативную или позитивную окраску. Но если вы хотите знать, как люди относятся к рабочей среде и насколько благоприятно она влияет на их работу и их цели, если вы хотите знать, почему они ведут себя именно так, как вы это наблюдаете, вы должны спросить их об этом. И лучшим вариантом сделать это систематическим, надежным способом, который поддается сравнению с течением времени, будет проведение опросов. И это стоит спрашивать. Исследования показали, что организационная культура предсказывает технологии, организационную эффективность и результаты работы и что командная работа и психологическая безопасность являются наиболее важными аспектами в понимании эффективности работы команды (Google, 2015).

Глава 15

Данные для проекта

Данный проект начался с желания понять, как сделать технологию превосходной и каким образом технология может сделать организацию лучше. В частности, мы хотели изучить новые способы, методы и парадигмы, которыми пользуются организации для разработки и доставки программного обеспечения, с акцентом на методологию Agile и бережливые практики, которые простираются от разработки и предполагают приоритет культуры доверия и информационного потока

внутри небольших кроссфункциональных команд, создающих ПО. В начале проекта в 2014 году данная методология разработки и доставки была широко известна как DevOps, и поэтому мы использовали этот термин.

Наш исследовательский проект — поперечный сбор данных[43] на протяжении четырех лет — привлекал профессионалов и организации, знакомых со словом «DevOps» (или хотя бы готовых прочитать электронные письма и сообщения в социальных сетях со словом «DevOps»), которые были нашей целевой аудиторией. Любой хороший исследовательский проект определяет целевую популяцию, и это была наша. Мы выбрали данную стратегию по двум основным причинам.

1. Это позволило нам сфокусироваться на сборе данных. В данном исследовании респондентами стали люди, чья деятельность связана с разработкой и доставкой программного обеспечения, независимо от того, была ли отрасль их материнской организации технологической сама по себе или технологии были ее движущей силой (розничная торговля, банковское дело, телекоммуникации, здравоохранение и ряд других).
2. Это позволило нам сосредоточиться на пользователях, которые были относительно знакомы с идеями DevOps. Наше исследование было нацелено на пользователей, уже знакомых с терминологией технических специалистов и применяющих современные методы разработки и доставки программного обеспечения, независимо от того, считали ли они себя приверженцами DevOps или нет. Это было важно, поскольку время и пространство были ограничены и слишком много времени, затраченного на основные определения и длинные объяснения концепций, таких как непрерывная интеграция и управление конфигурацией, могло создать риск отказа респондентов от участия в исследовании. Если читающий опрос человек должен потратить 15 минут на изучение концепции, чтобы ответить на вопросы о ней, он будет разочарован, раздражен и не завершит опрос.

Этот проект целевого исследования был сильной стороной наших исследований. Ни один исследовательский проект не в состоянии дать ответы на все вопросы, и все проектные решения подразумевают компромиссы. Мы не собирали данные от профессионалов и организаций, которые не были знакомы с такими понятиями, как управление конфигурацией, инфраструктура-как-код и непрерывная интеграция. Не собирая данные по этой группе, мы упускаем контингент, который, вероятно, работает еще хуже, чем наши участники с низкими показателями эффективности. Это означает, что наши сравнения ограничены и мы не обнаруживаем действительно убедительных и радикальных трансформаций, которые возможны. Тем не менее мы получаем объяснительную силу, ограничивая целевую популяцию теми,

кто подпадает под более жесткое определение группы. Такое улучшение объяснительной силы происходит за счет исключения из охвата и анализа поведения тех, кто не использует современные технологические практики для создания и обслуживания программного обеспечения.

Такой отбор данных и проект исследования требовали определенной осторожности. Изучая только тех, кто знаком с DevOps, мы должны были быть осмотрительны в наших формулировках. То есть некоторые из тех, кто ответил на наш опрос, возможно, захотят представить свою команду или организацию в выгодном свете или у них могут быть собственные определения ключевых терминов. Например, все знают (или утверждают, что знают), что такое непрерывная интеграция (НИ), и многие организации заявляют, что непрерывная интеграция является ключевой компетенцией. Поэтому в наших опросах мы никогда не спрашивали респондентов, практикуют ли они непрерывную интеграцию. (По крайней мере, мы не задавали никаких вопросов о НИ, которые были бы использованы для любого прогностического анализа.) Вместо этого мы задали бы вопросы о практиках, которые являются главным аспектом НИ, например, запускаются ли автоматизированные тесты при регистрации кода. С помощью этого нам удалось избежать предвзятости, которая могла бы проникнуть из-за ориентации на пользователей, знакомых с DevOps.

Однако, основываясь на предыдущих исследованиях, нашем собственном опыте и опыте тех, кто проводил технологические трансформации на крупных предприятиях, мы считаем, что многие из наших выводов широко применимы к командам и организациям, проходящим через преобразования. Например, использование контроля версий и автоматизированного тестирования с высокой вероятностью даст положительные результаты независимо от того, использует ли команда методы DevOps, методологию Agile или надеется улучшить свои методы синхронной каскадной разработки (lockstep waterfall development). Аналогично наличие организационной культуры, которая ценит прозрачность, доверие и инновации, вероятно, окажет положительное влияние в технологических организациях независимо от парадигмы разработки программного обеспечения — и так в любой отраслевой вертикали, поскольку данная концепция предсказывает результаты работы в различных контекстах, включая здравоохранение и авиацию.

Как только мы определили нашу целевую популяцию, мы приняли решение по методу выборки: как нам пригласить людей принять участие в опросе. Существуют две обширные категории методов выборки: вероятностная выборка и детерминированная выборка[44]. Мы не смогли использовать методы вероятностной выборки, поскольку это потребовало бы, чтобы каждый член целевой аудитории был известен и имел равные шансы на участие в исследовании. Это невозможно,

поскольку исчерпывающего списка профессионалов в области DevOps в мире просто не существует. Ниже мы объясним это более подробно.

Чтобы собрать данные для нашего исследования, мы рассылали электронные письма и использовали социальные сети. Электронные письма были отправлены по нашим собственным спискам рассылки, которые состояли из технических специалистов и профессионалов, которые работали в DevOps (например, они находились в наших базах данных, потому что участвовали в исследованиях предыдущих лет, были в маркетинговых базах данных компании Puppet в связи с их работой с управлением конфигурациями, присутствовали в базах данных Джина Кима и Джеза Хамбла из-за своего интереса к их книгам и работе в отрасли). Электронные письма также были отправлены по спискам рассылки для профессиональных групп. Особое внимание также уделялось рассылке приглашений группам, включающим недостаточно представленные группы и меньшинства в области технологии. В дополнение к прямым приглашениям по электронной почте мы использовали социальные сети: авторы и спонсоры опроса публиковали ссылки на опрос в Twitter и LinkedIn. Направляя приглашения к участию в опросе из нескольких источников, мы увеличили наши шансы на контакт с большим количеством профессионалов DevOps, в то же время решая проблему ограничений выборки методом снежного кома, который обсуждается ниже.

Чтобы расширить наш охват технических специалистов и организаций, занимающихся разработкой и доставкой ПО, мы также пригласили рефералов. Данный аспект наращивания нашей исходной выборки называется реферальной выборкой, или выборкой методом снежного кома, поскольку выборка растет, собирая дополнительных респондентов по мере своего распространения, точно так же, как растет снежный ком, когда вы катите его по снегу. Выборка методом снежного кома стала подходящим методом сбора данных для этого исследования по нескольким причинам.

- Идентификация популяции тех, кто создает ПО с использованием методологии DevOps, затруднена или невозможна. В отличие от профессиональных организаций, например, в области бухгалтерского учета или гражданского строительства, которые в США имеют государственную сертификацию, такую как CPA (Certified Public Accountants) или PE (Practice of Engineering), централизованного аккредитационного совета, который мог бы выдать нам справочный список профессионалов, просто не существует. Кроме того, мы не могли бы просматривать организационные схемы (даже если они находились в открытом доступе) на предмет названий должностей, поскольку не у всех в названии должности имеется «DevOps» или другие важные для нас ключевые слова. Кроме того, многие технические специалисты, особенно в начале исследовательского проекта, имели

нестандартные должности. Даже если организационные схемы находились в общем доступе, многие названия должностей являются слишком общими, чтобы быть полезными в привлечении участников в исследование (например, «инженер-программист», что может подразумевать разработчиков, работающих в командах, использующих каскадный метод или методы DevOps). Выборка методом снежного кома хорошо подходит для изучения конкретных групп, популяции которых сложно идентифицировать.

- Популяция традиционно неохотно становится объектом изучения. Существует богатая (и неудачная) история организационных исследований технических работников, которые привели к «бережливым трансформациям», что на самом деле означало просто значительное сокращение рабочих мест. Выборка методом снежного кома — это метод, идеально подходящий для популяций, которые зачастую не хотят быть изученными; отсылая других к исследованию, люди могут поручиться за вопросы (заверяя новых участников, что вопросы не являются пропагандой) или даже за репутацию исследователей.

Существуют некоторые ограничения, присущие выборке методом снежного кома. Первое ограничение — это возможность того, что изначально отобранные пользователи (в нашем случае — приглашенные по электронной почте) не являются представителями сообществ, к которым они принадлежат. Мы компенсировали это наличием начального набора приглашений (или информантов), который был максимально большим и разнообразным. Мы достигли этого, объединив несколько списков рассылки, в том числе наш собственный список рассылки опроса, который содержал разнообразный набор респондентов из разных стран и из компаний разной величины. Мы также связались с недостаточно представленными группами и меньшинствами в области технологий через их собственные списки рассылок и организации.

Еще одно ограничение выборки методом снежного кома заключается в том, что на собираемые данные сильно влияют первоначальные приглашения. Это вызывает беспокойство только в том случае, если в целевую аудиторию попадает лишь небольшая группа людей, у которых затем запрашиваются рефералы, и выборка растет уже с этого момента. Мы решили эту проблему, пригласив очень большую и разнообразную группу людей принять участие в исследовании, как описано выше.

Наконец, может возникнуть опасение, что результаты не будут репрезентативными для того, что на самом деле происходит в отрасли, что у нас могут остаться слепые пятна, которые мы не видим в наших данных. Мы решаем эту проблему несколькими способами.

Во-первых, сообщая наши выводы, мы не просто полагаемся на результаты ежегодных исследований; мы активно взаимодействуем с отраслью и сообществом, чтобы убедиться в том, что мы знаем, что

происходит, и осуществляем триангуляцию наших результатов с новейшими тенденциями. Это означает, что мы активно ищем обратную связь по нашему опросу через сообщество на конференциях, а также через коллег и саму отрасль; после чего мы сравниваем записи, чтобы понять, какие появляются тенденции, никогда не полагаясь только на один источник данных. При возникновении каких-либо расхождений или несоответствий мы пересматриваем наши гипотезы и выполняем очередную итерацию опроса.

Во-вторых, у нас есть внешние тематические эксперты в отрасли, которые ежегодно пересматривают наши гипотезы, чтобы убедиться в их актуальности. В-третьих, мы изучаем существующую литературу, чтобы найти закономерности в других областях, которые могут дать более глубокое понимание нашего исследования. И наконец, мы каждый год запрашиваем у сообщества информацию и идеи и используем их при разработке исследования.

ЧАСТЬ III

ТРАНСФОРМАЦИЯ

Мы представили наши выводы о том, какие возможности важны для улучшения доставки программного обеспечения и результатов деятельности организации. Однако принятие и применение такой информации для изменения вашей организации является чрезвычайно сложной и пугающей задачей. Вот почему мы так рады, что Стив Белл и Карен Уитли Белл согласились написать главу о лидерстве и организационной трансформации и поделиться своим опытом и идеями, чтобы направить читателей на их собственном пути.

Стив и Карен являются пионерами бережливого (Lean) ИТ. Они применяют независимый от методологии подход, опираясь на различные практики: DevOps, Agile, Scrum, канбан (kanban), бережливый стартап, Kata, Обея (Obeya), стратегическое развертывание и другие, — в зависимости от культуры и ситуации, чтобы обучать и поддерживать лидеров в развитии высокоэффективных практик и возможностей организационного обучения.

В Части III они опираются на свой опыт в ING Netherlands — глобальном банке с более чем 34,4 млн клиентов по всему миру и 52 000 сотрудников, в том числе более чем 9000 инженеров, — чтобы показать, почему и как лидерство, менеджмент и командные практики способны изменить культуру. А это, в свою очередь, обеспечивает устойчивую высокую производительность в сложных и динамичных условиях.

Стив и Карен расширяют наше видение, выводя его за пределы взаимодействия команды, управленческих и лидерских практик, за пределы умелого внедрения DevOps, а также за пределы преодоления

разобщенности — все это необходимо, но недостаточно. Здесь мы видим эволюцию всесторонней, комплексной организационной трансформации, полностью вовлеченной и согласованной с целями предприятия.

Глава 16

Высокоэффективное лидерство и менеджмент

Авторы: Стив Белл и Карен Уитли Белл

Лидерство действительно оказывает существенное влияние на результаты. …Хороший лидер влияет на способность команды доставлять код, разрабатывать надежные системы и применять бережливые принципы к тому, как команда управляет своей работой и разрабатывает продукты. Все это, «как показывает исследование, оказывает ощутимое влияние на прибыльность, производительность и долю рынка организации. Они также оказывают влияние на удовлетворенность клиентов, эффективность и способность достигать организационных целей»45. Тем не менее Николь, Джез и Джин также отмечают, что «роль лидерства в технологической трансформации была одной из наиболее недооцененных проблем в DevOps».

Почему так? Почему технические специалисты постоянно стремились улучшить подход к разработке и развертыванию программного обеспечения, а также стабильность и безопасность инфраструктуры и платформ, однако в значительной степени упускали из виду (или не понимали) способ руководства, управления и поддержания этих усилий? Это справедливо как для крупных традиционных предприятий, так и для цифровых «аборигенов».

Давайте рассмотрим данный вопрос не в контексте прошлого — почему мы это не сделали, а в контексте настоящего и будущего: почему мы должны улучшить то, как мы возглавляем и управляем ИТ46, и на самом деле переосмыслить то, как каждый на предприятии взаимодействует с технологиями.

Мы находимся в процессе полной трансформации способа создания, доставки и потребления ценности. Наша способность быстро и эффективно представлять, разрабатывать и доставлять связанную с технологией ценность для повышения качества обслуживания клиентов становится ключевым конкурентным преимуществом. Но пиковые технические характеристики — это лишь одна часть конкурентного преимущества, необходимая, но не достаточная.

Мы можем достичь великолепных результатов в быстрой разработке и предоставлении надежных, безопасных, высокотехнологичных

продуктов, но как мы узнаем, какой опыт ценят наши клиенты? Как мы расставляем приоритеты в том, что мы создаем, чтобы предпринимаемые каждой командой усилия способствовали продвижению стратегии предприятия в более широком смысле? Как мы учимся у наших клиентов и друг у друга, какие уроки мы выносим из наших действий? И по мере того, как мы учимся, как мы делимся приобретенными знаниями со всеми сотрудниками предприятия и как мы используем это обучение для непрерывной адаптации и инноваций?

Другим необходимым компонентом для поддержания конкурентного преимущества является легкая, высокоэффективная структура менеджмента, которая связывает стратегию предприятия с реальными действиями, ускоряет поток идей в направлении создания ценности, облегчает быструю обратную связь и обучение, а также по максимуму использует и объединяет творческие возможности каждого человека в компании для создания оптимального клиентского опыта. Как выглядит такая структура не в теории, а на практике? И как мы можем улучшить и трансформировать наши собственные лидерские, управленческие и командные практики и поведение, чтобы стать тем предприятием, которым мы стремимся быть?

Структура высокоэффективного менеджмента на практике

На протяжении всей этой книги Николь, Джез и Джин обсуждают несколько практик бережливого менеджмента, которые, как было установлено, коррелируют с высокими организационными показателями, а именно: «прибыльностью, долей рынка и производительностью… [в дополнение к показателям, которые охватывают] более широкие организационные цели — то есть цели, которые выходят за рамки простых показателей прибыли и доходов»47. Каждая из этих практик в той или иной степени синергична и взаимозависима с другими. Чтобы проиллюстрировать, как эти лидерские, управленческие и командные практики работают вместе, и показать фундаментальное мышление, которое делает их возможными, мы делимся опытом ING Netherlands, глобального финансового института, ставшего пионером цифрового банкинга и получившего признание за свое ориентированное на клиента технологическое лидерство. Сегодня ИТ является драйвером цифрового преобразования ING.

«Вы должны понять почему, а не просто копировать поведение»48, — говорит Яннес Смит, ИТ-менеджер интернет-банка и направления Omnichannel в ING Netherlands, который семь лет назад решил поэкспериментировать с путями развития организационного обучения среди своих команд. Существует много способов описать эту практику управления в действии. Пожалуй, лучший способ — это взять вас на виртуальную экскурсию — хотя бы на страницах книги. (ING с радостью

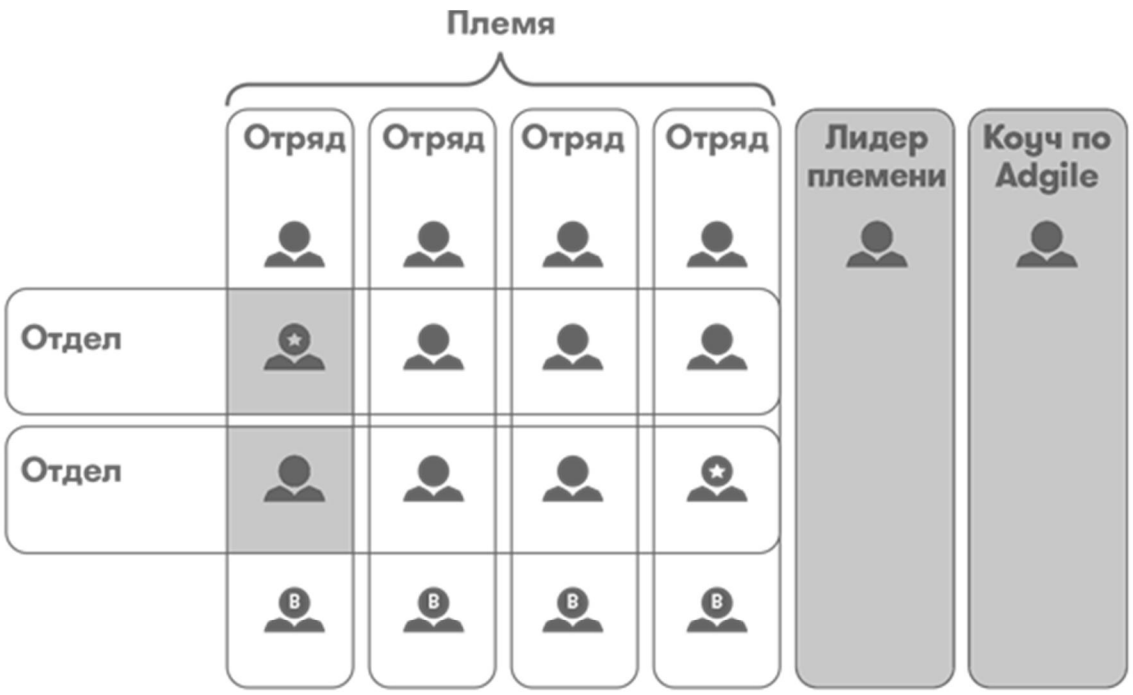
делятся историей своего обучения, но они не готовы показать вам свои «шпаргалки»!) Мы поделимся с вами тем, что представляют собой события одного дня в ING, показав вам, как практики и процедуры соединяются, чтобы создать обучающую организацию и обеспечить высокую эффективность и ценность.

То, что вы видите сегодня, мало похоже на то, что мы впервые наблюдали, когда периодически посещали так называемые учебные лагеря, чтобы переосмыслить, как Яннес и его менеджеры возглавляли команды и управляли ими. Как и многие корпоративные ИТ-организации, они были расположены за пределами основного кампуса и рассматривались многими как функция, а не как жизненно важный вклад в реализацию стратегии предприятия. Сегодня же мы входим в главную штаб-квартиру корпорации, где команды Яннеса теперь расположены на один этаж ниже высшего руководства. Пространство открытое и светлое. После прохождения контроля безопасности мы попадаем в большую открытую зону для общения, в которой расположены кафе-бары и закусочные с видом на сады — все это предназначено для создания уютной атмосферы для собраний и обмена идеями. Затем мы входим в апартаменты Племени. Сразу слева от нас находится большая комната со стеклянными стенами, создающими видимость пространства внутри. Это комната Обея (Obeya), где визуализируются работа, действия и приоритеты лидера для команд и всех, кто может запланировать здесь встречу или заглянуть между встречами, чтобы обновить или проверить статус задач. Здесь Яннес регулярно встречается со своими непосредственными подчиненными, где они могут быстро увидеть и понять статус каждой из его стратегических целей. Визуализируются четыре отдельные зоны: стратегическое улучшение, мониторинг эффективности, дорожная карта портфеля задач и действия лидеров, — каждая из которых содержит текущую информацию о целях, пробелах, прогрессе и проблемах. Используется цветовая маркировка — красный и зеленый цвета, чтобы явно обозначать проблемные места. Каждая ИТ-цель напрямую и измеряемым образом связана со стратегией предприятия (см. рис. 16.1).

Рис. 16.1. **Лидерство Обея (панорама на 360°)**

Два года назад ING претерпела значительный сдвиг в сторону многомерной, матричной структуры, организованной по направлениям бизнеса, что позволило обеспечить непрерывный поток потребительской ценности (то, что lean-практики называют потоками создания ценности). Каждое направление бизнеса организовано как отдельная группа

(«племя»), предоставляющая портфель связанных товаров и услуг (например, «племя ипотечных услуг»). Каждое племя состоит из нескольких самоуправляющихся команд, называемых отрядами, каждый из которых отвечает за определенную клиентскую задачу (например, «отряд ипотечных заявок»). Каждый отряд управляется владельцем продукта, которого ведет (в случае ИТ) лидер ИТ-области. Размер команды определяется в соответствии с «правилом двух пицц» Безоса — ни одна команда не может быть настолько большой, чтобы ей потребовалось больше двух пицц. Большинство отрядов являются кроссфункциональными, состоящими из инженеров и маркетологов, сотрудничающих как единая команда с общим пониманием ценности клиента. В ING этот состав команды называется BizDevOps. Недавно они выявили необходимость в новой мостовой позиции, которую они планируют назвать лидером продуктовой области, чтобы объединить несколько тесно связанных отрядов. Эта новая функция не была запланирована — она появилась благодаря опыту и обучению. Есть также отделы, состоящие из членов одной и той же дисциплины (например, отдел аналитиков данных), которые матрично распределены между отрядами и привносят специализированные знания для содействия обучению и повышению квалификации участников отряда. И, наконец, существуют экспертные центры, объединяющие людей с определенными навыками (например, проектировщиков коммуникаций, см. рис. 16.2).



Племя
(совокупность отрядов со взаимосвязанными задачами)

- включает в среднем 150 человек
- наделяет лидера племени полномочиями устанавливать приоритеты, распределять бюджеты, взаимодействовать с другими племенами для обмена знаниями и идеями

Коуч по Agile

- обучает отдельных людей и отряды созданию высокоэффективных команд

Отряд
(базовая единица новой Agile-организации)

- включает не более 9 человек, автономная и самоуправляющаяся структура
- включает представителей различных функций, работающих в одном пространстве
- сквозная ответственность за достижение целей клиентов
- может менять функциональный состав с изменением задачи

Владелец продукта
(член отряда, не является лидером)

- отвечает за координацию работы отряда
- отвечает за портфель заказов, список задач и расстановку приоритетов

Отдел
(развивает экспертные знания и навыки в отрядах)

Руководитель отдела

- отвечает за один отдел
- выполняет руководящие функции по отношению к членам отряда (личное развитие, наставничество, подбор персонала, управление показателями эффективности)

Владелец продукта

Руководитель отдела

Рис. 16.2. **Новая организационная модель ING в концепции Agile не имеет фиксированной структуры — она постоянно развивается. (Источник: ING)**

Мы идем дальше от комнаты Обея Яннеса в сопровождении его коучей по непрерывному внутреннему совершенствованию: Дэвида Богартса,

Иаиля Шуера, Пола Волхоффа, Лидевия ван дер Шеера и Ингеборга Тен Берге (David Bogaerts, Jael Schuyer, Paul Wolhoff, Liedewij van der Scheer, Ingeborg Ten Berge). Вместе они формируют небольшой, но эффективный экспертный отряд по бережливому лидерству (Lean Leadership Experience Squad) и обучают лидеров, руководителей отделов, владельцев продуктов и руководителей ИТ-областей, которые, в свою очередь, тренируют сотрудников своих отделов или отрядов, создавая эффект рычага для масштабируемых изменений поведения и культуры.

Чуть впереди находится рабочее пространство команды — открытая площадка с окнами и стенами, покрытыми визуальными материалами (их собственная Обея), которые позволяют команде контролировать эффективность в режиме реального времени и видеть препятствия, статусы задач и другую информацию, представляющую ценность для отряда. Через центр этого пространства проходит ряд столов и стульев с регулируемой высотой, чтобы члены отряда могли видеть друг друга через свои мониторы. Стулья имеют различную форму и цвет, что делает пространство визуально интересным и эргономичным. Визуальные материалы отряда обладают некоторыми общими характеристиками; сходство в дизайне комнат Обея позволяет коллегам извне с первого взгляда сразу же понять определенные аспекты работы, что способствует коллективному обучению. Стандартные рекомендации включают визуализацию целей, текущей производительности и пробелов, новых и обострившихся проблем, спроса, НЗП и завершенных задач. Визуализация спроса помогает расставить приоритеты и сохранить небольшой объем НЗП. В то же время визуализация работы отряда имеет некоторые отличия, что признает ее уникальность, и каждый отряд сам определяет, какая информация и какие способы ее визуализации лучше всего подходят, чтобы преуспеть в своей работе.

Когда мы проходим мимо, отряд проводит свое короткое ежедневное совещание (стендап), где происходит быстрое обучение и обратная связь. Стоя перед визуальной доской, отображающей спрос и НЗП, каждый участник кратко сообщает, над чем он/она работает (НЗП), какие есть препятствия и что было завершено. Во время того, как они говорят, визуальные материалы обновляются. Эти стендапы обычно длятся около 15 минут; они значительно сократили время, которое люди проводят на собраниях, по сравнению со временем, которое занимают ставшие привычными ежедневные встречи.

Во время совещаний проблемы не решаются, но существует стандартная практика, обеспечивающая их быстрое решение. Если проблема требует взаимодействия с другим членом отряда, это отмечается и эти сотрудники обсудят ее позже в тот же день. Если для решения проблемы требуется поддержка лидера ИТ-области, проблема отмечается и получает более высокий статус. Лидер ИТ-области может либо быстро решить вопрос, либо взять его на обсуждение и решить на

своем стендапе с другими лидерами ИТ-области или племени. После того как вопрос решен, информация быстро передается обратно. Проблема остается визуализированной до тех пор, пока она не будет решена. Аналогично, если проблема носит технический характер, она будет передана в соответствующий отдел или экспертный центр. Эта модель вертикальной и горизонтальной коммуникации является стандартной практикой лидерства, известной также как метод «кэтчбол» («поймай мяч» — catchball) (см. рис. 16.3).

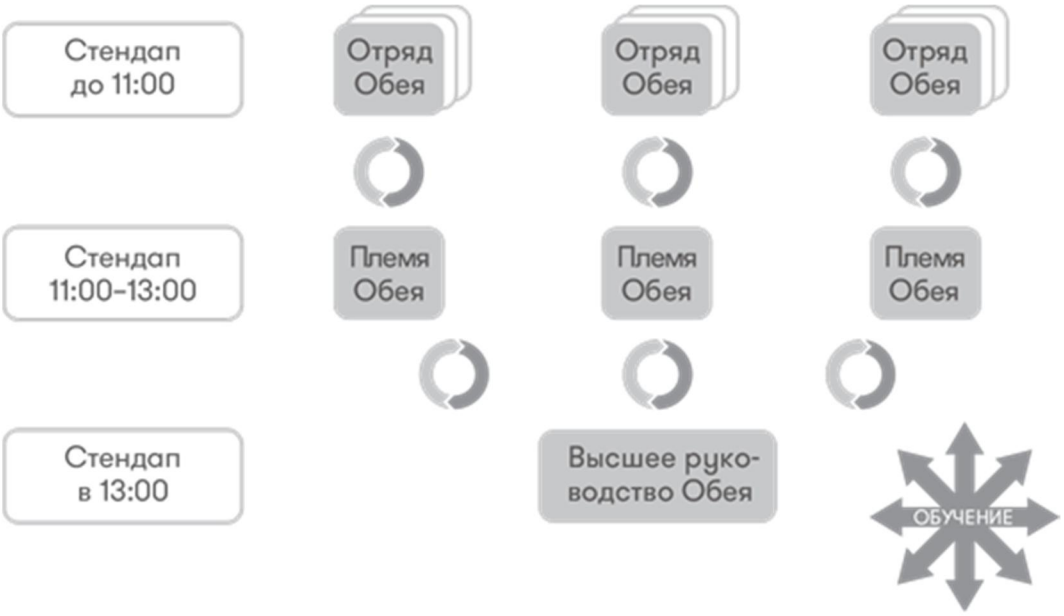


Рис. 16.3. **График стендапов и кэтчболов**

Используя ту же коммуникационную структуру, любое другое актуальное обучение распространяется между отрядами, отделами, экспертными центрами и племенами, создавая естественный вертикальный и горизонтальный поток процесса обучения во всех измерениях организации. Это позволяет командам самостоятельно определять, как лучше всего организовать свою работу для поддержки общей стратегии предприятия, и обеспечивает эффективную расстановку приоритетов. Лидер племени, в данном случае Яннес, также учится у членов команды и отдела, включая уроки, полученные при непосредственном взаимодействии с клиентами. Это позволяет ему адаптировать свое стратегическое мышление и цели, а также делиться знаниями со своими коллегами и руководством.

Эта практика быстрого обмена знаниями, позволяющая командам «на передовой» работы с клиентами узнавать о стратегических приоритетах, а лидерам — о клиентском опыте из взаимодействия команд с клиентами, является формой развертывания стратегии (приверженцы концепции бережливого производства используют термин Хосин Канри — Hoshin Kanri). Она создает на всех уровнях непрерывный быстрый цикл обратной связи обучения, тестирования, проверки и корректировки, также известный как PDCA (Plan — Do — Check — Act: планирование — реализация — контроль — корректировка).

В дополнение к регулярным стендапам с отрядами, владельцами продуктов, руководителями ИТ-областей и отделов лидер племени также периодически посещает отряды, чтобы задавать вопросы. Но не традиционные вопросы, такие как «Почему это не делается?», а скорее «Помогите мне лучше понять проблемы, с которыми вы сталкиваетесь», «Помогите мне понять, что вы изучаете» и «Что я могу сделать, чтобы лучше поддержать вас и команду?». Такого рода наставническое поведение нелегко дается некоторым лидерам и менеджерам. Это требует реальных усилий, связанных с коучингом, наставничеством и моделированием, чтобы трансформировать традиционную модель поведения, командования и контроля в модель лидеров-наставников, когда работа каждого заключается в том, чтобы (1) выполнять работу, (2) улучшать работу и (3) развивать людей. Третья цель — развитие людей — особенно важна в технологической области, где автоматизация разрушает многие рабочие места. Для того чтобы люди могли приложить все усилия к работе, которая фактически может уничтожить их рабочее место, им нужна непоколебимая вера в то, что их лидеры ценят их не только за их нынешнюю работу, но и за их способность совершенствоваться и внедрять инновации. Сама работа будет постоянно меняться; лидирующая организация — это та, в которой люди способны быстро учиться и адаптироваться.

Недалеко от этого пространства отряда в застекленном конференц-зале с белыми досками, монитором для телеконференций, мольбертами и цветными удобными стульями мы встречаемся с Джорди де Восом, молодым инженером, вся карьера которого развивалась в условиях нового способа работы Яннеса. Джорди — руководитель отдела, который также направляет усилия к одной из целей стратегического улучшения организации работы (напомним, что существуют стратегические улучшения, мониторинг показателей эффективности и стратегические цели дорожной карты). Джорди делится с другими тем, что он узнает о безопасности команды — психологической безопасности, которая позволяет людям открыто обсуждать проблемы и препятствия, не опасаясь пострадать или быть наказанными. Он рассказывает об этом и других исследованиях, которые он проводит, как он экспериментирует, чтобы понять, что получит наибольший отклик среди отрядов, и какие ощутимые изменения создаются и поддерживаются. Фиксированный процент времени каждого отряда и отдела выделяется на улучшение рабочего процесса. Джорди говорит, что отряды считают деятельность по улучшению обычной работой.

Мы спрашиваем Джорди, каково это — работать в рамках этой культуры. Он задумывается на мгновение, а затем делится историей. Перед племенами Яннеса высшим руководством была поставлена задача стать вдвое более эффективными. «Был жесткий дедлайн и сильное давление. Наш лидер племени, Яннес, пришел к отрядам и сказал: "Если сомневаетесь в качестве, не выпускайте продукт. Я вас прикрою". Так мы

почувствовали, что контролируем качество. Это помогло нам делать правильные вещи».

Качество слишком часто отодвигается на второй план под давлением требования увеличить скорость. Смелый и поддерживающий лидер играет ключевую роль в помощи командам «замедлить, чтобы ускорить». Он дает им разрешение и обеспечивает безопасные условия, чтобы качество оставалось на первом месте, что в долгосрочной перспективе повышает скорость, стабильность и производительность, снижая при этом затраты, задержки и доработки. А больше всего это повышает удовлетворенность и доверие клиентов.

После этой встречи мы минуем еще множество рабочих мест отрядов и застекленных конференц-залов, каждый с одним и тем же набором элементов, но отличающихся цветами, текстурами и обстановкой. Вернувшись в комнату Обея руководства, мы встречаемся за здоровым обедом с командой наставников и размышляем о многих положительных изменениях, которые мы обнаружили с момента нашего последнего визита. Они делятся мыслями о своих текущих проблемах и некоторых подходах, с которыми они экспериментируют, чтобы продолжать распространять и выращивать производительную культуру, сосредоточившись на том, чтобы «идти вглубь, прежде чем идти вширь». Тем не менее давление задачи масштабироваться широко и быстро все еще существует. Прямо сейчас один из членов наставнической команды сфокусирован на поддержке культурных изменений всего в нескольких странах за пределами Нидерландов. Учитывая, что ING работает в более чем 40 странах, организация работы, позволяющая уделять время и внимание обучению, а не идти на крупномасштабные изменения, замечательна.

Еще одна проблема, с которой экспериментируют наставники, — это рассредоточенные команды. С недавней реструктуризацией в некоторых отрядах теперь работают участники более чем из одной страны, поэтому команда наставников экспериментирует и оценивает способы поддержания такого же высокого уровня сотрудничества и обучения среди трансграничных команд (очень трудно виртуально «разделить две пиццы»).

Неудивительно, что некоторые из самых высших лидеров и несколько других лидеров племен хотят иметь свою собственную Обея. Наставническая команда надеется реализовать это достаточно медленно — так, чтобы могло осуществиться реальное обучение. Трансформационное, созидательное лидерство выходит далеко за пределы того, что находится на стенах Обея, и того, как вы говорите об этом. «Как лидер вы должны посмотреть на свое собственное поведение, прежде чем просить измениться других», — говорит Яннес. Он будет первым, кто скажет вам, что он все еще учится. И в этом, как мы полагаем, кроется секрет его успеха.

После обеда мы направляемся на этаж руководства, где мы видим, как начинают формироваться некоторые из Обея старших лидеров. Мы сталкиваемся с Дэнни Вайнандом, главным инженером-конструктором, который работал под началом Яннеса, пока его не повысили в прошлом году до лидера собственного племени. Дэнни размышляет о распространении этого нового способа организации работы за пределы племен Яннеса в руководство и по всей остальной ING: «Ты теряешь терпение, желая ускорить их обучение, но потом понимаешь, что сам прошел через это и это заняло время. Рассказывать истории важно, но у них должно быть свое собственное обучение».

Вернувшись снова на этаж племени, мы встречаем Яна Рийхоффа, руководителя отдела. Мы хотели узнать о текущем подходе его отдела к решению проблем. На протяжении многих лет они экспериментировали с различными методами решения проблем, включая A3, Kata, Lean startup и другими, и, наконец, остановились на сочетании элементов, которые они нашли полезными, создав свой собственный подход. В ходе нашей сегодняшней экскурсии мы видели свидетельства многочисленных инициатив по решению проблем как непосредственно в процессе, так и визуализированные на стенах.

Их подход заключается в том, чтобы собрать правильных людей, которые обладают опытом и пониманием проблемы, чтобы тщательно изучить ее текущее состояние. Такая строгость окупается, поскольку команда получает знания, которые повышают вероятность выявления основной причины, а не только симптомов. С помощью этого обучения они формируют гипотезу о подходе к улучшению, включая и то, как и что измерять, чтобы узнать, дает ли эксперимент желаемые результаты. Если эксперимент успешен, они делают его частью стандартной работы, делятся полученными знаниями и продолжают следить за тем, чтобы улучшение было устойчивым. Они применяют этот подход к решению проблем на всех уровнях организации. Иногда проблема на уровне старшего руководителя анализируется и разбивается на более мелкие части, спускаясь до уровня отдела или отряда для анализа «на передовой» и контролируемых экспериментов с обратной отдачей полученных знаний. «Этот подход работает, — говорит нам Пол, когда мы снова встречаемся, — потому что он помогает людям принять изменения, позволяя внести свои собственные идеи, которые они затем могут проверить».

Среди этой красочной, творческой рабочей среды с философией «сделай это своим» идея стандартизации работы может показаться прямо противоположной или даже контрпродуктивной. В конечном итоге это работа со знанием. Рассмотрим понятие процесса (как что-то делается) и практики (выполнение чего-то, что требует знаний и суждений). Например, ритуалы Scrum — это процесс; действия по пониманию потребностей клиента и написанию кода — это практика. Таким образом, когда у команд есть стандартный способ работы, будь то

выпуск эффективного кода или проведение короткого совещания, следование этому стандарту экономит много времени и энергии. В ING стандарты работы устанавливаются не путем имитации способа работы, который предписан в книге или успешно используется другой компанией. Вместо этого команда ING экспериментирует с различными подходами и договаривается об одном лучшем способе выполнения работы. Эта установившаяся практика распространяется на все подобные команды. По мере изменения условий стандарт пересматривается и совершенствуется.

Мы догоняем Яннеса, когда он завершает свой день посещением комнаты Обея руководства, чтобы добавить несколько обновлений в заметки и посмотреть, какие обновления были сделаны другими. Мы спрашиваем о его мыслях по поводу пути, который они прошли. «Начальное понимание состояло в том, что наши команды не учились и не улучшались, — поделился он. — Мы не были способны вывести их на уровень, на котором они были бы постоянно обучающейся командой.

Я видел, что они боролись с проблемами, а у других команд были решения, а мы не могли собрать их вместе, чтобы они учились. Когда мы сами, как руководство, не были способны учиться, мы не могли помочь учиться командам. Мы должны были учиться сами, чтобы стать обучающейся командой. Мы [его управленческая команда] прошли через наше собственное обучение, а затем мы пошли к командам, чтобы помочь им научиться становиться обучающейся командой».

Затем мы спросили о его подходе к изменению культуры. «Прежде я никогда не обсуждал культуру, — сказал он. — Это была сложная тема, и я не знал, как изменить культуру устойчивым образом. Но я узнал, что когда вы меняете способ работы, вы меняете порядок, вы создаете другую культуру».

«Высшее руководство довольно нами, — добавляет он с широкой улыбкой, явно гордясь людьми в своих племенах. — Мы даем им скорость и качество. Иногда нам может потребоваться немного больше времени, чем другим, чтобы достичь зеленого статуса по задаче, но как только мы достигаем его, мы обычно остаемся зелеными, в то время как многие другие возвращаются к красному».

Трансформация лидерских, управленческих и командных практик

Руководители предприятий часто задают нам вопрос: как нам изменить нашу культуру?

Мы считаем, что лучше задать следующие вопросы: каким образом мы учимся, как учиться? Как мне учиться? Как я могу создать безопасную среду для того, чтобы другие учились? Как я могу учиться у них и вместе с ними? Как мы вместе устанавливаем новые модели поведения и новые

способы мышления, которые создают новые привычки, которые взращивают нашу новую культуру? И с чего мы начнем?

В ING Netherlands они начали с лидера, который задал себе эти вопросы. Затем он привлек хороших коучей, которым было поручено поставить перед каждым человеком (включая его самого) задачу подвергнуть сомнению предположения и попробовать новые модели поведения. Он собрал свою управленческую команду, сказав: «Давайте попробуем сделать это вместе. Даже если не сработает, мы узнаем что-то, что поможет нам стать лучше. Вы присоединитесь ко мне и посмотрите, чему мы можем научиться?»

Каждый квартал его управленческая команда собиралась вместе для получения новых знаний и в течение следующих месяцев применяла их на практике. То, что поначалу всем казалось неудобным, постепенно становилось немного легче и наконец превращалось в привычку — как раз к следующему учебному циклу. Они напряглись и, как только почувствовали себя комфортно, напряглись снова. Все это время вместе размышляли и при необходимости корректировали свои действия.

Мы напомним, что на одном из первых занятий в учебном лагере мы поставили перед членами управленческой команды задачу разработать простые стандартные рабочие процедуры для лидера: визуальное управление, регулярные стендапы и последовательный коучинг для членов своей команды, — заменив ими долгие совещания и «борьбу с пожарами», к которым они привыкли. Чтобы разработать этот новый способ работы, сначала им нужно было понять, на что они тратят свое время. Очевидно, это вызвало скептицизм и дискомфорт; тем не менее в течение нескольких недель каждый из них записывал и измерял, как он ежедневно проводит свое время. Они делились друг с другом тем, что узнали, и вместе вырабатывали новые способы работы.

Когда мы вернулись на следующий учебный лагерь три месяца спустя, Марк Найссен, один из менеджеров, приветствовал нас, сказав: «Я никогда больше не вернусь к старому способу работы!» Мало того, что внедрение базового стандарта работы лидера успешно помогло им повысить свою эффективность, им также удалось освободить 10% своего времени для работы над тем, что они сами выбирают.

Эта готовность экспериментировать с новыми способами мышления и работы привела к тому, где ING находится сегодня. Но важно признать, что нет никакого контрольного списка или плана. Вы не можете «выполнить» изменение культуры. Имплементационное мышление (попытка подражать специфическому поведению и практикам другой компании) по самой своей природе противоречит сущности производительной культуры.

В конце этой главы приведена таблица со множеством практик, описанных во время виртуального посещения ING. Отмеченные знаком (*) — это практики, которые, как показывают исследования, коррелируют

с высокой эффективностью. Мы надеемся, что в ходе будущих исследований будет изучен весь спектр перечисленных здесь практик. Эта таблица не должна использоваться в качестве контрольного списка, а скорее как выжимка или общее руководство для разработки собственных моделей поведения и практик (см. рис. 16.4).

	Командные практики	Управленческие практики	Лидерски
Культура	*Стимулирующая производительная культура	* Стимулирующая производительная культура	* Стимулирующая ная культура
	*Встраивайте качество в процесс, проводите постоянные измерения и мониторинг	* Фокусируйтесь на качестве, защищайте команду, чтобы обеспечить качество	* Фокусируйтесь н щайте команду, чт качество
	Фокусируйтесь на продвижении организационного обучения	Фокусируйтесь на продвижении организационного обучения	Фокусируйтесь на организационного
		*Предоставляйте командам время для улучшений и инноваций	* Предоставляйте к для улучшений и и
Организационная структура			* Выравнивайте, из и управляйте пото кроссфункционали онная структура п ценности)
		Формируйте небольшие кроссфункциональные команды широкого профиля; поддерживайте мостовые структуры, чтобы команды могли легко общаться и сотрудничать	Задействуйте и под кроссфункционали сокращения узких от экспертов, и фор ные сообщества
			Развивайте и подд ренних коучей и со инфраструктуру дл и масштабировани
Прямое обучение и согласование с ценностью	* Вовлекайте в процесс обучения клиентов, учитесь у них и проверяйте правильность идей вместе с ними (Gemba)	* Вовлекайте в процесс обучения клиентов, учитесь у них и проверяйте правильность идей вместе с ними (Gemba)	* Привлекайте и уч с клиентами, кома ми по поставкам и ресованными лица
	* Поймите и визуализируйте ценность для клиента, определите измеримые цели для качества	* Поймите и визуализируйте ценность для клиента, определите измеримые цели для качества	
	* Практикуйте творчество как часть общей работы	* Практикуйте творчество как часть общей работы, поощряйте членов команды использовать это время для обучения и инноваций	* Выделяйте бюдже творчества (в каче 20% в Google)
Развертывание стратегии	* Визуализируйте цели и задачи команды, добейтесь понимания того, как эти задачи продвигают стратегию предприятия	Помогите командам поставить и визуализировать цели и задачи, поймите и сообщите, как эти цели продвигают стратегию предприятия (catchball)	Практикуйте разве гии, визуализируй жайшие задачи, че об этом менеджер им выдвигать прав и инициативы
	* Проводите активный мониторинг и визуализируйте показатели эффективности по целям/задачам	* Проводите активный мониторинг и визуализируйте показатели эффективности по целям/задачам	* Проводите активн и визуализируйте тивности по целям
			Устраните ненужны контроля, вместо э в качество процесс и возможности ком которые сообщили процесса утвержд

	Командные практики	Управленческие практики	Лидерские практик
Улучшение прохождения потока через анализ и организованное решение проблем	Визуализируйте и анализируйте рабочий процесс, выявляйте препятствия (дорожная карта и анализ процесса и потока создания ценности); * поймите связь между работой, которую делают команды, и ее положительным влиянием на клиентов	Визуализируйте и анализируйте рабочий процесс, выявляйте препятствия (дорожная карта и анализ процесса и потока создания ценности), помогите командам понять, как они поддерживают больший поток создания ценности	Визуализируйте и анализир щие потоки создания ценнос хитектура предприятия), выя системные препятствия для расставляйте приоритеты и живайте дорожную карту и потоков более низкого уровн
	Расставьте приоритеты по задачам устранения препятствий для ценности и опыта клиента, а также по целям и задачам команды	Расставьте приоритеты по задачам устранения препятствий для ценности и опыта клиента, а также по целям и задачам команды	Отдайте приоритет задачам темным препятствиям для по
	Применяйте стандартную процедуру решения проблем к приоритетным проблемам, анализируйте, чтобы определить первопричины	Применяйте стандартную процедуру решения проблем к приоритетным проблемам, анализируйте, чтобы определить первопричины	Применяйте стандартную пр ру решения проблем к комп системным вопросам для оп ния целей и задач стратегиче улучшения (развертывание тегии), применяйте обучение обновления стандартов раб
	Передавайте наверх решение кроссфункциональных и системных проблем	Координируйте решение кросс-функциональных задач, решайте или передавайте наверх решение системных проблем	Спускайте вниз приоритетны чи, требующие решения на соответствующих заинтересо сторон, посредством catchb
	Формируйте гипотезы о первопричинах, разрабатывайте и проводите контролируемые эксперименты, измеряйте результаты, сообщайте о полученных знаниях, повторяйте при необходимости, внедряйте улучшения	Формируйте гипотезы о первопричинах, разрабатывайте и проводите контролируемые эксперименты, измеряйте результаты, сообщайте о полученных знаниях, повторяйте при необходимости, внедряйте улучшения	Учитесь на общеорганизаци циклах PDCA и повторяйте ц обучения/улучшения
Способ работы, ритм и стандартные практики	* Визуализируйте, измеряйте и проводите мониторинг рабочего процесса, следите за отклонениями, адекватно реагируйте на отклонения	* Визуализируйте, измеряйте и проводите мониторинг рабочего процесса, следите за отклонениями, адекватно реагируйте на отклонения	* Визуализируйте, измеряйте и проводите мониторинг раб процесса, следите за отклон адекватно реагируйте на отк
	* Разбейте спрос на небольшие элементы (MVP) и выпускайте продукты регулярно и часто		
	* Визуализируйте спрос, НЗП и сделанное (канбан)	* Визуализируйте спрос, НЗП и сделанное (канбан)	* Визуализируйте спрос, НЗ ланное (канбан)
	* Минимизируйте и визуализируйте НЗП	* Минимизируйте и визуализируйте НЗП	* Минимизируйте и визуали НЗП
	Расставьте приоритет целей и задач по удовлетворению спроса	Расставьте приоритет целей и задач по удовлетворению спроса	Расставьте приоритет целей по удовлетворению спроса
	Разрабатывайте и практикуйте стандарты работы команды (ритм и повседневные процедуры)	Разрабатывайте и практикуйте стандарты работы лидера (ритм и повседневные процедуры)	Разрабатывайте и практикуй стандарты работы лидера (р и повседневные процедуры)
	Проводите ежедневные стендапы со стандартной процедурой, передавайте наверх задачи по устранению препятствий по мере необходимости (catchball)	Проводите ежедневные стендапы со стандартной процедурой, передавайте наверх задачи по устранению препятствий по мере необходимости (catchball)	Проводите ежедневные стенд со стандартной процедурой передавайте наверх задачи по устранению препятствий необходимости (catchball)
	Поддерживайте обучение среди членов команды и коллег	Обучайте членов команды, поддерживайте командное обучение	Обучайте менеджеров, найд наставника самому себе
	Проводите регулярные встречи по ретроспективам (работа и способ работы)	Проводите регулярные встречи по ретроспективам (работа и способ работы)	Проводите регулярные встре по ретроспективам (работа работы)

Как вы видели в ходе нашей виртуальной экскурсии по ING, высокоэффективная культура — это гораздо больше, чем просто применение инструментов, внедрение набора взаимосвязанных практик, копирование поведения других успешных организаций или реализация предписанной, разработанной экспертами структуры. Это развитие посредством экспериментов и обучения новому способу совместной работы на основании фактических данных, который является ситуационно и культурно приемлемым для каждой организации.

Когда вы начинаете свой собственный путь к созданию обучающейся организации, важно принять и поддерживать правильное мышление. Ниже приведены некоторые наши предложения, основанные на нашем собственном опыте оказания помощи предприятиям, желающим развиваться в направлении высокоэффективной производительной культуры.

- Развивайте и поддерживайте правильное мышление. Речь идет об обучении и о том, как создать среду для совместного организационного обучения, а не просто о выполнении практик и, конечно же, не о применении инструментов.
- Сделайте это своим. Это означает три вещи.
- Не пытайтесь копировать методы и практики других предприятий или внедрять разработанную экспертами модель. Изучайте и учитесь у них, но затем экспериментируйте и внедряйте то, что работает для вас и вашей культуры.
- Не заключайте контракт с крупной консалтинговой фирмой, чтобы она быстро преобразовала вашу организацию или внедрила новые методологии или практики для вас. Ваши команды будут чувствовать, что эти методологии (Lean, Agile и т.п.) навязываются им сверху. В то время как ваши текущие процессы могут временно улучшиться, ваши команды не будут развивать в себе уверенность или способность поддерживать, продолжать улучшать или внедрять и развивать новые процессы и модели поведения самостоятельно.
- Развивайте своих собственных коучей. Сначала вам может потребоваться нанять внешний коучинг, чтобы заложить прочную основу, но в конечном итоге вы должны сами проводить свои изменения. Внутренний коучинг является ключевым рычагом для поддержания и масштабирования.
- Вам тоже нужно изменить свой способ организации работы. Независимо от того, являетесь ли вы старшим руководителем, менеджером или членом команды, подавайте пример. Производительная культура начинается с демонстрации новых моделей поведения, а не с их делегирования.
- Практикуйте дисциплину. Для управленческой команды Яннеса было нелегко записывать и размышлять, на что они тратят свое время, или пробовать новые вещи, которые им изначально не

нравились, на глазах своих подчиненных. Перемены требуют дисциплины и мужества.
- Практикуйте терпение. На то, чтобы утвердился ваш нынешний способ работы, ушли десятилетия. Потребуется время, чтобы изменить образ действия мышления, пока они не станут новыми привычками и в конечном итоге вашей новой культурой.
- Практикуйте новые практики. Вы просто должны пробовать: учиться, добиваться успеха, терпеть неудачу, учиться, корректировать, повторять. Ритм и рутина, ритм и рутина, ритм и рутина…

По мере того, как вы осваиваете новый способ руководства и работы, вы и те, кого вы берете с собой в это путешествие, будете исследовать, расширять мышление, делать ошибки, исправлять их, учиться, расти и продолжать учиться. Вы откроете для себя лучшие и более быстрые способы взаимодействия, обучения и адаптации к изменяющимся условиям. Таким образом вы повысите качество и скорость во всем, что вы делаете. Вы будете растить своих собственных лидеров, внедрять инновации и превосходить своих конкурентов. Вы будете быстрее и эффективнее повышать ценность для клиентов и предприятия. Как показывает исследование, вы «окажете ощутимое влияние на прибыльность, производительность и долю рынка организации. Они также оказывают влияние на удовлетворенность клиентов, эффективность и способность достигать организационных целей».

Мы желаем вам всего наилучшего в вашем обучающем путешествии!

Стив и Карен

Заключение

За последние несколько лет проведения опросов среди технических профессионалов и написания отчетов о состоянии DevOps с командой Puppet мы многое узнали о том, что отличает высокоэффективные команды и организации. Этот путь включал исследование технологических трансформаций, публикацию наших результатов в экспертных обзорах и работу с нашими коллегами и партнерами, которые оценивают и преобразуют свои собственные организации. На протяжении всего этого путешествия мы сделали много прорывных открытий о взаимосвязи между эффективностью доставки, техническими практиками, культурными нормами и организационной эффективностью.

Во всех наших исследованиях одна вещь оказалась неизменно верной: поскольку почти каждая компания полагается на программное обеспечение, эффективность его доставки имеет решающее значение для любой организации, занимающейся сегодня бизнесом. А

эффективность доставки программного обеспечения зависит от многих факторов, включая лидерство, инструменты, автоматизацию и культуру непрерывного обучения и совершенствования.

Эта книга — компиляция открытий, с которыми мы столкнулись во время этого путешествия. В Части I мы представили то, что обнаружили в ходе исследования. Она начинается с обсуждения, почему эффективность доставки ПО имеет значение и как она влияет на организационные показатели эффективности, такие как прибыльность, производительность и доля рынка, а также некоммерческие показатели — продуктивность, работоспособность, удовлетворенность клиентов и достижение целей миссии компании. Таким образом, способность доставлять качественное программное обеспечение стабильно и в высоком темпе является ключевым двигателем ценности и отличительным фактором для всех организаций, независимо от размера или отраслевой вертикали.

В Части II мы кратко изложили научные основы исследования и пролили некоторый свет на принятые нами проектные решения, а также используемые нами методы анализа. Это обеспечивает базу для результатов, которые мы обсуждаем в основной части текста.

Мы также определили ключевые возможности, которые способствуют эффективной доставке ПО статистически значимыми способами. Мы надеемся, что обсуждение этих практик с примерами поможет вам улучшить вашу собственную эффективность.

В Части III мы завершаем обсуждение вопросов управления организационными изменениями. Чтобы представить этот материал, мы обратились к коллегам Стиву и Карен Уитли Белл. В их главе представлен взгляд на то, что собой представляет следование возможностям и практикам, описанным в этой книге, и что оно может обеспечить для инновационных организаций. Вы можете начать свою собственную технологическую трансформацию со всем тем, что мы узнали в ходе нашего исследования, — трансформацию, которую многие другие уже с успехом смогли реализовать в своих собственных командах и организациях.

Мы надеемся, что эта книга поможет вам определить области, в которых вы можете улучшить собственные технологии и бизнес-процессы, культуру работы и циклы совершенствования. Помните: вы не можете купить или скопировать высокую эффективность. Вам нужно будет развивать свои собственные возможности по мере следования по пути, который соответствует вашим конкретным условиям и целям. Это потребует постоянных усилий, инвестиций, концентрации и времени. Однако вывод из нашего исследования однозначен. Результаты того стоят. Мы желаем вам всего наилучшего на вашем пути совершенствования и с нетерпением ждем ваших историй.

Приложение А

Возможности для управления улучшениями

Наше исследование выявило 24 ключевые возможности, которые статистически значимым образом влияют на повышение эффективности доставки программного обеспечения. В нашей книге подробно описаны эти открытия. Данное приложение содержит удобный список этих возможностей — каждая с указанием на главу, которая подробно ее рассматривает (также см. рис. А.1).

Мы разделили эти возможности на пять категорий:

- непрерывная доставка;
- архитектура;
- продукт и процесс;
- бережливое управление и мониторинг;
- культура.

В каждой категории возможности представлены в произвольном порядке.

Возможности непрерывной доставки

1. Используйте контроль версий для всех производственных артефактов. Контроль версий — это использование системы управления версиями, такой как GitHub или Subversion, для всех производственных артефактов, включая код приложения, конфигурации приложений, системные конфигурации и скрипты для автоматизации сборки и настройки среды. См. Главу 4.
2. Автоматизируйте ваш процесс развертывания. Автоматизация развертывания — это степень, с которой развертывания полностью автоматизированы и не требуют ручного вмешательства. См. Главу 4.
3. Внедряйте непрерывную интеграцию **(НИ).** НИ является первым шагом на пути к непрерывной доставке. Это практика разработки, в которой код регулярно проверяется и каждая проверка запускает набор быстрых тестов для обнаружения серьезных сбоев, которые разработчики немедленно исправляют. Процесс НИ создает канонические сборки и пакеты, которые в конечном итоге развертываются и выпускаются. См. Главу 4.
4. Используйте магистральные методы разработки. Было показано, что магистральная разработка является прогностическим фактором высокой эффективности в разработке и доставке программного обеспечения. Для нее характерны менее трех активных ветвей в репозитории кода, ветви и вилки с очень коротким сроком жизни

(например, менее одного дня) до слияния в магистраль и крайне редкие или отсутствующие периоды «кодовой блокировки» у команд приложений, когда никто не может проверить код или выполнить запрос на вытягивание из-за конфликтов слияния, заморозки кода или фаз стабилизации. См. Главу 4.

5. Внедряйте автоматизацию тестирования. Это практика, при которой тесты программного обеспечения выполняются автоматически (а не вручную) непрерывно на протяжении всего процесса разработки. Эффективные наборы тестов надежны, то есть тесты находят реальные сбои и пропускают только код, готовый к выпуску. Обратите внимание, что разработчики должны нести основную ответственность за создание и обслуживание наборов автоматических тестов. См. Главу 4.
6. Поддерживайте управление тестовыми данными. Они требуют тщательного обслуживания. Управление тестовыми данными становится все более важной частью автоматизированного тестирования. Эффективные практики включают наличие достаточных данных для запуска набора тестов, возможность получения необходимых данных по требованию, возможность создания условий для тестовых данных в конвейере и данные, не ограничивающие количество тестов, которые вы можете запустить. Однако мы предупреждаем, что команды должны по возможности минимизировать объем тестовых данных, необходимых для выполнения автоматических тестов. См. Главу 4.
7. «Сдвигайтесь влево» по безопасности. Интеграция безопасности в этапы проектирования и тестирования процесса разработки программного обеспечения является ключом к повышению эффективности ИТ. Это включает в себя проверку безопасности приложений, подключение команды по безопасности (infosec) в процесс разработки и демонстрации приложений, использование предварительно утвержденных библиотек и пакетов безопасности, а также тестирование функций безопасности в составе набора автоматических тестов. См. Главу 4.
8. Внедряйте непрерывную доставку (НД). НД — это практика разработки, когда программное обеспечение находится в развертываемом состоянии на протяжении всего жизненного цикла и поддержание ПО в развертываемом состоянии для команды приоритетнее, чем работа над новыми функциями. Быстрая обратная связь по качеству и развертываемости системы доступна всем членам команды, и когда они получают отчеты о том, что система не может быть развернута, оперативно исправляют ошибки. Наконец, система может быть развернута в производство или конечным пользователям в любое время по требованию. См. Главу 4.

Возможности архитектуры

9. Используйте слабосвязанную архитектуру. Это влияет на степень, с которой команда может тестировать и развертывать свои приложения по требованию, без согласования с другими службами. Наличие слабосвязанной архитектуры позволяет вашим командам работать независимо, не полагаясь на поддержку и услуги других команд, что, в свою очередь, позволяет им работать быстро и создавать ценность для организации. См. Главу 5.

10. Создавайте архитектуру для уполномоченных команд. Наше исследование показывает, что команды, которые могут сами выбирать, какие инструменты использовать, лучше справляются с непрерывной доставкой и, в свою очередь, улучшают эффективность разработки и доставки программного обеспечения. Никто лучше специалистов-практиков не знает, что им нужно, чтобы быть эффективными. См. Главу 5. (Аналог этого подхода в сфере управления продуктом можно найти в Главе 8.)

Возможности продукта и процесса

11. Собирайте и используйте обратную связь от клиентов. Наше исследование обнаружило, что для эффективности доставки программного обеспечения важно, чтобы организации активно и регулярно получали обратную связь от клиентов и включали ее в разработку своих продуктов. См. Главу 8.

12. Сделайте рабочий процесс видимым через поток создания ценности. Команды должны обладать хорошим пониманием и видимостью рабочего процесса от бизнеса до клиентов, включая статусы продуктов и функций. Наше исследование показало, что это положительно влияет на эффективность ИТ. См. Главу 8.

13. Работайте в небольших партиях. Команды должны «нарезать» работу на небольшие куски, которые могут быть завершены в течение недели или меньше. Суть состоит в том, чтобы работа была разделена на небольшие функции, которые можно быстро разработать, вместо того чтобы разрабатывать сложные функции на ветвях и выпускать их нечасто. Эта идея применима как на уровне функции, так и на уровне продукта. (MVP — это прототип продукта с достаточным количеством функций для достоверного изучения продукта и его бизнес-модели.) Работа в небольших партиях позволяет сократить время выполнения заказа и ускорить цикл обратной связи. См. Главу 8.

14. Поощряйте и создавайте условия для командных экспериментов. Командное экспериментирование — это возможность для разработчиков опробовать новые идеи и создавать и обновлять спецификации в процессе разработки, не требуя одобрения извне команды, что позволяет им быстро внедрять инновации и создавать ценность. Это особенно эффективно в сочетании с работой в небольших партиях, включая обратную связь с клиентами и создание видимого

процесса работы. См. Главу 8. (Технический аналог этого можно найти в Главе 4.)

Возможности бережливого управления и мониторинга

15. Обеспечьте облегченный процесс утверждения изменений. Наше исследование показывает, что облегченный процесс утверждения изменений, основанный на внутренней экспертизе (парном программировании или внутрикомандной проверке кода), обеспечивает более высокую эффективность ИТ, чем использование внешних органов по утверждению изменений (CABs). См. Главу 7.

16. Проводите мониторинг всех приложений и инфраструктуры для принятия бизнес-решений. Используйте данные из инструментов мониторинга приложений и инфраструктуры для принятия мер и бизнес-решений. Это лучше, чем искать виноватых, когда что-то идет не так. См. Главу 7.

17. Заранее проверяйте работоспособность системы. Мониторинг работоспособности системы с помощью предупреждений о пороговых значениях и скорости изменения помогает командам заблаговременно обнаруживать и устранять проблемы. См. Главу 13.

18. Улучшайте процессы и управляйте ограничением незавершенного производства (НЗП). Использование ограничений НЗП для управления процессом работы хорошо известно в «бережливом» сообществе. При эффективном использовании это приводит к улучшению процесса, увеличивает производительность и делает ограничения видимыми в системе. См. Главу 7.

19. Визуализируйте работу, чтобы контролировать качество и общаться всей командой. Визуальные дисплеи, такие как приборные панели или внутренние веб-сайты, используемые для мониторинга качества и НЗП, способствуют повышению эффективности доставки программного обеспечения. См. Главу 7.

Культурные возможности

20. Поддерживайте производительную культуру (по Веструму). Эта оценка организационной культуры основана на типологии, разработанной Роном Веструмом, социологом, который изучал критические с точки зрения безопасности сложные системы в области авиации и здравоохранения. Наше исследование показало, что этот показатель культуры предсказывает эффективность ИТ, организационную эффективность и снижение выгорания. Его отличительными чертами являются хороший поток информации, высокий

уровень сотрудничества и доверия, наведение мостов между командами и сознательное изучение ошибок. См. Главу 3.

21. Поощряйте и поддерживайте обучение. Считается ли обучение в вашей культуре необходимым условием для постоянного прогресса? Рассматривается ли обучение как затраты или как инвестиции? Это показатель обучающей культуры организации. См. Главу 10.

22. Поддерживайте и оказывайте содействие сотрудничеству между командами. Это отражает, насколько хорошо команды, которые традиционно были изолированы, взаимодействуют в области разработки, эксплуатации и информационной безопасности. См. Главы 3 и 5.

23. Обеспечьте ресурсы и инструменты, которые делают работу значимой. Этот показатель удовлетворенности работой состоит в выполнении сложных и важных задач, требующих от человека приложения его знаний и навыков. Речь также идет о получении людьми инструментов и ресурсов, необходимых для качественного выполнения работы. См. Главу 10.

24. Поддерживайте или воплощайте трансформационное лидерство. Трансформационное лидерство поддерживает и усиливает техническую и технологическую работу, которая так важна в DevOps. Оно состоит из пяти факторов: видение, интеллектуальная стимуляция, вдохновляющее общение, поддерживающее лидерство и личное признание. См. Главу 11.


Трансформационное лидерство
Видение
Вдохновляющее общение
Интеллектуальная стимуляция
Поддерживающее лидерство
Личное признание
Бережливая разработка продукта
Работа небольшими партиями
Командные эксперименты
Сбор и применение клиентских отзывов
Визуальный менеджмент
Бережливое управление
Ограничение незавершенного производства (НЗП)
Визуализации
Визуализация хода работ
Упрощенное утверждение изменений
Автоматизация тестирования
Автоматизация развертывания
Магистральная разработка
«Сдвиг влево» по безопасности
Слабосвязанная архитектура
Уполномоченные команды
Непрерывная интеграция
Контроль версий
Управление тестовыми данными
Мониторинг
Упреждающие уведомления

Приложение B

Статистика

Хотите знать, что мы обнаружили с точки зрения статистики? Ниже мы перечислим все это, организованное по категориям.

В качестве напоминания.

Корреляция отражает то, насколько согласованно (или нет) изменяются две переменные, но она не говорит нам, предсказывает или вызывает ли изменение одной переменной изменение другой переменной. Две переменные, изменяющиеся согласованно, всегда могут быть связаны с третьей переменной или — иногда — просто случайностью.

Предсказание говорит о влиянии одной конструкции на другую. В частности, мы использовали дедуктивно-прогностический анализ, один из наиболее распространенных видов анализа, применяемых в технологических и бизнес-исследованиях. Он помогает нам понять влияние HR-политики, организационного поведения и мотивации, а также то, как технологии влияют на такие результаты, как удовлетворенность пользователей, эффективность команды и эффективность организации. Дедуктивное проектирование применяется тогда, когда чисто экспериментальное проектирование невозможно, а предпочтение отдается полевым экспериментам, например, в бизнесе, когда сбор данных осуществляется в сложных организациях, а не в стерильных лабораторных условиях, и компании не будут жертвовать своей прибылью, чтобы вписаться в контрольные группы, определенные исследовательской командой. Методы анализа, используемые для проверки предсказаний, включают простую линейную регрессию и регрессию частичных наименьших квадратов, описанную в Приложении C.

Организационная эффективность

- Группа респондентов с высокими показателями в два раза чаще превышает показатели организационной эффективности, чем группа с низкими показателями, а именно: прибыльность, продуктивность, долю рынка, количество клиентов.
- Группа респондентов с высокими показателями в два раза чаще превышает некоммерческие показатели эффективности, чем группа с низкими показателями: количество продуктов/услуг, операционную эффективность, удовлетворенность клиентов, качество продуктов/услуг, достижение организационных целей и миссии.

- В ходе контрольного опроса по итогам первоначальной работы по сбору данных за 2014 год мы собрали данные о биржевых котировках и провели дополнительный анализ ответов чуть более 1000 респондентов из 355 компаний, которые добровольно предложили свою организацию для участия в опросе. Для сотрудников публичных компаний мы обнаружили следующее (этот анализ не был воспроизведен в последующие годы, поскольку наш набор данных был недостаточно большим):
- рост рыночной капитализации за три года у группы респондентов с высокими показателями оказался на 50% выше по сравнению с группой с низкими показателями.

Эффективность доставки ПО

- Четыре показателя эффективности доставки программного обеспечения (частота развертывания, время выполнения, среднее время восстановления, процент сбоев изменений) являются хорошими классификаторами для профиля эффективности доставки ПО. Группы, которые мы определили по показателям эффективности как высокие, средние и низкие, все значительно отличаются друг от друга по всем четырем показателям каждый год.
- Наш анализ участников с высокими, средними и низкими показателями свидетельствует о том, что нет никаких компромиссов между улучшением эффективности и достижением более высоких уровней скорости и стабильности: они движутся в тандеме.
- Эффективность доставки программного обеспечения предсказывает организационную и некоммерческую эффективность.
- Конструкция эффективности доставки ПО представляет собой комбинацию трех показателей: времени выполнения, частоты выпуска и среднего времени восстановления (MTTR — mean time to repair). Частота сбоев изменений не включена в конструкцию, хотя она сильно коррелирует с конструкцией.
- Частота развертывания сильно коррелирует с непрерывной доставкой и всесторонним использованием контроля версий.

Время выполнения сильно коррелирует с контролем версий и автоматизированным тестированием.

- MTTR сильно коррелирует с контролем версий и мониторингом.
- Эффективность доставки программного обеспечения коррелирует с организационными инвестициями в DevOps.
- Эффективность доставки программного обеспечения отрицательно коррелирует с «болью развертывания». Чем болезненнее

развертывание кода, тем хуже эффективность доставки ПО и культура.

Качество

- Незапланированная работа и доработки:
- респонденты с высокими показателями сообщили, что тратят 49% своего времени на новую работу и 21% на незапланированную работу или доработки;
- респонденты с низкими показателями тратят 38% своего времени на новую работу и 27% на незапланированную работу или доработки;
- в наших данных по доработкам есть доказательство J-кривой. Респонденты со средними показателями тратят больше времени на незапланированную доработку, чем респонденты с низкими показателями, причем 32% их времени тратится на незапланированную работу или доработки.
- Ручная работа:
- респонденты с высокими показателями сообщают о самом низком объеме ручной работы во всех практиках (управление конфигурацией, тестирование, развертывание, процесс утверждения изменений) на статистически значимых уровнях;
- опять же, есть доказательство J-кривой. Респонденты со средними показателями выполняют больше ручной работы, чем респонденты с низкими показателями, когда речь заходит о процессах развертывания и утверждения изменений, и эти различия статистически значимы;
- см. таблицу B.1.

Таблица B.1

Процент ручной работы

Ручная работа	Респонденты с высокими показателями	Респонденты со средними показателями	Респонденты с низкими показателями
Управление конфигурацией	28%	47%*	46%*
Тестирование	35%	51%*	49%*
Развертывание кода	26%	47%	43%
Процесс утверждения изменений	48%	67%	59%

* Различия между респондентами со средними и низкими показателями для управления конфигурацией и тестирования не являются статистически значимыми.

Выгорание и «боль развертывания»

- «Боль развертывания» отрицательно коррелирует с эффективностью доставки программного обеспечения и организационной культурой Веструма.
- Пять наиболее коррелирующих с выгоранием факторов — это организационная культура Веструма (в отрицательном значении), лидеры (в отрицательном значении), организационные инвестиции (в отрицательном значении), организационная эффективность (в отрицательном значении) и «боль развертывания» (в положительном значении).

Технические возможности

(Возможности архитектуры находятся в их собственном разделе ниже.)

- Магистральная разработка:
- респонденты с высокими показателями имеют самый короткий срок слияния в магистраль и время жизни ветви — обычно несколько часов или день;
- респонденты с низкими показателями имеют самый долгий срок слияния в магистраль и время жизни ветви обычно несколько дней или недель.
- Технические практики предсказывают непрерывную доставку, организационную культуру Веструма, идентификацию с компанией, удовлетворенность работой, эффективность доставки программного обеспечения, меньше выгорания, меньше «боли развертывания» и меньше времени, затраченного на доработки.
- Респонденты с высокими показателями тратят на 50% меньше времени на устранение проблем безопасности, чем респонденты с низкими показателями.

Возможности архитектуры

- Не было никакой корреляции между конкретным типом системы (например, системой взаимодействия или системой записи) и эффективностью доставки программного обеспечения.
- Респонденты с низкими показателями чаще сообщали, что создаваемое ими программное обеспечение или набор сервисов, с которыми они должны взаимодействовать, — это «заказное программное обеспечение, разработанное другой компанией (например, партнером на аутсорсинге)».
- Респонденты с низкими показателями с более высокой вероятностью будут работать на универсальных системах (mainframe systems).
- Необходимость интеграции с системами мейнфреймов не является статистически значимым показателем эффективности.

- Респонденты со средними и низкими показателями не имеют существенной корреляции между типом системы и эффективностью доставки программного обеспечения.
- Слабосвязанная, хорошо инкапсулированная архитектура повышает эффективность ИТ. В наборе данных 2017 года вклад этого фактора в непрерывную доставку оказался самым большим.
- Среди тех, кто развертывает код хотя бы раз в день, по мере увеличения количества разработчиков в команде мы обнаружили, что:
- респонденты с низкими показателями развертывают код с уменьшающейся частотой;
- респонденты со средними показателями развертывают код с постоянной частотой;
- респонденты с высокими показателями развертывают код со значительно возрастающей частотой.
- Высокоэффективные команды с большей вероятностью положительно отреагировали на следующие утверждения:
- мы можем сделать большую часть нашего тестирования, не прибегая к интегрированной среде;
- мы можем и делаем развертывание/выпуск наших приложений независимо от других приложений/служб, от которых они зависят;
- это заказное ПО, которое использует архитектуру микросервисов.
- Мы не обнаружили существенных различий в зависимости от того, какой тип архитектуры команды строили или интегрировали.

Возможности бережливого управления

Возможности бережливого управления предсказывают организационную культуру Веструма, удовлетворенность работой, эффективность доставки программного обеспечения и меньшее выгорание.

Утверждение изменений:

— консультативные советы по изменениям отрицательно коррелируют с эффективностью доставки программного обеспечения;

— утверждение только изменений с высоким риском не коррелировало с эффективностью доставки программного обеспечения;

— команды, которые сообщили об отсутствии процесса утверждения или использовали внутреннюю экспертизу, достигли более высокой эффективности доставки программного обеспечения;

— облегченный процесс утверждения изменений предсказывает эффективность доставки программного обеспечения.

Возможности бережливого управления продуктом

- Способность применять экспериментальный подход к разработке продукта в значительной степени коррелирует с техническими практиками, которые способствуют непрерывной доставке.
- Возможности бережливой разработки продукта предсказывают организационную культуру Веструма, эффективность доставки программного обеспечения, организационную эффективность и меньшее выгорание.

Возможности организационной культуры

- Данные показатели сильно коррелируют с культурой:
- организационные инвестиции в DevOps;
- опыт и эффективность работы лидеров команд;
- возможности непрерывной доставки;
- способность команд разработки, эксплуатации и информационной безопасности достигать взаимовыгодных результатов;
- организационная эффективность;
- «боль развертывания»;
- практики бережливого управления.
- Организационная культура Веструма предсказывает эффективность доставки программного обеспечения, организационную эффективность и удовлетворенность работой.
- Организационная культура Веструма отрицательно коррелирует с «болью развертывания». Чем болезненнее развертывание кода, тем хуже культура.

Идентификация, чистый индекс лояльности сотрудников (eNPS) и удовлетворенность работой

- Идентификация с компанией предсказывает организационную эффективность.
- У респондентов с высокими показателями выше лояльность сотрудников по результатам измерений eNPS. Сотрудники высокоэффективных организаций в 2,2 раза чаще рекомендовали свою организацию как отличное место для работы.
- eNPS значительно коррелирует со:
- степенью, в которой организация собирает отзывы клиентов и использует их в разработке продуктов и функций;
- способностью команд визуализировать и понимать поток продуктов или функций на протяжении всего процесса разработки вплоть до клиента;
- степенью, в которой сотрудники идентифицируют себя с ценностями и целями своей организации, и усилиями, которые они готовы приложить, чтобы сделать организацию успешной.

- Сотрудники высокоэффективных команд в 2,2 раза чаще рекомендуют свою организацию как отличное место для работы.
- Сотрудники высокоэффективных команд в 1,8 раза чаще рекомендуют свою команду как отличное место для работы.
- Удовлетворенность работой предсказывает организационную эффективность.

Лидерство

- Мы наблюдали значительные различия в лидерских характеристиках среди высоко-, средне- и низкоэффективных команд.
- Высокоэффективные команды сообщили, что у них есть лидеры с самым сильным поведением во всех измерениях: видение, вдохновляющее общение, интеллектуальная стимуляция, поддерживающее лидерство и личное признание.
- Низкоэффективные команды сообщили о самых низких уровнях всех пяти лидерских характеристик.
- Все эти различия были отмечены на статистически значимых уровнях.
- Характеристики трансформационного лидерства сильно коррелируют с эффективностью доставки программного обеспечения.
- Трансформационное лидерство в значительной степени коррелирует с eNPS.
- Команды с лучшими 10% заявленных характеристик трансформационного лидерства были одинаково или даже менее склонны к высоким показателям по сравнению со всей популяцией команд, представленных в результатах опроса.
- Лидерство является прогностическим фактором возможностей бережливой разработки продукта (работа в небольших партиях, командные эксперименты, сбор и внедрение отзывов клиентов) и технических практик (автоматизация тестирования, автоматизация развертывания, разработка на основе магистрали, «сдвиг влево» по безопасности, слабосвязанная архитектура, уполномоченные команды, непрерывная интеграция).

Разнообразие

- Из общего числа респондентов идентифицировали себя как женщин 5% в 2015 году, 6% в 2016 году и 6,5% в 2017 году.
- 33% наших респондентов сообщили, что работают в командах без женщин.
- 56% наших респондентов сообщили, что работают в командах, в которых менее 10% женщин.

- 81% наших респондентов сообщили, что работают в командах, в которых менее 25% женщин.
- Пол:
- 91% мужчины;
- 6% женщины;
- 3% другое.
- Недостаточно представленные группы:
- 77% ответили: «Нет, я не отношу себя к недостаточно представленной группе»;
- 12% ответили: «Да, я отношу себя к недостаточно представленной группе»;
- 11% ответили, что они предпочли бы не отвечать или не определились.

Другое

- Инвестиции в DevOps сильно коррелировали с эффективностью доставки ПО.
- Процент людей, сообщивших о работе в командах DevOps, вырос за последние четыре года:
- 16% в 2014 году;
- 19% в 2015 году;
- 22% в 2016 году;
- 27% в 2017 году.
- DevOps происходит во всех операционных системах. Впервые мы начали спрашивать об этом в 2015 году.
- 78% респондентов широко развернуты на 1–4 операционных системах, наиболее популярные из которых Enterprise Linux, Windows 2012, Windows 2008, Debian/Ubuntu Linux.
- На рис. B.1 показана «фирмографика» из данных за 2017 год. Отметим, что высоко-, средне- и низкоэффективные респонденты представлены во всех группах. То есть крупные предприятия есть в высоко-, средне- и низкоэффективных группах. Мы также видим стартапы в высоко-, средне- и низкоэффективных группах. Строго регулируемые отрасли (в том числе финансы, здравоохранение, телекоммуникации и т.д.) также встречаются в высоко-, средне- и низкоэффективных группах. Важно не то, в какой отрасли вы работаете или насколько велика ваша компания; даже крупные, строго регулируемые организации способны разрабатывать и доставлять программное обеспечение с высокой эффективностью, а затем использовать эти возможности в обеспечении ценности для своих клиентов и своей организации.

Рис. B.1. **«Фирмографика»: размер организации, отрасль,**

Приложение C

Статистические методы, использованные в нашем исследовании

Это приложение представляет собой краткое изложение статистических методов, которые мы использовали в нашем исследовании. Оно служит в качестве справочной информации, а не подробного статистического текста. Мы включили указатели на соответствующие академические ссылки там, где это уместно. Приложение примерно следует нашему пути через проектирование и анализ исследования.

Подготовка опроса

Как только мы определились с конструкциями и гипотезами, которые мы хотим проверять каждый год, мы начинаем процесс исследования с разработки инструмента опроса[49].

По возможности используются ранее проверенные элементы. Например, организационная эффективность (Widener, 2007) и некоммерческая эффективность (Каваллуццо и Иттнер, 2004). Когда мы создаем свои собственные измерения, инструмент опроса разрабатывается в соответствии с общепринятыми процедурами, адаптированными из работы Дона Дилмана (Dillman, 1978).

Сбор данных

Вооружившись нашим исследовательским проектом и вопросами опроса, мы приступили к сбору данных.

Мы собирали данные, используя выборку методом снежного кома, то есть технику неслучайной выборки. Более подробно о том, почему она является подходящей техникой, как мы собрали нашу выборку и какие стратегии мы использовали для противодействия ограничениям этого метода, мы рассказываем в Главе 15.

Тесты на смещения

Как только мы получаем наши данные, мы начинаем тестирование на смещение.

- Тесты по критерию хи-квадрат. Тест на различия. Он используется для проверки наличия существенных различий в переменных,

которые могут принимать только безусловные значения (например, пол).

- Тесты по t-критерию. Тест на различия. Он используется для проверки наличия существенных различий в переменных, которые могут принимать значения масштаба (например, значения по шкале Ликерта). Мы использовали его для проверки различий между ранними и поздними респондентами.
- Смещение общего метода (CMB — common method bias) или дисперсия общего метода (CMV — common method variance). Включает в себя проведение двух тестов:
- однофакторный тест Хармана (Подсакофф и Далтон, 1987). Он проверяет, не оказывает ли один-единственный фактор значительную нагрузку на все элементы;
- тест на маркерную переменную (Линделл и Уитни, 2001). Он проверяет, остаются ли все первоначально значимые корреляции значимыми после корректировки на вторую самую низкую положительную корреляцию среди конструкций.

Мы не видели смещения между ранними и поздними респондентами. Смещение общего метода, похоже, не является проблемой для нашей выборки.

Тестирование на взаимосвязь

В соответствии с лучшими практиками и принятым исследованием мы провели наш анализ в два этапа (Гефен и Штрауб, 2005). На первом этапе мы проводим анализ измерений, чтобы проверить и сформировать наши скрытые конструкции (см. Главу 13). Это позволяет нам определить, какие конструкции могут быть включены во второй этап нашего исследования.

Тесты модели измерения

- Анализ главных компонентов (PCA — principal components analysis). Тест для подтверждения конвергентной валидности. Этот метод используется для объяснения дисперсионно-ковариационной структуры набора переменных.
- Анализ главных компонентов проводился с переменной ротацией, с отдельным анализом для независимых и зависимых переменных (Штрауб и соавторы, 2004).
- Существует два типа PCA: подтверждающий факторный анализ (CFA — confirmatory factor analysis) и исследовательский факторный анализ (EFA — exploratory factor analysis). Почти во всех случаях мы выполняли EFA. Мы выбрали этот метод, потому что он является более строгим тестом для выявления базовой структуры переменных без наложения или предложения структуры априори.

(Одно заметное исключение было, когда мы использовали CFA для подтверждения обоснованности трансформационного лидерства, поскольку элементы в этом случае общеприняты в литературе.) Элементы должны нагружать свои соответствующие конструкции выше, чем 0,60, и не должны иметь перекрестную нагрузку.

- Усредненная дисперсия (AVE — average variance extracted). Тест, помогающий подтвердить как конвергентную, так и дискриминационную валидность. AVE — это мера величины дисперсии, которая захватывается конструкцией по отношению к величине дисперсии из-за ошибки измерения.
- AVE должна быть больше 0,50, чтобы показать конвергентную валидность.
- Квадратный корень AVE должен быть больше, чем перекрестная корреляция конструкций (когда вы помещаете квадратный корень AVE на диагональ таблицы корреляции), чтобы показать дивергентную валидность.
- Корреляция. Этот тест помогает подтвердить дивергентную валидность, когда корреляции между конструкциями ниже 0,85 (Браун, 2006). Были использованы корреляции Пирсона (подробнее см. ниже).
- Надежность.
- Альфа Кронбаха — показатель внутренней согласованности. Пороговое значение для составной надежности (CR — composite reliability) равно 0,70 (Нунналли, 1978); все конструкции соответствуют либо этому значению, либо CR (перечислены ниже). Обратите внимание, что альфа Кронбаха, как известно, смещается при небольших масштабах (то есть в конструкциях с небольшим количеством элементов), поэтому для подтверждения надежности были использованы альфа Кронбаха и композитная надежность.
- Составная надежность (CR) — показатель внутренней согласованности и конвергентной валидности. Пороговое значение для CR составляет 0,70 (Чин и соавторы, 2003); все конструкции соответствуют либо этому значению, либо альфе Кронбаха (см. выше).

Все вышеуказанные тесты должны быть проведены для того, чтобы конструкция считалась пригодной для использования в дальнейшем анализе. Мы говорим, что конструкция «проявляет хорошие психометрические свойства», и если это так, мы продолжаем. Все конструкции в нашем исследовании прошли эти тесты.

Тесты на взаимосвязь (корреляцию и прогнозирование) и классификацию

На втором этапе мы принимаем показатели, которые прошли первый этап подтверждения измерений, и проверяем наши гипотезы. Это

статистические тесты, которые используются на данном этапе исследования. Как было описано в Главе 12, в этом проекте исследования мы проводим тесты на дедуктивные предсказания, что означает, что все проверенные гипотезы поддерживаются дополнительными теориями и литературой. Если не существует никаких поддерживающих теорий, позволяющих предположить наличие предиктивных отношений, мы сообщаем только о корреляциях.

- Корреляция. Означает взаимное отношение или связь между двумя или более конструкциями. В этом исследовании мы используем корреляцию Пирсона, чаще всего используемую сегодня в бизнес-контекстах. Корреляция Пирсона измеряет силу линейной связи между двумя переменными, называемую r Пирсона. Она часто называется просто корреляцией и принимает значение между –1 и 1. Если две переменные имеют идеальную линейную корреляцию, то есть изменяются полностью согласованно, r = 1. Если они изменяются в совершенно противоположных направлениях, r = –1. Если они вообще не коррелируют, r = 0.
- Регрессия. Используется для проверки предиктивных отношений. Существует несколько видов регрессии. В этом исследовании мы использовали два типа линейной регрессии, как описано ниже.
- Регрессия частичных наименьших квадратов (PLS — partial least squares). Мы использовали ее для тестирования предиктивных отношений в 2015–2017 годах. PLS — это метод регрессии на основе корреляции, который был выбран для нашего анализа по нескольким причинам (Чин, 2010):
- этот метод используется для прогнозирования переменной результата. Поскольку мы хотели, чтобы наши результаты были полезны для практиков в отрасли, это было важно для нас;
- PLS не требует предположений о многомерной нормальности. Другими словами, этот метод не требует, чтобы наши данные были нормально распределены;
- PLS — отличный выбор для экспериментальных исследований, а это именно наша исследовательская программа!
- Линейная регрессия. Она использовалась для проверки предиктивных отношений в нашем исследовании 2014 года.

Тесты на классификацию

Эти тесты могут быть сделаны в любое время, потому что они не полагаются на конструкции.

- Кластерный анализ. Он был использован для разработки основанной на данных классификации эффективности доставки программного обеспечения, что дало нам респондентов с высокими, средними и низкими показателями. В кластерном

анализе каждая оценка помещается в отдельное измерение, и алгоритм кластеризации пытается минимизировать дистанцию между всеми членами кластера и максимизировать дистанцию между кластерами. Кластерный анализ был проведен с использованием пяти методов: метода Варда (1963), метода связи между группами, метода связи внутри групп, центроидного метода и метода медианы. Результаты для кластерных решений сравнивались с позиции: (a) изменения коэффициентов слияния; b) количества индивидов в каждом кластере (решения, включающие кластеры с небольшим количеством участников, были исключены); и c) однофакторной F-статистики (Ульрих и Маккелви, 1990). Исходя из этих критериев, решение с использованием метода Варда подходило лучше всего, и мы остановили выбор на нем. Мы использовали метод иерархического кластерного анализа, потому что:

- он обладает сильной разъяснительной силой (позволяя нам понять отношения «родители — дети» в кластерах);
- у нас не было ни отраслевых, ни теоретических оснований иметь заранее определенное количество кластеров. То есть мы хотели, чтобы данные определяли количество кластеров в нашем распоряжении;
- наш набор данных был не слишком большим. (Иерархическая кластеризация не подходит для очень больших наборов данных.)
- Дисперсионный анализ (ANOVA). Для интерпретации кластеров постфактум были проведены сравнения средств оценки результатов эффективности доставки программного обеспечения (частота развертывания, время выполнения, MTTR и частота сбоев изменений) с использованием теста Тьюки. Тест Тьюки был выбран, потому что он не требует нормальности; тест множественного диапазона Дункана также был запущен для проверки существенных различий, и во всех случаях результаты были одинаковыми (Хаер и соавторы, 2006). Попарные сравнения были проведены между кластерами с использованием каждой переменной эффективности доставки программного обеспечения, и значительные различия разделили кластеры на группы, где среднее значение этой переменной существенно не отличалось среди кластеров внутри группы, но отличалось на статистически значимом уровне ($p < 0,10$ в нашем исследовании) среди кластеров в разных группах. Во все годы, кроме 2016 (см. выноску «Сюрприз» в Главе 2), лучшие демонстрировали лучшие показатели по всем переменным, худшие демонстрировали худшие показатели по всем переменным, а средние демонстрировали средние показатели по всем переменным — все на статистически значимых уровнях.

Слова благодарности

Эта книга появилась в результате партнерства между компаниями DORA и Puppet при подготовке отчетов о состоянии DevOps. Поэтому мы хотели бы начать с благодарности команде Puppet и, в частности, Аланне Браун и Найджелу Керстену, которые были главными участниками проекта со стороны Puppet. Мы также хотели бы поблагодарить Ализу Эрншоу за ее кропотливую работу по редактированию отчетов о состоянии DevOps в течение нескольких лет. Отчет не был бы тем же без ее внимательного взгляда. Авторы также хотели бы поблагодарить нескольких человек, которые помогли разработать гипотезы, которые мы проверяем в отчете. Начиная с 2016 года мы благодарим Стивена Белла и Карен Уитли Белл за их стремление исследовать бережливое управление продуктами, а также за время, потраченное на исследование и обсуждения с командой теорий потока создания ценности и видимости обратной связи с клиентами. С 2017 года мы благодарим Нила Форда, Мартина Фаулера и Мика Керстена за элементы измерения архитектуры, а также Эми Джо Ким и Мэри Поппендик за командные эксперименты.

Несколько экспертов любезно пожертвовали своим временем, чтобы помочь оценить ранние проекты этой книги. Мы хотели бы выразить глубокую благодарность Рину Дэниелсу, Дженнифер Дэвис, Мартину Фаулеру, Гэри Груверу, Скотту Хейну, Дмитрию Кирсанову, Кортни Кисслер, Бриджит Кромхаут, Карен Мартин, Дэну Норту и Тому Поппендику.

Мы хотели бы поблагодарить Анну Ноак, Тодда Саттерстена и всю команду IT Revolution за всю их тяжелую работу над этим проектом. Наконец, благодарим Дмитрия Кирсанова и Алину Кирсанову, которые с особой тщательностью и обстоятельностью позаботились о технической редактуре, корректуре, индексации и компоновке книги. Спасибо.

Николь

Прежде всего, большое спасибо моим соавторам и соратникам, без которых эта работа была бы невозможна. Вы не выгнали меня из проекта, когда я впервые появилась и сказала вам, что все это неправильно, — вежливо, я надеюсь. Джез, я научилась терпению, сочувствию и новой любви к технологиям, которая, как я думала, угасла. Джин, благодаря твоему безграничному энтузиазму и стремлению к «еще одному анализу!» наша работа оставалась сильной и захватывающей. Данные для этого проекта поступают из отчетов о состоянии DevOps, которые были проведены вместе с Puppet Inc. Команда Puppet, Найджел Керстен и Аланна Браун, спасибо за ваше сотрудничество и помощь нам в создании истории, которая находит отклик у нашей аудитории. И, конечно же, Ализа Эрншоу: ваше мастерство выходит далеко за рамки

технического редактирования, и оно сделало мою работу бесконечно лучше. Мне нравилось, что мы можем обсуждать это до тех пор, пока не придем к согласию; когда вы сказали мне, что я «дотошно строга», это был лучший комплимент в моей жизни.

Особая благодарность моему отцу за то, что он привил мне любопытство, потребность в совершенстве и неспособность принимать плохое от людей, которые не верят в мои способности. Все это пригодилось на протяжении многих лет, особенно учитывая, что я женщина в сфере технологий. Жаль, что ты пропустил вечеринку, пап. Большое спасибо моей маме за то, что она всегда была моей сторонницей и болельщицей № 1; независимо от моих сумасшедших планов, она всегда доверяет мне. Я люблю вас обоих.

Как всегда, моя самая большая благодарность и глубочайшая признательность Хавьеру Веласкесу. Мой лучший друг и первый слушатель, ты был со мной на протяжении всего пути — когда я была вдохновлена странным исследованием юзабилити, затем во время резкого поворота в моей программе PhD, затем пригласив меня в проект отчетов о состоянии DevOps, и теперь, наконец, эта книга. Твоя поддержка, ободрение и мудрость — в жизни и в технологиях — были неоценимы.

Сьюзи! Как мне вообще так повезло? У меня был советник, который сделал ставку на аспирантку, пообещавшую, что изучение технических профессионалов, их инструментов и их среды — и как все это влияет на их работу — будет важным и актуальным. (Те, кто занят в лучших программах PhD, поймут, что это на самом деле азартная игра, связанная с реальным риском.) Десять лет спустя мое исследование выросло и развилось, и мы называем это DevOps. Большое спасибо вам, Сюзанна Вайсбанд, за то, что доверяли моим инстинктам и руководили моим исследованием в те ранние годы. Вы были лучшим советником, болельщиком, а теперь и другом.

Моим советникам, наставникам и частым соавторам по научному рецензированию в период после защиты докторской Александре Дурчиковой и Радживу Сабхервалу: вы также рисковали, проводя со мной исследования в новом контексте, и я многому научилась из нашего сотрудничества. Мои методы стали сильнее, аргументы более обоснованными, а моя способность видеть проблемные места более развита. Спасибо.

Большое спасибо сообществу DevOps, которое приветствовало и приняло сумасшедшего исследователя, приняло участие в исследовании и поделилось своими историями. Благодаря вам моя работа стала лучше и, что более важно, я стала лучше. Очень вас люблю.

И, наконец, спасибо Diet Coke за то, что она помогла мне пройти через долгие периоды написания и редактирования.

Джез

Большое спасибо моей жене и лучшему другу Рани за поддержку в работе над этой книгой даже после того, как я пообещал, что больше писать не буду. Вы самые лучшие! Я люблю вас. Спасибо моим дочерям за то, что они привнесли столько веселья и радости в этот процесс, и моим маме и папе за поддержку моих приключений с компьютерами в детстве.

Николь взяла отраслевой опрос компании Puppet «Отчет о состоянии DevOps» и превратила его в научный инструмент. Наша отрасль всегда боролась с применением науки к разработке и эксплуатации программных продуктов и услуг. Социальные системы, поддерживающие доставку программного обеспечения, слишком сложны, чтобы проводить рандомизированные контролируемые эксперименты на практике. В ретроспективе решение было ясным: использовать поведенческую науку для изучения этих систем. Кропотливое, глубокое новаторство Николь в этом подходе привело к невероятным результатам, и трудно переоценить влияние ее работы. Для меня было честью стать ее партнером в этом исследовании, и я очень многому научился. Спасибо.

Причина, по которой я вообще оказался в этом проекте, — это Джин, который пригласил меня стать частью команды State of DevOps еще в 2012 году. Джин, твоя страсть к этому проекту и лично к оспариванию моих гипотез и анализа (да, я говорю о магистральной разработке) сделала работу как значительно более строгой, так и очень полезной.

Я также хочу поблагодарить команду Puppet, чей вклад в эту работу сложно переоценить и без которой она вообще бы не состоялась, особенно Аланну Браун, Найджела Керстена и Ализу Эрншоу. Спасибо.

Джин

Я благодарен Маргарет, моей любящей жене на протяжении двенадцати лет, а также моим сыновьям Риду, Паркеру и Гранту — я знаю, что не смог бы заниматься любимым делом без их поддержки и терпимости к дедлайнам, работе по ночам и круглосуточным сообщениям. И, конечно же, моим родителям, Бену и Гейл Ким, за то, что помогли мне стать «ботаником» в раннем возрасте.

Это исследование с Джез и Николь было одним из самых насыщенных и просветляющих, над которыми я когда-либо имел честь работать, — никто не мог бы пожелать лучшей команды соратников. Я искренне верю, что эта работа значительно продвигает нашу профессию, помогая нам лучше определить, как мы совершенствуем технологическую работу через строгое построение и проверку теории.

И, конечно же, спасибо Аланне Браун и Найджелу Керстену из Puppet за удивительное более чем пятилетнее сотрудничество по проекту State of DevOps, на котором основано так многое из этой книги.

Об авторах

Доктор Николь Форсгрен является генеральным директором и главным научным сотрудником компании DevOps Research and Assessment. На сегодняшний день она наиболее известна как ведущий исследователь крупнейших проектов по изучению DevOps. Ранее она была профессором и инженером по производительности, а ее работы были опубликованы в нескольких научных журналах.

Джез Хамбл является соавтором книг «Руководство по DevOps» (The DevOps Handbook) и «Бережливое предприятие» (Lean Enterprise). За свою работу «Непрерывная доставка» (Continuous Delivery) он также получил премию Jolt. В настоящее время изучает, как создавать высокоэффективные команды в своем стартапе — компании DevOps Research and Assessment, LLC, а также преподает в университете UC Berkeley.

Джин Ким — многократно удостоенный наград технический директор, исследователь и автор «Проекта Феникс» (The Phoenix Project), «Руководства по DevOps» (The DevOps Handbook) и «Руководства по Visible Ops» (The Visible Ops Handbook). Он является основателем компании IT Revolution, а также основателем и организатором конференций DevOps Enterprise Summit.

[1] Здесь и далее коммит (от англ. commit) — сохранение, фиксация изменений в программном коде. — Прим. ред.

[2] Shift Left — устойчивый термин, обычно означающий привлечение команды тестировщиков на ранней стадии разработки ПО. Здесь и далее — встраивание информационной безопасности в процессы разработки и доставки ПО вместо выделения ее в отдельную фазу. — Прим. ред.

[3] Важно отметить, что «Отчет о состоянии DevOps» был запущен еще до 2014 года. В 2012 году команда в Puppet Inc. пригласила Джина принять участие во второй итерации исследования, которое он разрабатывал, чтобы лучше понять малоизвестный феномен под названием DevOps: как он был принят и какие преимущества от него увидели организации. Puppet Inc. была ярым сторонником и драйвером движения с момента, когда идея DevOps начала формироваться после первых конференций DevOpsDays, обсуждений в Twitter и плодотворной беседы Джона Олспоу и Пола Хаммонда. Затем Джин пригласил Джеза присоединиться к исследованию, и вместе они собрали и проанализировали 4000

опросников со всего мира, что сделало его самым крупным исследованием в своем роде.

[4] Эти 24 возможности вместе с указателем глав, в которых каждая из них обсуждается, перечислены в Приложении А.

[5] Есть хорошая история о том, как руководство команды Apple Lisa обнаружило, что строки кода были бессмысленными в качестве показателя производительности: https://www.folklore.org/StoryView.py?-story=Negative_2000_Lines_Of_Code.txt

[6] Строго говоря, частота развертывания обратно пропорциональна размеру пакета — чем чаще мы развертываем, тем меньше размер пакета. Дополнительные сведения об измерении размера пакета в контексте управления ИТ-сервисами см. Форсгрен и Хамбл (2016).

[7] Более детальную информацию по кластерному анализу смотрите в Приложении В.

[8] В 2016 году 31% респондентов были классифицированы как патологические, 48% — как бюрократические и 21% — как производительные.

[9] Эти гипотезы основаны на предыдущих исследованиях и существующих теориях и подкреплены нашим собственным опытом и опытом других специалистов в отрасли. Таким образом построены все наши исследовательские гипотезы. Это пример дедуктивно-прогностического исследования, о котором вы можете подробнее прочитать в Главе 12.

[10] История этого преобразования рассказана в эпизоде 561 радиошоу This American Life на радиостанции WBEZ (2015).

[11] Согласно Google Trends, Scrum обогнал экстремальное программирование примерно в январе 2006 года, и его популярность продолжала расти, в то время как XP практически сошло на нет.

[12] Ключевой шаблон, который соединяет эти циклы обратной связи, известен как конвейер развертывания.
См. https://continuousdelivery.com/implementing/patterns/

[13] User experience (пользовательский опыт) — сфера на стыке дизайна и аналитики, сфокусированная на создании прототипов на основе анализа ощущений пользователей. — Прим. ред.

[14] Значимым исключением является развертывание автоматизации.

[15] Из-за ограничений по длине было протестировано только подмножество технических возможностей. См. диаграмму в конце Приложения А.

[16] Дополнительную информацию смотрите по ссылке https://martinfowler.com/articles/nonDeterminism.html

[17] Описание GitHub Flow смотрите по ссылке: https://guides.github.com/introduction/flow/

[18] Мы определяем интегрированную среду как среду, в которой несколько независимых служб развертываются вместе, как, например, промежуточная среда. На многих предприятиях интегрированные среды являются дорогостоящими и требуют значительного времени на настройку.

[19] Дополнительную информацию см.: https://www.thoughtworks.com/radar/techniques/inverse-conway-maneuver

[20] «Платформа болтовни» Стива Йегге содержит несколько отличных советов по достижению этих целей: http://bit.ly/yegge-platform-rant

[21] Дополнительную информацию см.: https://www.owasp.org/index.php/-Category:OWASP_Top_Ten_Project

[22] Risk Management Framework (RMF). — Прим. ред.

[23] National Institute of Standards and Technology — Национальный институт стандартов и технологий (США). — Прим. ред.

[24] Security assessment report (SAR) — отчет об оценке безопасности. — Прим. ред.

[25] Подробнее о конвейерах развертывания см.: https://continuousdelivery.com/implementing/patterns/

[26] https://www.devopsdays.org/events/2016-london/program/thiago-almeida/

[27] Один из примеров набора архитектурных шаблонов, которые обеспечивают этот вид процесса, можно найти по ссылке https://12factor.net/

[28] Отметим, что в научной литературе есть и другие модели выгорания; одним из заметных примеров является работа Мари Осберг, старшего профессора кафедры клинических наук Каролинского института, Швеция. Мы же в нашем исследовании сосредоточились на работе Маслах.

[29] Обратите внимание, что проблемы, возникающие после развертывания, также важно отследить. Ломающиеся системы, которые постоянно вызывают ваш дежурный персонал в нерабочее время, разрушительны и нездоровы.

[30] STEM — science, technology, engineering, mathematics. — Прим. ред.

[31] Отметим, что Лесли и соавторы в своем исследовании изучали только женщин и афроамериканцев, но результаты, вероятно, можно обобщить и для других меньшинств.

[32] https://anitab.org/

[33] http://geekfeminism.wikia.com/wiki/Geek_Feminism_Wiki

[34] http://projectinclude.org/

[35] Наш анализ подтвердил, что эти вопросы являются хорошими показателями трансформационного лидерства. См. Главу 13 для обсуждения скрытых конструкций и приложение С для используемых статистических методов.

[36] Johns Hopkins Bloomberg School of Public Health. — Прим. ред.

[37] http://www.tylervigen.com/spurious-correlations

[38] Корреляции Пирсона измеряют силу линейной связи между двумя переменными, которую называют r Пирсона. Часто она просто называется корреляцией и принимает значения от –1 до 1. Если две переменные имеют идеальную линейную корреляцию, то есть если они изменяются точно вместе, r = 1. Если они изменяются в совершенно противоположных направлениях, то r = –1. Если они вообще не коррелируют, то r = 0.

[39] Эти элементы обычно называют вопросами. Однако на самом деле это не совсем вопросы — это утверждения. В данной книге мы будем говорить о них как об элементах опроса.

[40] Redundant Array of Independent Disks — избыточный массив независимых дисков. — Прим. ред.

[41] Это, конечно, предполагает, что вы занимаетесь сбором данных с целью улучшения, не сообщая каждому, что он должен отвечать положительно или как-либо иначе. Это было бы равносильно шутке: «Избиения будут продолжаться до тех пор, пока не поднимется боевой дух». Вы бы получили необходимые вам данные — хорошие ответы, но это было бы лишено смысла. Одним из способов поощрения честных ответов является обеспечение анонимного сбора данных.

[42] Интересный пример использования удержания кадров как способа определения эффективности процесса собеседования можно найти в работе Канемана 2011 года.

[43] Проект поперечного исследования означает, что данные были собраны в один и тот же момент времени. Однако это исключило для нас возможность проведения продольного сбора данных, поскольку наши ответы не были связаны между собой от года к году. Повторяя данное исследование в течение четырех лет, мы смогли выявить закономерности, характерные для всей отрасли. Хотя нам и хотелось бы собрать продольный набор данных, то есть такой, для которого мы выбираем одних и тех же людей из года в год, однако это могло бы снизить частоту ответов из-за проблем с конфиденциальностью. (А что происходит, когда эти люди меняют команду или место работы?) В настоящее время мы проводим исследования в этой области. Проект поперечного исследования имеет свои преимущества: сбор данных в один момент времени уменьшает вариативность в ходе исследования.

[44] Вероятностная выборка представляет собой любой метод статистической выборки, использующий случайный отбор; в более

широком смысле детерминированная выборка — это любой метод, не использующий случайный отбор. Случайный отбор гарантирует, что все индивиды в популяции имеют равные шансы попасть в выборку. Поэтому вероятностная выборка, как правило, является предпочтительной. Однако методы вероятностной выборки не всегда возможны вследствие средовых или контекстуальных факторов.

[45] См. Главу 11.

[46] Примечание от Николь, Джеза и Джина. Термин «ИТ» в этой главе используется для обозначения программного обеспечения и технологического процесса, то есть намного шире, чем просто одиночная функция в рамках технологической группы в компании, как ИТ-поддержка или служба технической поддержки клиентов.

[47] См. Главу 2.

[48] Эта и все другие прямые цитаты сотрудников ING почерпнуты из личного общения с авторами данной главы.

[49] Мы принимаем решение о нашей исследовательской модели каждый год на основе обзора литературы, обзора результатов наших предыдущих исследований и здоровой дискуссии.

Переводчик А. Техненко

Редактор Е. Закомурная

Руководитель проекта М. Пикалова

Дизайн Е. Шестернина

Корректоры Е. Якимова, Е. Тимошкина

Компьютерная верстка Б. Руссо

Форсгрен Н., Хамбл Д., Ким Д.

Ускоряйся! Наука DevOps: Как создавать и масштабировать высокопроизводительные цифровые организации / Николь Форсгрен, Джез Хамбл, Джин Ким; Пер. с англ. А. Техненко. — М.: Интеллектуальная Литература, 2020.

ISBN 978-5-9072-7433-4