Культуры программных проектов

Энтони Лаудер

Перевод: Альберт Мустафин

Опубликовано: Happy-PM.com

Содержание

Предисловие

Почему я написал эту книгу?

Я взялся за написание этой книги почти 10 лет назад, потому что коллеги, клиенты и друзья просили меня об этом. Общим у этих людей было ощущение наводнения различными методологиями разработки приложений, каждая из которых прокламирует свои собственные «правила» управления проектами и разработки.

Люди пытались понять, что имеет смысл на практике, а что просто хорошо выглядит на бумаге. Мне говорили, что трудно представить общую картину; детали захлёстывают с головой. Поэтому я начал писать книгу, предназначенную именно для таких людей. Она рассмотрит широкий диапазон методологий разработки приложений и объяснит, как они связаны между собой и чем различаются.

После нескольких месяцев глумления над клавиатурой я выпустил несколько глав и был шокирован, когда несколько рецензентов из тех людей, кто побудил меня написать эту книгу, сказали: «Интересно. Но какая тут практическая польза?»

Моё эго было малость подавлено, поэтому я перестал писать. Тем не менее вопрос практической пользы никуда не делся. Он всё время крутился у меня в голове.

В течение нескольких следующих лет консалтинга я стал замечать, что и другие люди задают похожие вопросы:

- Почему наш начальник заставляет нас следовать методологии, которая очевидно нам не подходит?
- Почему программисты не делают того, что им говорят?
- Почему я не могу внедрить у нас в компании методологию АБВ?

Постепенно до меня дошло, что людям не нужно говорить *как* они *должны* работать, они хотели понять, *почему* разные люди *делают* работу по-разному.

Они спрашивали меня не столько о методологиях, сколько о том, как методологии срабатывают или дают осечку в конкретных культурах разработки приложений. И не только это, но и:

- Если это наша культура, то что нужно сделать, чтобы изменить её?

Чем эта книга отличается от других?

Эта разница между методологиями и культурами кажется важной. Никто о ней не писал, но люди спрашивали об этом. На рынке явно существовала пустая ниша, которая убедила меня вернуться к написанию этой книги.

Эта книга показывает, как осведомлённость о культурах разработки приложений может отделить успех от провала в реальных проектах. Вместо лишнего теоретизирования книга на жизненных примерах углубится в то, какие бывают культуры разработки приложений, откуда они происходят и какое влияние оказывают. Три доминирующие культуры будут определены и разобраны в деталях наряду с четвёртой культурой, которая сейчас совсем мало используется, а потому вынесена в приложения.

Мы посмотрим на историю каждой из трёх доминантных культур, их базовые предпосылки и методики разработки приложений, соответствующие этим культурам.

Вооружившись углубленным осознанием культур книга исследует довольно интересный вопрос - как очень успешные компании изменили свои культуры. Также мы увидим, что для закрепления культурных изменений нужно больше, чем навязывание новой методологии в командах. Культура разработки приложений должна постепенно проталкиваться по культурной лестнице, изменяя один уровень за другим.

Feedback

The most important part of this book is you. You and other readers will know far better than I possibly

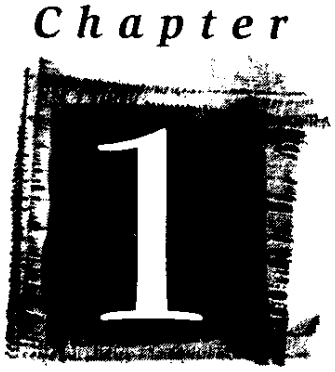
can how relevant the ideas here are to your daily work. Do let me have your feedback. I am sure I can learn far more from you, than you could ever learn from me. Together, we can make future editions of this book more valuable to its audience.

Thank you

Anthony Lauder

anthony@anthonylauder.com

Глава 1: Введение



О культурах

Если из всей книги вы усвоите только одну вещь, то пусть ею будет это: культура имеет значение. И большое значение. Культура разработки приложений не то же самое, что методология разработки приложений. Под методологией понимается набор правил, которых надо придерживаться, а культура - это набор укоренившихся устоев, вокруг которых сплотилась группа людей. Методология может говорить людям как себя вести, а культура определеяет человеческую потребность вести себя определённым образом.

Культура определяет эмоциональную реакцию людей на навязываемые правила. Если методология и культура не совпадают, люди инстинктивно чувствуют обеспокоенность, и методология встречает неприязнь. Если менеджер является приверженцем одной культуры, а его/её команда сплотилась вокруг другой, то все будут чувствовать себя не в своей тарелке, и произойдёт болезненное столкновение культур. В отношениях появляется антагонизм и прогресс тормозится.

Конечно, если вы обладаете некоторой властью над кем-то - например как менеджер над своей командой - вы можете заставить их делать то, что вам нужно. Если всё, что вам нужно, это послушание, то никаких проблем. Именно так заключённых в цепях заставляют целый день дробить камни. К сожалению, я очень много раз видел, что этот подход не очень работает в индустрии разработки приложений. И вот почему: Хотя угрозы и принуждение могут заставить людей подчиниться, это не может пробудить в них добровольное сотрудничество. Вы можете кричать, визжать и наказывать сколько угодно, но без добровольного сотрудничества люди никогда не вложат в свою работу ни сердца ни души. В лучшем случае результаты будут так себе.

Если вы хотите исключительных результатов, то нужно принять во внимание культуру.

Следуйте за своими эмоциями

Мы начнём сразу с вопросов с несколькими вариантами ответов, которые помогут исследовать вашу собственную реакцию на несколько разных аспектов разработки приложений. Ваши ответы покажут, насколько сильны ваши эмоции и насколько сильным может быть их влияние на то, какой культуры вы придерживаетесь.

Отвечая на эти десять вопросов полагайтесь, в основном на свои инстинкты. Обратите внимание на то, какие ответы вас беспокоят, а с какими вы чувствуете себя комфортно. Иногда более одного ответа будут казаться правильными, а иногда ни один. Не пытайтесь углубляться в анализ - позже в книге для этого будет достаточно времени. Пока что просто выберите то, что кажется правильным.

По окончании опросника мы рассмотрим ваши ответы. Результаты должны показать, что ваши эмоциональные реакции делают вас приверженцем одной культуры разработки и противником других.

Десять Вопросов

1: От кого больше всего зависит успешность проекта разработки приложений?

А: От менеджера: Эффективное управление проектом может быть ключевым параметром, разделяющим проекты, которые не укладываются в бюджет и сроки и производят приложения низкого качества, и проекты, удовлетворяющие требованиям, приносящие существенную прибыль и завершённые в рамках отведённого времени и бюджета.

Б: Техническая команда: Управление - это в основном администрирование, а потому мало что привносит в создание классных приложений, для чего требуются талантливые, опытные и очень техничные команды разработчиков.

В: Заказчики: Без заказчиков не будет приложений. Только заказчики знают, что им на самом деле нужно и что представляет ценность. Поэтому заказчики должны оставаться у руля, чтобы мы могли фокусироваться на постоянном изготовлении приложений, которые имеют для заказчиков наибольшее значение и решают их наиболее актуальные задачи.

2: Как лучше всего собирать требования?

А: Собрать их полностью в самом начале: В идеале, заказчик даст нам подробный список требований, или же мы можем поработать с заказчиком, чтобы выявить все нужды заранее, что даст нам фиксированную чёткую цель, которой должен будет достичь проект.

Б: Нужно дать им стабилизироваться: Вначале требования обычно не сильно чёткие, но детали проясняются со временем, по крайней мере, для наиболее важных требований, и всё больше помогают нам стабилизировать дизайн архитектуры, на которой будет строиться вся система.

В: Будьте готовы к изменениям: Мы не можем полагать, что требования когда-либо стабилизируются. Ранее казавшиеся важными требования теряют важность со временем, а непредвиденные события приводят к появлению совершенно новых и неожиданных нужд заказчика. Мы должны фокусироваться только на тех требованиях, которые наиболее важны для заказчика на данный момент, и ожидать, что эти приоритеты будут постоянно изменяться.

3: Где лучше всего записывать требования, дизайн и другие решения?

А: Текстовые документы: Пишите документы, в основном текстовые, но со вспомогательными графиками и диаграммами, если нужно для более полного охвата требований, дизайна и других решений, чтобы информация была задокументирована, легко доступна для других, и не терялась.

Б: Формальные заметки: Используя формальные графические обозначения, создавайте диаграммы, чтобы выражать требования и дизайны более чётко и ясно, чем в запутанных текстах. Можно использовать специальные приложения, умеющие работать с такими диаграммами.

В: Человеческий мозг: Тесно сотрудничайте с коллегами и клиентами, чтобы убедиться, что постоянно развивающиеся мысленные представления о меняющихся требованиях и дизайне остаются всегда тесно синхронизированы.

4: Предположим, у нас есть требования. Какова следующая цель?

А: Преобразование: Требования должны быть переведены в дизайн, который в свою очередь может быть воплощён в программном коде.

Б: Построение модели: Приложение всегда поддерживает некую часть реального мира. Поэтому мы должны строить модели, улавливающие суть этих частей мира, выражающие их наиважнейшие вещи и их связи друг с другом. Затем мы можем отобразить эти абстракции в дизайне и исполнении в виде чёткой программной архитектуры, на которой с уверенностью может быть построено целое приложение.

В: Поставка приложения: Документация, модели и другие промежуточные артефакты важны только если они помогают отдать приложение в руки заказчика. Поэтому, сфокусируйтесь на быстрых и частых поставках работающего приложения, пусть для начала очень базового, чтобы заказчик смог начать получать выгоду от него как можно раньше.

5: Как может помочь менеджер в ускорении прогресса проекта?

А: Планирование и контроль: Создайте план, включая расписание, затем следите и контролируйте проект, чтобы убедиться, что он чётко следует плану, а все люди упорно работают, чтобы выполнить взятые на себя обязательства.

Б: Руководство: Наберите в проект правильных людей, помогите им притереться, дайте им направление, если нужно, но уйдите с дороги, если они быстро выполняют задания без вашего вмешательства.

В: Поддержка: Заохотьте команду уведомлять вас о препятствиях, возникающих на их пути, которые мешают прогрессу или угрожают срокам. Затем поддержите команду, используя все свои возможности и, таким образом, увеличив шансы команды на успех.

6: Как лучше всего контролировать стоимость проекта?

А: Управление проектом: Чтобы использовать все ресурсы как можно эффективнее, используйте проверенные методики управления затратами.

Б: Наймите классных людей: Наймите самых лучших людей, так как они в десять раз эффективнее среднестатистических, а стоят всего раза в два больше.

В: Фокус на ценности: Пишите приложение только для наиболее важных требований заказчика, чтобы постоянно удерживать высокое отношение ценности минус стоимость.

7: Как мы можем убедиться в высоком качестве приложения?

А: Процесс: Найдите надёжный, проверенный способ разработки приложений с серией полностью определённых и повторяемых шагов, в каждом из которых качество уже встроено. Каждому сотруднику дайте чётко определённую роль с конкретным спектром ответственности, а потом требуйте буквального соблюдения процесса.

Б: Инструменты: Вместо того, чтобы мучить людей старым оборудованием и устаревшими принципами и практиками, дайте им наилучшие инструменты разработки

приложений и образование по новейшим методикам, чтобы они могли работать как можно более продуктивно и эффективно, производя наилучшие результаты.

В: Обратная связь: Поскольку заказчик решает, насколько приложение хорошо, разработайте с ним методы раннего определения, удовлетворяет ли приложение требованиям. Полагайтесь на обратную связь с заказчиком, чтобы определить, что сделает приложение лучше.

8: Какова наилучшая структура эффективной команды разработчиков?

А: Определённые роли: Каждому человеку предписана определённая роль - аналист, программист, тестер - с чёткими задачами и областями ответственности. Каждая из них привносит свою долю в общую последовательность работ, необходимых для завершения проекта.

Б: Профессионализм: Оставьте архитектуру самым умным и талантливым техническим сотрудникам. Остальные пусть приносят пользу там, где они больше всего подходят.

В: Самоорганизация: Мотивированные люди, сплочённые вокруг общих ценностей и целей, должны поощряться на самоорганизацию так, как им это кажется наиболее подходяще для достижения максимальной эффективности.

9: Как избежать ненужной работы?

А: Исключите брак: Работа спустя рукава выбивает людей из колеи, что приводит к переделкам. Поэтому убедитесь, что каждый сотрудник полностью исполняет свои обязательства и делает работу высокого качества, прежде чем передаёт её другим.

Б: Уменьшить бюрократию: Не тратьте время на длинные документы, которые далеко не все будут читать. То же относится и к статус репортам, часто являющимся отписками. Вместо этого сфокусируйтесь на моделировании ядра деловой области, которую поддерживает приложение. Использование программ для записи этих моделей обеспечивает непрерывность и отслеживаемость кода через дизайн назад к требованиям.

В: Не делайте её: Не работайте ни над чем, если заказчик не видит в этом большой ценности. Задачи с низким приоритетом не стоят предварительного исследования. А когда их ценность возрастёт так, что обстоятельства изменятся, то их всё равно придётся переделывать скорее всего.

10: Что указывает на завершённость проекта?

А: Завершённые задачи: Когда все запланированные задачи были выполнены правильно, проект завершён.

Б: Стабильность: Когда дизайн приложения достаточно стабилен и надёжен, приложение может быть отдано в руки заказчика.

В: Ценность для заказчика: Когда заказчик решит, что следующая версия приложения не даст достаточной дополнительной ценности, останавливаем работу.

Ваши результаты.

Некоторые ответы показались вам логичными, другие немного неполноценными или даже абсолютным бредом. Вполне возможно, что выбранные вами ответы были разбросаны между А, Б и В вариантами, но скорее всего можно проследить некую тенденцию. У большинства людей есть некий инстинкт, приводящий их к тому или иному набору ответов.

Эти инстинкты важны, поскольку они формируют наши ощущения и реакцию на всевозможные аспекты разработки приложений. Они приводят нас к конкретным методологиям, основываясь не на анализе каждого возможного правила, а на нашем общем ощущении. Кроме того они нам помогают распознать единомышленников. Если большинство ваших ответов А как и у кого-то ещё, то с этим человеком вы очень хорошо сработаетесь. Но вам будет довольно сложно ужиться с людьми, предпочитающими ответы Б или В. Особенно в вопросах ведения проектов. Поэтому ваши ответы на опросник не только помогают раскрыть некоторые из ваших эмоциональных предпочтений, но и дают возможность определить тип культурной группы, в которой вам будет наиболее комфортно.

Если вы выбрали в основном один тип ответов, то вполне возможно следующее будет вам тоже импонировать:

Большинство А

Основная масса ответов Б скорее всего потакает требованиям технарей и упускает из виду управленческий аспект, необходимый для контроля рисков проекта.

Возможно вам кажется, что большинство из ответов В слишком интуитивно-чувственны и не вполне базируются на лучших практиках, чтобы применяться в жизни.

Если вы активно вовлечены в разработку приложений, то вполне возможно, что вы используете или склоняетесь к Водопадной методике, вроде методы Gane and Sarsen или Systems Analysis and Design Method (SSADM).

Скорее всего вы хорошо впишетесь в организацию с доминирующей Заводской Культурой. Чуть позже будет краткое описание этой и двух других доминирующих культур. Лучше немного подождать, но если не терпится, то можете проскочить несколько страниц и подсмотреть.

Большинство Б

Возможно в прошлом вы выбрали бы больше ответов А, но со временем ваши взгляды эволюционировали, увидев как задыхаются проекты, хотя в теории всё звучит логично.

Может и ответы В в чём-то звучат близко по духу, но недостаточно веско на ваш взгляд, и им не хватает технической твёрдости вашего собственного подхода.

Если вы активно вовлечены в разработку приложений, то полне возможно, что вы используете или склоняетесь к Объектно-Ориентированному подходу (ООР).

Скорее всего вы хорошо впишетесь в организацию с доминирующей Дизайнерской Культурой.

Большинство В

Ответы А вам кажутся несовместимыми с истинной природой разработки приложений и являются пережитком времён, когда люди недооценивали сложность правильного ведения разработки приложений.

Возможно, ответы Б вам импонировали в прошлом и до сих пор представляют некоторую выгоду, но теперь вы считаете, что они слишком фокусируются на технических аспектах и забывают о более важной вещи - следовать за постоянно меняющимися нуждами заказчика.

Если вы активно вовлечены в разработку приложений, то вполне возможно, что вы используете или склоняетесь к гибким методам вроде Scrum или Crystal.

Скорее всего, вы хорошо впишетесь в организацию с доминирующей Сервисной Культурой.

Пример из жизни

Надеюсь, что этот опросник проявил для вас эмоциональные составляющие разных культур. Это может помочь в понимании тщетности навязывания новой многообещающей методики человеку или группе, если эта методика конфликтует с их собственной доминирующей культурой. Навязанная конфликтующая методология будет выводить людей из равновесия, что повлечёт их неуверенность в методике. Они будут инстинктивно сопротивляться ей и сомневаться в ваших мотивах при навязывании.

Проблема в том, что навязывание может заставить изменить поведение. Но оно не может изменить ни доминирующий культурный склад ума ни инстинктивные реакции, обусловленные этим складом ума. Результаты скорее всего будут печальными. Основной тезис этой книги - если хотите лучших результатов, учитывайте культуру.

Конечно, всё это может быть просто интеллектуальными домыслами. Поэтому позвольте мне усилить этот тезис примером из жизни, где хорошо известная компания пренебрегла своей собственной культурой разработки приложений и в результате потеряла огромную долю рынка. Надеюсь, что этот пример чётко покажет влияние, которое культуры разработки приложений имеют в реальном мире, и объяснит, почему оно должно быть важным и для вас.

Времена конкуренции

Не так давно я работал пару месяцев с одной известной “dot com” компанией. Дабы сохранить анонимность, будем называть её СамоСервис.

Самосервис революционизировал индустрию, в которой промышлял, переведя её из клиент-сервис разряда в онлайновый бизнес, где люди обслуживали себя сами (Internet-based self-service). На этом СамоСервис неплохо заработал - годовой оборот составлял несколько миллиардов долларов. И тем не менее, их рыночное доминирование стало снижаться довольно высокими темпами. За прошлые два-три года несколько конкурентов стали вырываться вперёд. Они стали отхватывать куски рынка с ужасающей скоростью. Руководство СамоСервиса сказало мне, что они серьёзно озабочены. СамоСервис нанял меня, и я провёл два месяца исследуя, что же было не так.

Я обнаружил, что у СамоСервиса сильное руководство, классные технологии, супер-умные люди и захватывающий своей высотой дух бюджет. Судя по этому, казалось, что

они всё делают правильно. Чего им серьёзно не хватало, так это способности вовремя реагировать и адаптировать бизнес достаточно быстро, чтобы опережать конкурентов. Разумеется они понимали, что требовалось провести в плане изменения бизнес процессов и разработки приложений, но непосредственная реализация этого всегда занимала слишком много времени. Это напоминало плавание в киселе.

Заводская Культура

Что-то сдерживало СамоСервис. Что-то не давало компании корректировать бизнес процессы и приложения со скоростью, которую держали конкуренты. Каждый день СамоСервис всё больше отставал.

Они пробовали мотивационные речи; они пробовали угрозы; они пытались вливать деньги в проблемные участки; но ни один из подходов не оказался достаточно успешным. Тогда я решил порасспрашивать вокруг. Я говорил с менеджерами, я говорил с тимлидами, и говорил с рядовыми сотрудниками. В результате я обнаружил, что СамоСервис является наиболее типичным представителем Заводской Культуры, что и сдерживало бизнес.

Заводская Культура подразумевает, что проект может и должен вестись как конвейерное производство на заводе. Работники должны быть расставлены в ряд вдоль конвейера, где каждый выполняет чётко определённые действия согласно рассписанию, представленному в плане. Каждый работник такого конвейера получает для работы то, что сделал работник, стоящий перед ним, производит свои действия согласно плану и полученное изделие передаёт дальше по конвейеру. Завершённые изделия сходят с конвейера вовремя, согласно бюджету, и в точности по спецификациям. Менеджеры следят, чтобы работники строго придерживались плана, и наказывают нарушителей, ибо шаг в сторону от плана угрожает успеху проекта.

Доводы, обосновывающие Заводскую Культуру, звучат убедительно, но в этом-то и кроется проблема. Они были действительно убедительными во времена медленно развивавшегося мира бизнеса 60-70-х годов, когда Заводская Культура находилась на пике популярности в индустрии программных приложений. В те времена компьютеры в основном предназначались для технарей, занимавшихся решением алгоритмических задач в закрытых кабинетах. Бизнес процессы менялись редко. Требования к приложениям были стабильны. Кроме того, в те времена значимость программ была не так существенна.

Как изменились времена? В наши дни программные приложения стали настолько важны для эффективной работы любой организации, что было бы неразумным разделить развитие обеих вещей. В сегодняшнем скоростном деловом мире программы, не поспевающие за изменениями бизнес процессов, затормаживают те самые бизнес процессы и становятся причиной убытков. Программные приложения переходят из графы капиталовложений в графу расходов. В эти конкурентные времена ни один бизнес не может себе такого позволить.

На удивление, много компаний, включая СамоСервис, всё ещё разрабатывают приложения так, будто бизнес процессы не имеют особого значения: используют методики разработки, которые были созданы для решения проблем, с которыми бизнес сталкивался в 60-70х годах.

Правила, которые существовали в те спокойные времена нынче просто не работают. Теперь вместо помощи в достижении успеха они мешают. Это фактически извращение. Представьте себе, что какая-нибудь другая часть вашего бизнеса придерживается тех же

методик, что и сорок лет назад. Вам бы это не понравилось. Это же не должно нравиться и в разработке приложений.

Дизайнерская Культура

Вместо того, чтобы продолжать тратить время на исследование того, что СамоСервис делал не так, я решил выяснить, что же такого правильного делают наиболее успешные конкуренты СамоСервиса.

Удивительно, как много можно узнать о внутренних делах компании, немного порывшись вокруг. А потому я скоро узнал, что многие из конкурентов СамоСервиса отвернулись от Заводской Культуры и стали культивировать Дизайнерскую Культуру.

Дизайнерская Культура основана на совершенно других посылах в отношении разработки приложений. В частности, она предполагает, что будущие изменения бизнес процессов неизбежны. Приложения, которые не готовы к таким изменениям, довольно хрупки и скоро перестают удовлетворять нуждам бизнеса. С другой стороны, гибкие приложения разработаны с пониманием этой концепции. У гибкого приложения есть сильная стабильная архитектурная основа со встроенными частями, которые позволяют вносить изменения, соответствующие изменениям бизнес процессов. Эта готовность к изменениям позволяет приложению быстро адаптироваться и оставаться ценным для нужд бизнеса. Хорошие архитекторы создают такие архитектуры, делая масштабируемые модели для исследования и уточнения неясностей, чтобы решить сложные проблемы, чтобы поэкспериментировать с многообещающами решениями, чтобы предусмотреть и приготовиться к будущим изменениям, и вцелом, чтобы дать дизайну утрястись. Как только архитектурный дизайн стабилизируется, можно уверенно строить остальные части продукта.

Культивируя Дизайнерскую Культуру, некоторые из конкурентов СамоСервиса смогли предвидеть изменения и отразит их в своих приложениях, добившись тем самым существенного преимущества перед СамоСервисом.

Но это всё же не объясняло, почему один из конкурентов СамоСервиса опережал всех прямо-таки огромными скачками. Эта компания явно делала что-то иначе, и я решил выяснить, что же это такое.

Сервисная Культура

Хоть это и заняло у меня некоторое время, но я всё же узнал, как эта компания оказывается впереди. Дело было не в людях, не в технололгиях и не в количестве денег, кидаемых на проекты. На самом деле у них было меньше людей, топорные технологии и довольно скромный бюджет.

Отличительной чертой было то, что они отошли от Заводской и Дизайнерской Культур и объединились вокруг Сервисной Культуры.

Как мы видели, Заводская Культура предполагает, что требования могут быть задокументированы до последнего болтика заранее, а изменения в требованиях нежелательны. С другой стороны, Дизайнерская Культура предполагает и готовится к будущим изменениям бизнес процессов. Это существенное улучшение, но что если вы столкнётесь с изменениями, к которым не были готовы?

Оправдания, что некие важные изменения требований не вписываются в дизайн, не помогут клиенту. Это его только задержит. Следовательно, Сервисная Культура заставляет нас перенаправить фокус наружу и руководствоваться меняющимися нуждами клиента, а не концентрироваться внутри на стабильности структуры приложения.

Сервисная Культура руководствуется тезисами, отличными от других культур. Она полагает, что приложение должно оставаться ценным в глазах клиента по мере того как меняются потребности самого клиента, пусть даже непредвиденным способом. Потому приложение должно оставаться гибким, а его дизайн должен непрерывно меняться в ответ на непредвиденные изменения. Заказчик должен занимать главную роль в определении своих меняющихся приоритетов и решении, какие функции приложения разрабатывать и в каком порядке их поставлять. Быстрые и частые поставки позволяют клиенту пользоваться функционалом приложения, пока он ещё представляет высокую ценность. Кроме того это позволяет им менять приоритет функционала для следующих поставок и останавливать разработку, когда стоимость начинает превышать потенциальную выгоду.

Почему же это имеет такое большое значение?

Заводская и Дизайнерская Культуры фокусируются на успехе *нашего* бизнеса. А Сервисная Культура во главу угла ставит успех *клиента*. Она понимает, что наш заказчик находится в непрерывной конкуренции на своём собственном рынке. Чтобы выжить и процветать, им приходится реагировать на возможности в любое время, что означает адаптирование их собственных процессов необходимым способом. Сместив наш фокус на поддержку этих меняющихся процессов мы становимся для клиента вложением, а не расходом.

Эта Сервисная Культура оказалась ключём к экстраординарному росту наиболее успешного конкурента СамоСервиса. Они не тратили деньги, время и усилия на догадки о том, что нужно клиенту. Вместо этого они вложились в построение отношений, основанных на реально доверительном партнёрстве, тесно работая с клиентами над отслеживанием и поддержкой их реальных меняющихся деловых приоритетов. Клиенты видели, что эти отношения были выгодны для их успеха, и это немедленно отражалось на успехе компании-поставщика. Всё работало как по волшебству, и это демострирует замечательный эффект, который культура может оказать на успешность организации.

Заключение

Итак, у вас могут быть лучшие технологии, самый высокий бюджет и лучшие люди, но вы будете существенно затормаживаться своей культурой. Если ваша организация держится за культуру, наполненную тезисами 1960-1970-х годов, то вы и получите результаты тех годов. Ваши конкуренты ухватятся за возможность маршировать вперёд и со временем выкинут вас из бизнеса.

Если вы абсолютно уверены, что это не относится к вашему бизнесу, то вы один из немногих счастливчиков. Вы уже в другой лиге в отличие от компаний, угнетаемых пережитками прошлого, а эта книга только подтвердит то, что вы и так уже знаете.

В то же время, никто в здравом уме не будет добровольно использовать сорокалетние практики. Большинство компаний чистосердечно полагают, что они и так уже делают всё возможное, чтобы оставаться в лидерах. И тем не менее, очень часто они даже не замечают, как всё больше насыщаются культурными тезисами/предположениями, которые

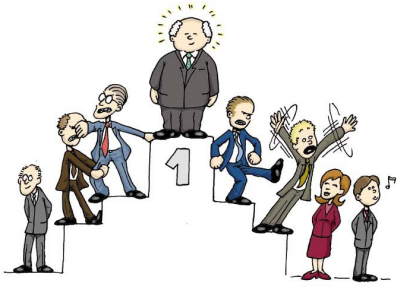
приводят их к устаревшим методикам разработки приложений, что в свою очередь сдерживает их от достижения тех результатов, к которым они стремятся.

Значит, наиболее успешные компании развивают наиболее успешные культуры. Если вы просто попытаетесь скопировать методологию, которой пользуется ваш наиболее успешный конкурент, вы не преуспеете, если ваша культура не впитает её. Успех проистекает не из выбора и навязывания методологии, а из изменения культуры так, чтобы она приняла методологию всем сердцем.

Эта книга поможет вам определить вашу собственную культуру разработки приложений. Она выявит предположения, которые фактически заставляют вас заниматься разработкой приложений именно так, как вы это делаете. Вы сами увидите, какие культуры затормаживают бизнес, а какие помогают им вырваться вперёд. Что, возможно, более важно, эта книга расскажет, как некоторые компании обновили свои культуры, преобразовав приложения из пункта статьи расходов в одно из самых лучших бизнес вложений. Как вы на это среагируете, уже ваше дело.

Глава 2: Правила игры

Я не виноват



Когда проект идёт успешно, то вполне понятно, кого надо хвалить: нас! «Я сделал всё, что мог, для этого проекта, а потому результат налицо. Без моего великого вклада проект бы не преуспел даже на малую часть.»

Когда я был студентом, то временно подрабатывал программистом в государственном учреждении. Там я встретился с Дэвидом; одним из наименее способных программистов, которых я когда-либо видел. Когда что-нибудь на проекте шло хорошо, Дэвид был уверен, что это его заслуга. Стоило чему-то пойти не так, он был убеждён в секретном сговоре завистников, желающих унизить его. В конце каждого проекта Дэвид возмущался недостатком похвалы, которую он явно заслуживал. Он ушёл с обидой. Странно, но проекты, над которыми он работал, стали улучшаться. Без сомнения, Дэвид был уверен, что это положительный эффект от его наработок.

Кого же мы можем обвинять в провале проектов? Точно не себя! Это всё те придурки!

Был 1988 год. Я только выпустился из университета и начал работать программистом. Глава компании Ричард созвал митинг всех девелоперов. «Я не доволен количеством багов в нашем софте. Если так будет продолжаться и далее, я уволю всех младших программистов, а менеджеры будут программировать.» Новые программисты, включая меня, сидели в ужасе, а более опытные закатывали глаза.

Когда проект идёт плохо, мы склоняемся к мысли «Кто вообще мог надеяться на удачу в этом месте? Проект был обречён изначально, с этими клоунами в командах и ленивыми неудачниками, отвечающими за критические участки работы!»

Почти в каждой организации есть похожие истории: «Эти аналитики замедляют проекты, вместо того, чтобы помочь им.»; «Мой менеджер - это дурной босс из Дилберта.»; «Этот чувак просто примадонна: слишком занят, гоняясь за техническим совершенством, вместо того, чтобы сделать что-нибудь.»

Год 2008. Я директор в большом известном доткоме. Логан, старший директор, не мог понять, почему постоянно пропускаются сроки. Он следил за сроками столько, сколько кто-нибудь мог вспомнить, но несмотря на постоянные изменения тактики, они постоянно ускользали у него из рук. У Логана возникла новая идея: «Слишком много проектов не укладываются в сроки. Теперь, когда вы услышите крайнюю дату, мысленно вычитайте две недели. Тогда все проекты будут сделаны вовремя или даже раньше срока!» Менеджеры мысленно заворчали. Позже, один из них мне сказал в корридоре: «Логан никак не поймёт; это всего лишь ещё одна тупая идея, которая увеличит стресс, уменьшит качество продукта и не принесёт ни капельки пользы в плане соблюдения сроков.»

Игра по другим правилам

Действительно ли мир полон дурных боссов? Проекты полны некомпетентных работников? Технические ребята всегда зацикливаются на своих гиковских штучках в ущерб общему успеху проекта?

Конечно, некоторые люди могут быть довольно плохими, точно так же, как есть и супер-звёзды. В среднем же, большинство людей не будут исключительными личностями, о которых рассказывают легенды. Большинство менеджеров и членов проектных команд достаточно компетентны для выполнения хорошей работы, и у большинства вполне хорошие намерения; они делают всё в своих силах, чтобы успешно завершить проект.

Дело не в том, что «мы» умные, а «они» идиоты. Дело в том, что у разных людей разные убеждения, глубоко внутри, о том, что можно считать успехом проекта и как его достичь, а что считается неудачей проекта, и как этого избежать.

Если уж говорить прямо, то разработка приложений - это игра. В играх мы делаем ходы, основываясь на ходах других. Правила игры - это то, что мы принимаем за непреложную истину, которой руководствуемся в игре. В разработке приложений никто до конца не уверен, каковы же правила. В зарекомендовавших себя играх, вроде Монополии, шахмат и футбола, есть твёрдо установленные правила. В разработке их нет. Это относительно новая игра, и люди всё ещё пытаются понять, как в неё играть. В отсутствие чётких правил, люди стараются их угадать. В результате, разные люди играют в эту игру по разным правилам.

А откуда люди берут эти свои разные правила? Похоже, что ответ зависит от опытности человека.

Начинающие: Предписанные правила

Начинающие программисты находятся в сложной ситуации. Им не хватает опыта разработки, а потому они сталкиваются с огромной неопределённостью, даже когда работают над рутинными задачами. Чтобы преодолеть эту неопределённость, а заодно и расстройства, связанные с ней, они обращаются к более опытным разработчикам за предписанными правилами, которым они могут следовать буква в букву.

Многие из опытных людей с удовольствием делятся своими правилами, а наиболее активные из них группируют правила в некие евангелия и называют их ***методологиями***.

В качестве примера такой группы правил, названной методологией, давайте заглянем в книгу Unified Software Development Process (USDP), от Booch, Jacobson и Rumbaugh, опубликованную Addison Wesley в 1998. Вот некоторые случайно выбранные правила:

Страница 19: «При назначении ресурсов менеджер проекта должен минимизировать передачу артефактов из одних рук в другие так, чтобы ход процесса был максимально гладким.»

Страница 34: «Разработчики начинают работу со сбора требований заказчика в виде юзкейсов. Затем они анализируют и разрабатывают систему так, чтобы она соответствовала юзкейсам, создав сперва аналитическую модель, а потом и имплементацинную модель.»

Страница 76: «Архитектура создаётся архитектором заранее (во время фазы детализации требований). Это требует существенных предварительных затрат времени.»

В книге сотни таких правил, а сама книга - это всего лишь подмножество более полного Rational Unified Process (RUP), который доступен по подписке.

Есть и другие способы группировки правил, но наиболее распространены книги с предписанными правилами. Это обусловлено тем, что начинающие программисты образуют наибольшую аудиторию этих книг, а что-либо слишком философское будет чрезмерно абстрактным для них.

Разумеется, когда я только начал работу программиста, я считал, что толстые популярные книги куда более полезны, чем тонкие книжки, предлагающие только общие советы. В университете в восьмидесятых нам дали следующие правила для разработки прилоложений: нарисуйте эти конкретные диаграммы; заполните эти формы; пишите код, используя эти методики; отдайте дизайн этим людям для проверки; и т.д. Правила были такие чёткие и ясные, что следовать им было очень легко.

Мне было довольно неуютно, когда в начале моей карьеры один из старших разработчиков сказал: «Никто этого не делает в реальной жизни». Он не был особо хорошим ментором, ибо он не предложил никаких альтернативных правил кроме «мы должны соблюдать сроки» и «твой код должен работать». Я чувствовал себя потерянным в море.

Получается, что методики, нацеленные на начинающих, предписывают авторитарные правила, которым надо следовать, авторы которых обещают: «эти правила работали для меня, и если вы будете им следовать, как положено, они и для вас сработают».

Конечно, существует много конкурирующих методологий, каждая из которых предписывает разные правила, стараясь захватить доминирующую позицию. Наиболее заносчивые попытались предпринять войну методологий с маркетинговой уловкой: они называют свои правила ***best practices***. Их цель - уменьшение чувства неуютности новичков при адаптации этих правил и увеличение неуютности при адаптировании альтернатив, которые, разумеется, ***worse practices***, из-за которых проекты проваливаются.

Наиболее трусливые знают, что они немного обманывают людей, объявляя свои практики лучшими, а потому добавляют в своих книгах хитрый дисклеймер: «Адаптируйте согласно вашим специфическим нуждам проекта». Это помогает примерно так же, как инструкции по приготовлению на очень дорогих макаронах, купленных мной в Риме: «Варите до готовности». Автор всё ещё должен вам рассказать, что означает «готовность». Не удивительно, что я не раз слышал циничные заявления, что такие книги являются маркетинговым материалом консалтингового бизнеса авторов книг, помогающих запутавшимся людям разобраться.

Несмотря на заявления, ни одна методология ещё не стала *тем самым* набором правил для разработки приложений. Я видел, как некоторые консультанты отмахиваются от этого неудобства скользкими советами вроде: «Не существует идеальной методологии; вам нужно выбирать методологию, подходящую вашей организации, от проекта к проекту».

Хоть этот совет может удовлетворить кунсультанта, вряд ли он поможет клиентам. Просто нереально ожидать, что начинающие разработчики станут экспертами во всех существующих методологиях, а потом смогут выбрать правильную и адаптируют её к каждому проекту. Кроме того, как может кто-то определить лучшую методологию для конкретного проекта, пока с её помощью не попробуют углубиться в проект и раскрыть его специфические нужды?

Почти год я работал с лондонским подразделением компании Лоджика. Проектные команды должны были продемонстрировать, что они смогли выбрать методологию, подходящую к данному проекту. На практике же, команды не знали, да и не могли знать, достаточно о специфике проектов, а потому и не смогли выбрать методику в первую очередь. Вместо этого, я наблюдал, как многие команды работали, используя методики, которые они лично предпочитали. Они оправдывали свой выбор сделанными постфактум документами, основанными на уже раскрытых деталях проекта, которых они не могли знать вначале.

Учитывая очарование абсолютной уверенности, почти все из моих неопытных клиентов на каком-то этапе указывали, а потом и выбирали, одну методологию, чьи предопределённые правила можно было использовать с уверенностью в любое время. Некоторые выбирали чужие методики из купленной книги или побывав на курсах, тогда как другие полагались на правила, скопированные или спущенные им сверху.

Иногда я сталкиваюсь с неопытными командами, которые считают, что они не пользуются никакими методиками. На деле же это означает, что, хотя они и не следуют никакому набору правил, команда научилась следовать устоям, которые описывают правила, которым они следуют.

Например, Ричард и Мэл были начинающими программистами в английской компании EDP. Ричард и Мэл мне сказали: «Нам не нужны никакие вонючие правила; мы просто делаем проекты». Я в этом усомнился и стал слушать разговоры в команде в течение целого дня. Вот некоторые вещи, услышанные мной:

- Этот дурной босс хочет, чтобы мы это сделали сегодня.
- Пора уже остановить работу и устроить полную чистку кода.
- Я ещё не занимался производительностью, так как работаю над функционалом.
- Теперь наша очередь прийти к вам, ребята, и интегрировать наши последние изменения.

Хотя они и «просто делали проекты», они явно переняли (как оказалось, от более опытных в компании) многие обычаи, описывающие правила игры, которым они следовали без вопросов:

- Босс может быть плохим, но его приказов надо слушаться.
- Периодически замораживайте новую работу и фокусируйтесь на чистке кода.
- Стабилизируйте функционал, прежде чем переходить к производительности.
- Члены команды должны еженедельно встречаться, чтобы синхронизировать изменения.

Связавшись с некоторыми предписанными правилами начинающие стремятся принять их как абсолютный закон, а потому могут проявить сильную эмоциональную неприязнь к любым предложениям или попыткам изменить эти правила. Любой, кто нарушает правила, воспринимается как безответственный, ненадёжный человек, подвергающий проект опасности и заслуживающий наказания. В этом случае появляется очень реальная опасность, что методология станет догмой, которой следуют непрекословно, даже до степени, когда это становится разрушительно для проекта.

Мне довелось немного поработать с большой софтовой компанией, которая довольно быстро росла. Компания наняла довольно много неопытных новичков и навязала им очень строгую тяжеловесную методологию, чтобы следовали ей как закону. Это сразу же дало новичкам уверенность в работе, которую им предстояло выполнить, но мне кажется, что куда больше времени тратилось на следование правилам, чем на дизайн и разработку. Менеджерам проектов, да и другим сотрудникам, было поручено следить за людьми, нарушающими правила, что запустило волны депеш вверх и вниз по проектовой цепочке. Не удивительно, что уровень бюрократии обеспечил растяжение даже самых простых проектов на месяца, прежде чем они доходили до стадии работающих приложений. Но при этом компания свято верила в непогрешимость методологии.

Более опытные участники проектов давно поняли, что методология, как строгий костюм, - больше стесняла проекты, чем помогала им. Попытки обсудить это отвергались другими как непрактичная трата времени. Подпольные попытки работать более эффективно рассматривались руководством как нарушение субординации.

Сотрудникам проекта, которые не верили в навязанные правила, был дан выбор: либо сделать остальных в компании счастливыми, соблюдая правила, либо осчастливить клиента хорошими результатами. Инстинкт выживания заставил большинство сомневающихся следовать правилам.

Наибольшей проблемой на мой взгляд было то, что все знали методологию, но очень немногие её понимали. В частности, менеджеры проектов так фокусировались на соблюдении буквы закона, что не могли посмотреть на результаты со стороны. Без чёткой перспективы и более широкой картины они даже не могли спросить, есть ли вообще смысл работать таким образом?

Таким образом начинающие попадают в ловушку, где вера в строгие догматичные правила выбраной ими методологии ослепляет их, и они уже не видят существующих альтернатив и возможных улучшений. Как я объясняю многим своим клиентам - если вы не желаете рассматривать альтернативы и быть готовыми к улучшениям, будьте уверены - ваши конкуренты это обязательно сделают. Несмотря на это, начинающие больше боятся неуверенности, чем посредственности, и чувствуют непреодолимое желание следовать чётким правилам.

Большая американская компания - назовём её ИнтерСофт - купила задокументированную методологию у компании, которую назовём МетодВаре. МетодВаре была сертифицирована третьим уровнем CMM.

Компания ИнтерСофт была впечатлена этим уровнем сертификации; она узаконила методологию компании МетодВаре. Меня это малость передёрнуло. Я объяснил, что CMM нацелена на то, чтобы показать зрелость процессов в организации (то есть в МетодВаре), а не зрелость методологии, которую они продают.

SEI CMM определяет пять уровней зрелости. Первый - это вцелом неконтролируемый хаос. Второй - это когда некоторый контроль существует и применяется на некоторых проектах. На третьем уровне методология детально задокументирована и является обязательной во всей организации.

Компания ИнтерСофт считала, что третий уровень - это предел мечтаний, а не просто этап на длинном пути. Они были уверены, что они теперь используют лучшие практики, и отступ от них будет опасен. Но ведь после третьего уровня рекомендуется стремиться к

четвёртому, на котором для мониторинга эффективности методологии и получаемых приложений используются детальные измерения. В ИнтерСофте об этом не знали.

Наивысшим уровнем зрелости является пятый уровень, на котором наработки четвёртого уровня используются для постоянных улучшений методологии путём постоянного внедрения новых идей и измерения их эффективности. В ИнтерСофте этого слушать не захотели. Убеждённая, что купленная методика третьего уровня заключала в себе все лучшие практики, компания застряла на том уровне незрелости способностей, из которого уже не могла вырасти.

Тем не менее, приверженность конкретной предписанной методологии не гарантирует уверенности. Часто разражаются споры о том, что на практике означают конкретные правила. Что имеется ввиду в некоторых запутанных инструкциях? Если два правила конфликтуют, то какому отдать предпочтение? Можно ли изменить или даже отбросить какое-нибудь правило, если оно не подходит конкретным обстоятельствам, но продолжать считать себя сторонниками той же методологии? Частенько я замечал, что такие разговоры перерастают в горячие споры вокруг довольно простых вещей.

На одной конференции я встретил тимлида. Его команда использовала RUP и его унифицированный язык моделирования UML. Они использовали коллаборационные диаграмы UML как часть своей работы над дизайном, но заспорили о том, не являются ли последовательные UML диграмы лучше. Они беспокоились, правильный ли путь они выбрали, и спросили меня, не надо ли им переключиться.

Вместо того, чтобы начать подробное обсуждение небольших различий двух типов диаграмм, я спросил, а работает ли для них RUP вообще. Тимлид признался, что создание всяких диаграмм на самом деле замедляет проекты, а потому лучшие из его программистов старались использовать инструменты, которые генерировали диаграмы автоматически прямо из приложения, когда оно уже было написано.

Покопавшись в ситуации мы поняли, что главная выгода, почерпнутая из RUP, состояла в том, что ребята научились мыслить объектно-ориентировано, и тимлид был уверен, что это улучшило дизайн их приложений. Все остальные требования RUP были второстепенными на этом фоне, а его беспокойства о переключении с одного типа диаграмм на другой мало повлияют на результаты проекта.

Эта потребность в уверенности, хоть и оправдывает, скрывает более существенную проблему, с которой в конце концов сталкиваются начинающие. Они ожидают, что методология ответит на каждый вопрос, с каким они столкнутся. Когда это не срабатывает на практике, новички начинают сильно нервничать. Они вложили большую веру в выбраную методологию, потому что они верят, что её авторы знают больше их самих. Обнаружение дыр в предписаниях бросает тень сомнения на методологию и её авторов и подрывает веру новичков.

В прошлом я пытался объяснить начинающим программистам, что нет ничего зазорного в том, чтобы искать в методологии руководства, даже на довольно детализированном уровне, но нельзя ожидать, что она будет ответом на все вопросы. Игра «разработка приложений» существенно сложнее монополии, шахмат и футбола. В ответ на это я обычно вижу кивок головы, но я уверен, что люди редко воспринимали мою теорию. Чаще меня спрашивали: «Ну, и какую же методологию мы тогда должны использовать?» Начинающим трудно отбросить комфорт абсолютной уверенности.

Многие методологи очень восприимчивы к таким дырам, и, когда им их показывают, они стараются заполнить их в следующих версиях книг с правилами. В результате предписанные правила становятся ещё более сложными, а методологии со временем становятся более догматичными. Разумеется, это делает методологии менее субъективными, что может помочь разрешить некоторые споры, но создаёт опасность того, что методология станет менее гибкой и слишком громоздкой для применения. Кроме того, невозможно заткнуть все дыры, а потому, стремление к совершенству просто бесконечно.

В конце девяностых я провёл некоторое время с Дереком Колманом, одним из авторов популярной в то время методологии Fusion. Он признался, что текущая версия была на сотни страниц длиннее оригинала и содержала столько заковыристых правил, что он уже сам не мог увидеть лес за деревьями, а потому был захлёстнут сложностью собственной методологии. Дерек сильно беспокоился, что методология Fusion разваливалась под собственным весом, и решил не развивать её дальше.

Опытные: Статистические Значения

Итак, начинающие стараются вложить свою веру в предписанные правила других людей. Вера, разумеется, без доказательств. Начинающим нужна уверенность, а не доказательства, а потому они следуют правилам без вопросов, работают ли правила на практике. Возможно, это пройдёт для начинающих, но не для опытных людей.

Опытным нужно немного больше, чем вера. Им нужны доказательства. Если какое-то правило хорошо звучит, то опытный человек может взять его на вооружение как надо-проверить-на-деле правило, но проверка покажет, подтвердить или отвергнуть это правило. Постоянно несрабатывающие правила постепенно будут отвергнуты насовсем. А те, что подтверждаются на опыте, станут регулярно используемыми.

В результате постоянных испытаний опытные люди всё меньше полагаются на предписанные правила и больше на проверенные «законы». Они составляют свой собственный свод статистических значений, помогающий им решать, что работает, что нет и при каких обстоятельствах.

В книге Тома Демарко и Тимоти Листера Peopleware есть история забастовки под названием работай-по-правилам. В ней рабочие буквально следовали формально навязанным правилам (процедурным руководствам), и, разумеется, прогресс сильно застопорился. Знание, когда правилам следовать, а когда нет, очень важно для эффективной работы. Вот, где опыт играет главную роль.

Статистические значения ближе к сердцу, чем заимствованные правила, потому что они являются нашими собственными детищами, рождёнными потом наших трудов. Они становятся тем, о чём мы старательно печёмся. Мы используем их не только для управления нашими действиями, но и для определения успешности себя и других.

Так же как начинающие чувствуют эмоциональную несовметимость с методологиями, которые конфликтуют с заимствованными ими правилами, так и опытные сотрудники могут чувствовать сильную эмоциональную неприязнь по отношению к методологии, которая не соответствует их собственным статистическим значениям. Когда мы сталкиваемся с чем-то, что подходит под нашу статистику, то испытываем приятное чувство, закрепляющее нашу уверенность в наших данных. И наоборот, когда что-то

противоречит нашей статистике, у нас появляется некомфортное ощущение неприязни. Увиденное или услышанное *кажется* нам неправильным, а потому раздражает нас.

Конечно, вполне возможно, что опытные люди смогут составить из своих статистических данных методологию, пригодную для других, но это рискованное предприятие, так как начинающим не хватит опыта, чтобы воспринимать это не только как правила для слепого соблюдения. Эффективность статистических значений зависит не столько от абсолютного повиновения, сколько от здравомыслия и уверенности, основанных на личном опыте. Главная разница состоит в том, что предписанные правила - это нечто, что вы *считаете* правильным, а статистические данные - это то, что вы *знаете* как правильное глубоко внутри, потому что оно было не раз доказано на практике. Они отображают опыт, полученный на передовых, который и отличает опытных от начинающих.

Эксперты: Эволюционирующие Метафоры

Мы видели, что начинающие восновном полагаются на веру в заимствованные предписанные правила, принимаемые ими как факт, не требующий доказательств. Более опытные люди приняли правила, подтверждённые на практике, тем самым сформировав собственные статистические значения. Это помогает начинающим и опытным разработчикам справляться с рутинными аспектами проектов, где задачи можно решать способами, которыми уже раньше пользовались или они были запланированы. Но всё это не объясняет, как люди справляются с непредвиденными задачами по ходу проекта.

Вопрос того, как люди справляются с пробелами в предписанных правилах и статистических данных, интересовал меня уже несколько лет. Я работал со многими командами на разных проектах, и становилось ясно, что и начинающие и опытные люди часто заполняют пробелы, взывая к экспертам.

Сперва я предположил, что у экспертов просто-напросто более богатая подборка статистических значений. И я решил найти их, чтобы другие тоже могли ими пользоваться. Годами я общался с экспертами и пытался выманить их тайные познания. Но это оказалось не так-то просто.

Оказалось, что многим экспертам было очень сложно выразить их знания словами. Частенько они запутывались по ходу нашего общения, что в результате выливалось в сумбурные записи вроде: «Ну, главное правило в том, что программисты должны сами тестировать свой код, но, в то же время, заказчики должны помочь им в определении того, как должны выглядеть тесты, что на практике не всегда хорошо срабатывает, ибо заказчики не привыкли работать таким образом, а потому иногда разработчикам нужно писать свои тесты и просить заказчиков просмотреть их, но это опять же не всегда срабатывает, так как чаще всего заказчики не знают, как должны выглядеть тесты, пока не поиграют с рабочей системой, что в свою очередь чревато, потому что тесты на самом деле являются требованиями к приложениям, которые нельзя создать в отсутствие требований, но ведь иногда создание прототипа является единственным способом получения хоть каких-то ответов при том, что менеджмент может не разрешить создание прототипа, и даже если разрешит, то по прототипу заказчик может получить неправильное представление о якобы готовом приложении. Так что, всё зависит от ситуации.»

Что ж, статистика вроде этой не особо пригодна к использованию!

Мне показались странными сложности, с которыми эксперты пытались выразить свои познания. Я стал склоняться к мнению, что они справляются с непредвиденным как-то по-

другому. Я заподозрил, что они вообще не обращались к неким сложным статистическим наработкам, а делали что-то совершенно другое.

Ясно, что они не специально скрывали свои know-how, а скорее изобретали их на лету. Такой ответ внезапно пришёл мне в голову во время разговора с одним довольно высокопоставленным менеджером софтверовой компании. У парня не было никакого опыта программирования, на котором можно было бы базировать свои решения, но он был уверен в своей способности вести проекты. По мере разговора с ним становилось ясно, что его решения, как и решения других экспертов, действительно генерировались на лету. Ключом было его обращение к своему опыту из других проектов. Он занимался умственной гимнастикой, трансформируя все свои знания из одного типа бизнеса (менеджмент консалтинг) в другой (разработка приложений). Другими словами, он брал набор глубоко устоявшихся метафор и на их основе создавал правила и статистику, которыми руководствовался при ведении проекта.

Это, по моему мнению, и есть разница между экспертами и остальными: Эксперты эволюционируют метафоры, которые составили основу успеха их предыдущего опыта, и которые они постоянно используют для преобразования незнакомой среды в родную стихию. Метафоры используются экспертами, как факелы для освещения дороги.

Метафоры позволяют нам работать с незнакомой областью в терминах знакомой.

Наш дед спросил однажды моего брата, работавшего программистом, в чём заключается его работа. Тот попытался объяснить буквально, в технических терминах, но было ясно, что дед только запутывался. Тогда брат обратился к метафоре: «Программирование - это решение кроссвордов целый день».

И тут, словно включили свет, наш дед сразу же понял возможные сложности и начал долгое обсуждение того, как должны подходить друг другу все детали одного большого решения; иногда ключи оказываются зашифрованными; иногда это достаточно простые задачки; бывает, что случается разочарование из-за якобы правильной догадки; а случается и радость от того, что вспомнил ускользающее слово; и, разумеется, удовлетворение от наконец-то разрешённой головоломки.

Наш дед понимал, что объяснение моего брата не нужно воспринимать буквааально. Мой брат явно не сидел целый день с газетой и карандашом, решая кроссворды целый день. Он просто сказал, что вещи, присущие кроссвордам, также присущи и программированию.Он использовал метафору как связующее между непонятным (программирование) и чем-то хорошо знакомым нашему дедушке (кроссворды), а уже на этой основе мог продолжать разговор.

Таким образом получается, что метафоры - это живые существа, уменьшающие наш страх перед неизвестным. Они более активны, чем списки предписанных правил или статистических значений, так как могут меняться и адаптироваться к меняющимся обстоятельствам. Тогда как начинающие полагаются на предписанные правила и следуют дорогой, проложенной другими, опытные люди полагаются на собственные статистические значения, чтобы принимать решения, подходящие к ситуации, а эксперты используют эволюционирующие метафоры как генераторы новых правил и статистики, чтобы построить мосты к незнакомым ситуациям.

Я пересмотрел правила многих экспертов, которые не прижились на практике, и решил начать сначала, в этот раз стараясь найти метафоры, которые ими двигали.

Не удивительно, что я обнаружил существование огромного количества метафор, которыми пользуются эксперты. Что меня удивило, так это то, что совершенно не имело значения, в какой области человек был экспертом. Они всё равно руководствовались метафорами, взрощёнными годами. Я столкнулся с большим количеством людей, никогда не работавшими в софтовой индустрии, которые просто сменили индустрию, но продолжали опираться на метафоры, оказавшиеся успешными в предыдущих индустриях. Это придавало им уверенности во времена неясности.

Я познакомился с менеджерами со степенями MBA и финансовым опытом, с их метафорами, основанными на управлении стоимостями, что проявлялось на том, как они вели проекты. Я пересекался с людьми из отделов продаж, которые теперь ведут софтовые команды. Они оговаривали условия с командами, а потом придерживались их, как контрактов. Мне довелось поработать с пилотом спасательного вертолёта, руководящим теперь программистами, которые вытаскивают проекты из кризисов.

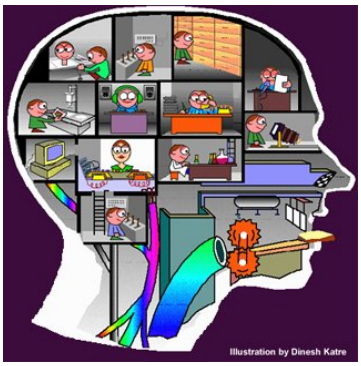
Конкретный пример: однажды я работал с Эн Верховен - директорм одного известного вебсайта. Раньше Эн работала ресторанным менеджером. Рестораны - это очень стрессовая среда, где обслуживание клиента жизненно важно. Я заметил, что при каждом возникавшем кризисе на проектах Эн сразу же вспоминала свою лучшую метафору: держи хаос на кухне (в нашем случае разработчиков), спрятанным от посетителей (заказчиков).

Когда сотрудники предлагали возможные решения кризисным ситуациям, она оставалась невозмутимой и спрашивала, получит ли заказчик выгоду от результатов этого решения. Радуйте заказчиков, и они будут приходить к вам снова и снова. Покажите плохой сервис и ненадёжный продукт, и они станут сотрудничать с кем-то другим, а вы вылетите в трубу.

Метафоры очень привлекательны в индустрии программных приложений, так как мы всё ещё пытаемся выяснить, каковы же правила в нашей игре. Метафоры позволяют нам опираться на более устоявшиеся области жизни, где правила игры более ясные, что позволяет нам заполнять пробелы, оставляемые неполными предписнными правилами, выложенными в методологических книгах. На самом деле, многие эксперты говорили мне, что, будучи вооружёнными надёжными метафорами, они могут отбросить все книги о методологиях, так как они инстинктивно знают, как справляться с любой ситуацией, независимо от новизны и неожиданности.

В следующих главах мы увидим, что каждая из трёх доминантных культур разработки приложний основывается на специфической метафоре, подкрепляющей эту культуру и усиливаемой самой культурой. Также мы увидим, что, если мы хотим изменить культуру, то сперва придётся помочь людям изменить метафоры, уменьшающие стресс, к которым люди прикипели и сильно на них полагаются.

Глава 3: Культурные Метафоры



Очаровательные значения

(от переводчика: вместо слова «значения» лучше всего подходит слово «смысл», но для правильности перевода пришлось бы полностью перефразировать несколько абзацев текста.)

Судя по многочисленным исследованиям, оказывается, что мы используем метафоры постоянно. Они настолько распространены, что мы даже не осознаём, что используем их.

Небольшая замечательная книга «Метафоры, по которым мы живём», написанная Lakoff и Johnson, показывает, как ещё с младенчества мы учим основные пространственные метафоры вроде вверх - это хорошо, вниз - это плохо. Мы держимся за них, исследуем и расширяем всю нашу жизнь: «Пора повысить уровень жизни!»; «Его шансы на повышение всё ниже и ниже!» Позже мы узнаём о других метафорах, где идеи получают направление: «Я вижу, куда вы направляетесь с этим!»; «Я просто не пойму, откуда взялся этот парень!». Потом о том, что время - это деньги: «Таким способом ты сэкономишь много времени»; «Мы инвестировали месяца в этот проект». И потом мы проводим остаток жизни, создавая новые более богатые метафоры, переплетённые самыми разными способами. Эти метафоры формируют наше мышление и мировоззрение, позволяя нам исследовать идеи и делиться ими с другими.

Основной принцип метафор состоит в том, что черты, присущие одному предмету, переносятся на другой. Эти черты называются значениями (смыслом) метафор. Исследование этих значений помогает нам понять что-то незнакомое в терминах чего-то, что мы уже хорошо понимаем.

Мой коллега услышал о новой книге о DDD (Domain Driven Design). Он подумал, что она может быть интересной, и спросил меня, о чём это. Я объяснил, что это означает достаточно тщательное исследование области бизнеса, чтобы построить её модель в программном виде. У меня не было возможности углубиться в более детальное объяснение, так как он сразу же понял эту метафору о построении модели и, следуя её значению, закатил мне длинную лекцию на тему DDD, о которой у него вдруг оказались глубокие познания!

Конечно, метафоры работают только когда есть соответствующее значение. Поэтому некоторые метафоры очевидны, а другие кажутся сложноватыми.

Когда мы впервые услышали разговоры о «дот ком пузыре», мы поняли, что именно имелось ввиду: пузыри быстро надуваются, лопаются, и все видят, что в них не было ничего, кроме воздуха.

Связь видна сразу, так как значение очевидно. Метафоры работают замечательно.

Аналогично, если бы кто-то говорил о «золотой лихорадке дот ком», то мы бы сразу поняли значение - толпы людей надеются на высокие доходы от спекулятивных цен, но только единицы реально зарабатывают, а остальные остаются разочарованными.

Но если бы кто-то сказал «дот ком гамбургер», то большинство из нас вряд ли бы преуспели в связывании значения гамбургера и того, что случилось с дот комами. Гамбургеры - это довольно слабая метафора для дот комов.

Разве что вы работаете в индустрии гамбургеров и можете найти некую связь.

В этом состоит критический аспект метафор. Разные люди могут интерпретировать одну и ту же метафору по-разному, так как их различающийся жизненный опыт влияет на то, как они воспринимают метафоры и делают о них выводы. Как мы увидим дальше, эта открытость личным интерпретациям делает метафоры такими полезными. Однако, это также означает, что одни люди воспринимают метафоры в положительном ключе, а другим они не нравятся.

Многие программисты с пониманием улыбаются, когда кто-то говорит: «Мой менеджер - тупой босс Дилберта», но их менеджерам эта фраза вряд ли понравится (если только они не думают о своём боссе точно так же!). Чтобы метафора сработала, её значение должно иметь смысл конкретно для того, к кому она обращена.

Если значение метафоры нам понятно, то мы можем развивать её сколько угодно. А если метафоры нам не нравятся, то мы наверняка будем пытаться найти другие, более подходящие.

Я сказал коллеге метафору своего брата, что программирование - это решение кроссвордов. Эта метафора сработала для моего деда, но коллега был шокирован и возмущён. «Это не так!» - сказал он, - «У кроссвордов всего один правильный ответ, а в программировании их несколько». Метафора, так хорошо воспринятая дедушкой, не сработала для коллеги.

Мой коллега предложил другую метафору: «Программирование - это игра для молодых людей.»

Эта метафора, возможно, очень хороша для молодых людей вроде него самого, а дедушка и ухом бы не повёл.

Столкновения Культур

Удобные, понятные метафоры помогают людям меньше беспокоиться при поиске решения проблем, с которыми они сталкиваются в течение дня. Люди, которым подходят одни и те же метафоры, разделяют общий взгляд на вещи и решают похожие задачи похожими способами. Эти люди будут понимать друг друга с полуслова и притягиваться друг к другу. А те, кто не разделяет их взглядов, будут смотреть на мир иначе, с трудом вписываться в коллектив, и, скорее всего, не задержатся в нём надолго.

Компания СамоСервис попросила меня помочь в построении одной из их команд. На тот момент было пять вакансий. Мы получили 585 заявок на эти места. Из них отобрали двадцать финалистов, которые казались очень способными.

С ними побеседовали потенциальные будущие сотрудники и менеджеры. В итоге наняли только двоих. Двое получили отказы, так как явно приукрасили свои способности в резюме. А 16 были отвергнуты из-за личностных нестыковок: «Он просто не вписывается в команду»; «Я не могу представить себя, работающим с ней»; «Он тут никому не понравится».

Эти личностные нестыковки свелись к несовместимым мировоззрениям. Найм любого из тех шестнадцати было бы сродни впуску лисы в курятник. Или курицы в стаю лис.

Превалирующее мировоззрение формирует культуру группы людей или организации. Общее определение культуры: «то, как мы тут работаем», но это фокусирует внимание на том, что люди делают, а не на том, почему они это делают вообще. Люди могут менять работу довольно часто, но это не означает, что они меняют свою культуру. Возможно, более подойдёт определение культуры: «как мы видим мир вокруг себя». Это наше мировоззрение, формирующее наш взгляд на вещи, а с помощью используемых метафор определяющее не только то, как мы работаем, но и то, как мы мыслим и с какого боку подходим к решению ежедневных проблем.

Столкновения культур происходят, когда мировоззрения, и, соответственно, метафоры, не стыкуются. Часто мы полагаем, что люди как-нибудь адаптируются и сработаются. Люди не могут просто решить сработаться - метафоры, ведущие их по жизни, означают, что либо они инстинктивно вольются в коллектив, либо они будут бороться с культурой и будут вытеснены из компании, или же они будут выполнять ненавистную работу стиснув зубы, пока это их не достанет настолько, что они уйдут сами.

В одной компании в течение пары лет никак не кончались проблемы с проектами. Руководство надеялось, что найм лучших работников решит их неурядицы. Они анонсировали, что теперь будут нанимать только «самых лучших». Под этим они понимали людей с большим опытом в современных технологиях. Были предложены высокие зарплаты, озвучены красивые обещания, и несколько новых человек появились в компании.

Первые несколько дней были довольно интересными, но вскоре дела покатились к чертям. Вновь нанятым сотрудникам поручили хорошо работать, но не разрешили вноситть никаких изменений. Руководство считало, что внесение изменений в существующие процессы и правила неприемлемо.

Я хорошо знаю кое-кого, кто лично пережил столкновение культур. После многомесячного процесса найма он уволился в первые несколько недель работы. Руководство не хотело его отпускать и предложило выбрать почти любую позицию в компании. Поискав с неделю он сказал: «Тут нет места, где бы я смог что-то изменить к лучшему. Позиции варьируются, а культура нет.» - и ушёл.

Начальники надеялись, что простого увеличения мозговой силы будет достаточно. Вновь нанятым сотрудникам пришлось встроиться в конфликтующую культуру, что убило в зародыше те улучшения, которые они могли привнести. Они не могли работать своими проверенными способами и были вынуждены использовать методики, которые по их мнению были причинами неудач.Большинств из них уволилось в первые несколько недель. Некоторые приспособились: им пришлось наречь себя наёмниками; они делали ненавистную работу за большие деньги.

«Самые лучшие» в итоге ничего компании не дали. Мораль сей басни такова: не нанимайте людей, чтобы они что-то изменили, если вы не готовы изменить культуру.

Эмоционально Заряженные Метафоры

Позвольте мне остановиться на минутку и предупредить.

Поскольку метафоры могут быть очень убедительны, нужно быть осторжными в их выборе и использовании, чтобы не переусердствовать. Некоторые метафоры изначально заряжены эмоциями, а потому при неправильном использовании могут оттолкнуть людей от конкретных методологий или культур.

Например, я столкнулся с тем, что некоторые эксперты пользуются метафорами с сильным эмоциональным зарядом при обсуждении своих культур со мной. Со временем я понял, что это даже не их метафоры; они скорее игрались провокационными метафорами, чтобы убедить меня в преимуществах их культуры над остальными.

Я не могу винить их за это, поскольку почти все стараются обратить других в свою веру, но поначалу это меня сильно отвлекало от обнаружения метафор, реально используемых программистами.

Я стал осторожнее относиться к эмоционально заряженным метафорам и старался более тщательно проверять, используют ли их люди на практике или просто используют их для манипулирования мнением других.

Негативно Заряженные Метафоры

Разумеется, существует множество метафор, которые люди используют или могут использовать для усиления отрицательного мнения, часто по отношению к культурам и методологиям, которые они не очень хорошо знают. Такие метафоры могут быть довольно заразными и убедительными. Например, описание какой-нибудь методологии как Чёрной Дыры возбудит только отрицательные эмоции. Смысл таких метафор не помогает людям работать над софтовыми проектами и их заковыристыми проблеммами. Это простое эмоциональное манипулирование. Нужно иметь такие вещи в виду и уметь защищаться от них.

Я слышал, как один человек поносил методилогию, называя её подходом 1984 года к разработке приложений, намекая на ужасы одноимённой книги Джорджа Оруэлла. Я знаю людей, которые стали побаиваться довольно популярной методологии, когда слышали, что её называют Лабораторией Безумного Учёного, ассоциируя методику с работой лунатиков, преследующих собственные опасные желания.

Положительно Заряженные Метафоры

Конечно же, тот же фокус можно проделать и с целью привлечения людей на свою сторону.

Я слышал, как один менеджер по продажам постоянно называл методологии и инструменты, которые он продавал, Последним Словом Техники - что не шибко много говорит вам о способе применения на проектах, но вполне может дать клиенту приятные ощущения комфорта. А один из моих бывших начальников называл свою любимую довольно устаревшую методологию Практиками, Проверенными Годами, при этом клеймя новую многообещающую методологию Дикой Дорогой к Хаосу. Нетрудно догадаться, как регировало руководство, когда им предлагали выбрать между Практиками, Проверенными Годами, и Дикой Дорогой к Хаосу.

Нейтральные Метафоры

Когда мы видим, что люди используют метафоры, нужно пытаться понять, есть ли в них эмоциональный заряд. Если да, то метафоры скорее всего будут иметь довольно одностороннее влияние на принятие решений говорящего и слушающего.

При подготовке к написанию этой книги, когда люди описывали мне свои взгляды на разработку приложений, я тщательно следил за эмоционально заряженным языком и метафорами, наполнявшими его, всегда стараясь вернуть разговор назад в эмоционально нейтральное русло. Надеюсь, что я в этом преуспел, метафоры, представленные далее о разных методологиях разработки приложений, не несут в себе ни положительного ни отрицательного заряда.

Разумеется, индивидуальная эмоциональная реакция людей на ту или иную метафору может быть как позитивной, так и негативной - что и притягивает их к той или иной культуре - но я надеюсь, что эти реакции идут от их личной интерпретации, а не от эмоциональной нагрузки самих метафор.

Когда я впервые услышал о методологиях, к которым тянутся люди Сервисной Культуры, их восновном называли Легковесными методологиями. Те, кто придумал эту метафору, надеялись, что люди поймут подразумеваемое отсутствие бюрократии. И тем не менее, многие ассоциировали Легковесность с ненадёжностью, а потому решали, что нет смысла их рассматривать. Методололгии быстро переименовали в Проворные (Agile).

К сожалению, как я заметил, слово Проворные тоже не без изъяна. Эта метафора наводит некоторых людей на образы о проектах, завершающихся очень быстро. Этот факт зацикливает их на скорости и желании ещё больше ускориться. И хотя эти методологии действительно могут помочь ускорить проекты, это не является их главной идеей. Их соль (по крайней мере из моего опыта) состоит в непрерывном слежении за результатами и удовлетворении меняющихся нужд заказчиков.

Вот почему я предпочитаю название Сервисная Культура в качестве метафоры для этих методологий. Я обнаружил, что эта метафора заставляет новичков исследовать ассоциации, в результате чего им приходят на ум образы, более соответствующие образам людей, которые уже используют «Проворные» методологии.

Обнаружение Культурных Метафор

Изучив разные компании и тесно поработав с людьми, чтобы обнаружить метафоры, которыми они пользуются каждый день, я обнаружил некоторые тенденции. Существует три чётко доминирующих мировоззрения, каждое из которых подпитывается кучей метафор, и, как результат, отображает разные культуры. Три культуры разработки, с которыми я уже несколько раз столкнулся, были упомянуты в предыдущих главах:

- Заводская Культура
- Дизайнерская Культура
- Сервисная Культура

Разумеется, этоне единственные существующие культуры разработки программных приложений. Просто они наиболее популярны среди всех. Когда и если другие культуры возьмут верх, то я опубликую новое издание этой книги, включив их в список.

Разумеется, с появлением новых культур, доминирующие культуры могут потерять свою популярность. В пятидесятых годах, например, ни одна из этих трёх культур не была

распространена. Доминирующей была Научная Культура. Она подразумевала, что пррограммисты должны носить белые халаты. Нельзя было подпускать простых смертных к компьютерам - это была работа для учёных. Программирование было чистым применением математики и использованием научных принципов для получения точных предсказуемых результатов.

К сожалению, оказалось, что программирование менее наукоёмко, чем полагали люди. Проекты в руках учёных вылезали за рамки бюджета и требовали слишком много времени. Полученные программы часто не имели всего требуемого функционала, а качество и надёжность оставляли желать лучшего. Правила этой игры, судя по всему, не очень хорошо работали. Это пошатнуло уверенность в Научной Культуре, и многие люди, в частности промышленники, стали от неё отказываться.

Тем не менее, культурные привязанности могут быть довольно сильными, и даже сегодня некоторые люди, несмотря на все доказательства, цепляются за Научную Культуру и во всех неудачах обвиняют недостаточно строгое следование правилам, лежащим в её основе. Если кому интересно, то Научная Культура более подробно описана в Приложении А, а тут мы её более обсуждать не будем.

Каждая из трёх культур разработки приложений подкреплена очень разными метафорами, которые влияют на мнение и восприятие людей. Даже просто сами названия уже являются звучными метафорами.

Например, подумайте о Заводе, и, вероятно, куча всяких мыслей возникнет в голове. Может вы представите высококвалифицированных рабочих, управляющих сложными роботами в молчаливой рациональности, создавая высокотехнологичные компоненты.

Подумайте о Дизайне, и, возможно, вы представите талантливых людей, в творческой тишине рассматривающих различные подходы к решению проблеммы, чтобы получить один элегантный дизайн.

Подумайте о Сервисе, и, вероятно, вы увидите профессионалов, тесно сотрудничающих с клиентами, чтобы выявить и удовлетворить их конкретные нужды.

А может и нет.

Может, Завод ассоциируется у вас с шумными фабриками, летящими искрами, потными работниками, подающими сырой материал сложным станкам, которые далают из них некую продукцию, вываливаемую в кучи на пол.

Возможно, Дизайн напомнит вам о шумных мозговых штурмах с длительными жаркими обсуждениями всяких странных и диких предложений.

А Сервис может заставить вас скривиться от образов хамящих продавцов, работающих за мизерную плату.

Культуры, имеющие один набор ассоциаций для кого-то, могут иметь совершенно другой набор для другого.

Доминирующие Метафоры

Я обнаружил, что кроме самого названия каждая из трёх культур подкреплена ещё и одной доминирующей метафорой:

- **Заводская Культура**: Разработка приложений - это заводской конвейер
- **Дизайнерская Культура**: Разработка приложений - это архитектурный дизайн
- **Сервисная Культура**: Разработка приложений - это удовлетворение заказчика

Каждая из трёх доминирующих метафор окрашивает наше суждение и направляет наши линии мысли довольно эмоциональным способом.

Всё, что совпадает с доминирующей метафорой предпочитаемой нами культуры, будет нами сразу же принять. Всё, что противоречит, будет чаще всего отвергнуто, как не вызывающее доверия. При разговоре о разработке приложений с кем-то, чья доминирующая метафора отличается от вашей, почти всё, что вы скажете друг другу, будет конфликтовать. Скорее всего, каждый из вас решит, что другой немного не дружит с головой.

Один знакомый из отдела продаж рассматривал разработку приложений как жидкость. Об одном проекте он сказал «Мы будем доить эту корову годами», а о другом «Компания должна вынуть пробку в этом проекте.» Я было подумал, что столкнулся с ещё одной метафорой о разработке программных приложений, но потом до меня дошло, что он говорил не о проектах, а о деньгах, заработанных или потраченных проектами. Он говорил о жидком капитале. (*Прим.: Игра слов. В англ. языке liquid - означает ликвидный и жидкий.*)

Развитие доминирующих метафор культур помогает приверженцам данной культуры сформировать чёткое мнение о том, что же такое разработка программных приложений. Последние несколько лет я тесно сотрудничал с представителями всех трёх доминирующих культур и просил их вкратце представить их мнение по отношению к доминирующим метафорам предпочитаемых ими культур. Ниже я попытался достаточно правдиво описать оттенки тенденций в их представлениях:

- **Заводская Культура**: Разработка приложений - это заводской конвейер

Заводы уже оптимизировали наиболее рациональный способ производства, то есть использование конвейеров. Поскольку программы это тоже продукция, то и производить их наиболее рационально на (виртуальном) конвейере.

Слаженная работа конвейера требует предварительного планирования со стороны руководства, чтобы избежать неприятных сюрпризов вроде выхода за рамки бюджета или низкого качества продукции. Требования к приложениям до последней детали должны быть представлены в этом плане, чтобы у работников были чёткие спецификации продукта, который они собираются производить. Отдел технического контроля будет использовать те же самые спецификации для измерения качества продукции в конце производственного цикла.

Рабочие последовательно закрепляются на конкретных местах вдоль конвейера. Рабочий на каждом месте отвечает за конкретные строго определённые операции согласно описанию в плане, например, документирование требований, детальный дизайн, программирование, тестирование, и т.д. Каждое такое задание должно быть завершено в чётко запланированное время, что в совокупности даёт общее расписание целого производственного плана.

Каждый рабочий конвейера получает результат работы сотрудника перед ним, делает своё дело согласно плану и передаёт результат следующему рабочему конвейера для следующей стадии. Законченное программное приложение сходит с конвейера в рамках отведённого бюджета, времени, и согласно спецификациям.

Менеджеры должны тщательно следить за работой сотрудников, чтобы те следовали плану, и поправлять или наказывать тех, кто этого не делает, поскольку отклонение от плана является самой большой угрозой успеху проекта.

- **Дизайнерская Культура**: Разработка приложений - это архитектурный дизайн

Архитектура - это концептуальная основа, на которой строится продукт. Наиболее удачные архитектуры создаются с рассчётом на возможные изменения желаний клиента, а потому имеют некоторую встроенную гибкость.

Такие архитектуры создаются лучшими архитекторами, являющимися экспертами в выявлении требований к продуктам и в создании масштабируемых моделей для исследования неясностей, для решения заковыристых задач, для экспериментирования с удачными решениями, и для подготовки к грядущим измнениям. Во время разработки дизайна архитектор определяет точки архитектурной стабильности и встраивает места, где можно будет делать изменения в будущем. Как только дизайн стабилизируется, можно с уверенностью строить оставшуюся часть продукта.

Программы - это приложения, разработанные для поддержки бизнес процессов клиента. Дизайны, не предполагающие будущие изменения этих процессов, довольно хрупки. Они ломаются под давлением изменений и быстро теряют ценность для бизнеса.

С другой стороны, гибкое программное обеспечение построено на основе архитектуры, разработанной с мыслью о изменениях в голове. Такие архитектуры предвидят будущие изменения бизнес процессов за счёт встроенных изменяемых модулей. Это позволляет обеспечить адаптируемость программных приложений, что сохраняет их ценность длдя пользователей неизменной.

- **Сервисная Культура**: Разработка приложений - это удовлетворение заказчика

Клиенты хотят вести с нами дела, если мы помогаем им в удовлетворении их самых критичных желаний. Это означает фокусироваание больше на *их* успехе, а не на своём личном.

Сегодня компании находятся в условиях постоянной рыночной конкуренции. Чтобы выжить и преуспеть, им приходится отвечать на неожиданные ходы конкурентов с инновациями и находить новые возможности везде, где только можно. А потому, успех требует постоянного адаптирования бизнес процессов, порой совершенно непредсказуемого. Поскольку бизнес процессы со временем меняются, то и поддерживающее их программное обеспечение должно меняться тоже. Программы должны оставаться гибкими, а дизайн должен постоянно адаптироваться в ответ на эти непредвиденные изменения.

Заказчику должно быть предоставлено центральное место в идентификации их меняющихся деловых приоритетов и в определении, какой функционал должен быть разработан и в каком порядке. Ранняя и частая поставка готовой работы позволяет заказчикам использовать функционал, пока он ещё представляет ценность для бизнеса.

Также это позволяет им менять приоритеты следующего запланированного функционала и останавливать разработку, когда стоимость начинает превышать ожидаемый доход.

Таким образом, удовлетворение заказчика означает смещение нашего фокуса в разработке приложений от собственных желаний сделать стабильный дизайн к обслуживанию потребности клиента в наличии программ, которые всегда поддерживают их меняющиеся бизнес процессы. Тогда мы становимся реальным капиталом в глазах заказчика, а не просто статьёй расходов.

Следственные Метафоры

Доминирующие метафоры каждой культуры помогают людям сформировать общее мировоззрение, разделяемое приверженцами этой культуры. При наличии этого мировоззрения они могут развивать метафоры так, чтобы охватить все виды вопросов, включая довольно фундаментальные аспекты разработки приложений. Наиболее значимые из них:

- Что такое программные приложения, дизайн и требования
- Роли менеджеров и команд и их отношения с заказчиками
- Риски, которые могут повлиять на исход проекта, а в частности содержание, время, качество и деньги
- Методы контроля, которые определяют, что является прогрессом, а что тратой средств, и должен ли данный проект считаться успешным или нет

Имея это в виду вы можете играть в словесные ассоциации, называя слова одно за другим и предлагая назвать первое, что придёт в голову. Например, скажите кому-нибудь «Успех это…» и подождите ответа. Вы обнаружите, что их инстинктивная реакция почти неизменно совпадает со значением доминирующей метафоры их культуры.

Разумеется, разные люди одной и той же культуры дадут разные ответы. Например, когда я сказал «Деньги это…» людям заводской культуры, то их ответы сильно разнились. Одни сказали «власть», другие «прибыль», третьи «затраты», и т.д. Мне было сложно найти связь между ответами, но со временем я обобщил и структурировал их как можно лучше. Таблица, приведённая ниже, это моя скромная попытка резюмирования инстинктивных реакций, которых придерживаются люди разных культур.

Культуры	Заводская	Дизайнерская	Сервисная
Доминирующие Метафоры			
Разработка приложений - это	Конвейер	Дизайн Архитектуры	Удовлетворение клиента
Аспекты продукции			
Программы	Жёсткие	Гибкие	Мягкие
Дизайн - это	Проект	Техническая деятельность	Людская деятельность
Требования	Записываются	Моделируются	Делятся
Аспекты людей			
Менеджеры - это	Командиры и Надзиратели	Лидеры	Референты
Команды - это	Рабочие	Эксперты	Сотрудники
Заказчики - это	Покупатели	Деловые люди	Короли
Аспекты рисков			

Содержание	Фиксировано	Уточняемо	Вечно меняется
Время - это	Деньги	Качество	Изменения
Качество	Проверяется	Это Стабильность	Это приёмка клиентом
Деньги - это	Стоимость	Производительность	Незначимое
Аспекты Контроля			
Успех - это	Соответствие нормам	Продукция с правильным дизайном	Счастливые Заказчики
Неудача	Наказуема	Код-Спагетти	Выявляется рано
Растрата	Небрежность	Бюрократия	Низкая ценность
Прогресс - это	Завершение задачи	Улучшение дизайна	Частые поставки

Что особенно интересно в этой таблице, так это то, что многие из перечисленных значений сами являются метафорами. Это следственные метафоры доминирующих метафор культур. Когда кто-то говорит «Программы жёсткие», или «Деньги - это Производительность», или «Время - это Изменение», то совсем не нужно это воспринимать буквально. Даже те, которые поначалу кажутся буквальными, вроде «Качество проверяется», «Дизайн - это техническое действие», и «Неудача выявляется рано», тем не менее, являются метафоричными для представителей соответствующих культур, поскольку каждая из них запускает мысленную цепочку ассоциаций, приводящих к сути вопроса.

Таблица находится на той стадии, на которой я могу сказать, что после многих итераций она представляет аккуратное отражение значений каждой из трёх культур. Или если быть более точным - аккуратное отражение взглядов многих людей, с которыми я работал, и которые были погружены в эти культуры. Таким образом, она выиграла не от моих личных усилий, а, что более важно, от периодических вливов многих людей из каждой культуры. Им я очень признателен.

Поэтому, если вы были погружены в одну из культур с головой, и вам кажется, что я сильно ошибаюсь в выводах касательно вашей культуры, то, пожалуйста, напишите мне, чтобы я мог исправить ошибки в будущих версиях этой книги.

Доминирующая метафора каждой культуры предоставляет контекст, в котором определяются следственные метафоры. Фраза «Растрата - это небрежность» базируется на контексте метафоры Заводской культуры «Заводской конвейер», которая помогает понять, что именно понимаетсяс под небрежностью. Например, неподчинение приказам, лишь частичное завершение заданий, опечатки в документации, и т.д. Дизайнерская культура, в свою очередь, рассматривает небрежность по-другому (например, негибкий дизайн архитектуры), как и Сервисная культура (например, недостаточная забота об удовлетворении клиента).

В начале 2008 года я работал с компанией, которая хотела предложить пользователям более богатые возможности поиска по своей базе данных. Команда, назначенная на этот проект, считала, что бессмысленно тратить время на чёткий дизайн, пока не было выяснено, что же важно для клиентов. Поэтому они сфокусировались на определении реалистичных пользовательских сценариев.

Две недели спустя они сделали презентацию иполнительному вице президенту. Мне повезло присутствовать на этой презентации. Каждый раз, когда они представляли очередной сценарий, вице президент встревал с вопросами о деталях пользовательского интерфейса: «Будут ли результаты поиска слева или справа?», «Какую цветовую схему

будете использовать на окошках критериев поиска?», «Это будет квадратная или круглая кнопка?»

Команда обхъяснила, что они работали над определением наиболее ценных требований, а не на трате времени на графический дизайн, о котором ещё рано говорить. Вице президент выглядел рассержено. Он оборвал их объяснения и сказал, что ожидал увидеть очень детальный план, а не трату времени на посредственные вещи. Им дали ещё одну неделю на создание плана проекта, включающего расписание, подробный список задач и графический дизайн. Они должны были вернуться и показать то, что «должно было быть сделано уже сегодня!».

Понятно, что команда ушла, ворча о том, как им придётся потерять неделю на эту презентацию, а вице президент ушёл расстроенный, что команда потряла две недели.

Метафоры и Методологии

Итак, реальное богатство метафор происходит не столько от их значения, сколько от личных способов итерпретации в непредвиденных обстоятельствах и на разных уровнях сложности. Так доминирующая метафора может расти с нами по мере того как мы нарабатываем опыт, развивая следственные метафоры в разных новых ситуациях.

Процесс развития смысловых значений неизменно ведёт к новым внутренним метафорам, сплетая вместе новые концепции, новые словари и новые соотношения, котрые сами по себе тоже подвержены постоянным личным рекурсивным толкованиям. Эти постоянные открытия являются «Ага!» моментами, ибо метафоры ведут нас к свежим интересным выводам.

Стоя возле доски я помог менеджеру проектов сформулировать и нарисовать мысленную карту его собственных метафор об управлении софтовыми проектами. На довольно ранней стадии он размышлял об управлении рисками проекта и сказал «самый большой риск - это то, что заказчик не захочет этого». Это заставило его задуматься над приоритетами заказчика и переосмыслить список требований заказчика.

Вместо того, чтобы представить его как длинный список, он подумал о бутылке молока, где сливки всплывают на поверхность. Он говорил о снятии сливок. Было видно как новые метафоры развиваются в его голове, и приятно наблюдать за его возбуждением от того, что он сам до всего этого дошёл.

По мере накопления опыта с метафорами мы формируем целое семейство следствий, где каждый представитель семейства вдохновлён и адаптирован к конкретным обстоятельствам новых проектов, с которыми мы сталкиваемся. Каждая решающая точка может начать новую ветвь семейного дерева, а каждая новая находка может развить семейное дерево вширь или ввысь, развивая и обогащая представителей семейства.

Когда мы перестаём развивать метафоры, дерево умирает. Кстати, это неплохое определение методологии, описанной книгой с правилами: это место для успокоения разбросанных костей мёртвых метафор, когда-то развитых их автором. Потеря подкрепляющих метафор, на основе которых были составлены эти правила, лишает нас богатой коллекции смысловых интерпретаций, которые можно было бы развивать дальше. Мы понятия не имеем, как прикрепить плоть к костям. Нам остаётся лишь слепо следовать предписанным правилам методологии, которая есть ничто более, чем след замороженных интерпретаций автора.

Компания, публикующая статистику, наняла меня на два месяца в помощь старшему архитектору приложений, чтобы помочь с определённым замысловатым дизайном. Изучив требования я открыл текстовый редактор и начал набирать некий код: небольшой тест, проверяющий одно из требований. Архитектор предостерёг меня, что если начальник отдела меня увидит за этим делом, то мне сделают суровый выговор. Но ведь настоящие архитекторы зданий создают мини-копии конструкций и тестируют их, попробовал оправдаться я. Но он сказал, что к нам это не имеет отношения. Методологическая книга говорит, что архитекторам нельзя писать код; они рисуют высокоуровневый дизайн, который затем передаётся старшим разработчикам, которые пишут детальный дизайн, которые в свою очередь дорабатываются младшими разработчиками. Эта буквальная интерпретация их методологии показалась мне ненормальной, поскольку архитекторы не имели возможности убедиться в надёжности их дизайна.

Чтобы методология оставалась адаптируемой к меняющимся обстоятельствам, основывающие её метафоры должны оставаться очевидными. В таком случае люди смогут развивать их дальше, когда и где это необходимо. Тогда такие методологии остаются живыми, а не умирают.

Именно так эксперты обращаются с книжными методологиями, согласно моему исследованию. Несколко экспертов сказали мне, что пройдясь однажды по правилам они понимали суть методологии. Они осознавали, в каком направлении движется методология, а потому могли отбросить книгу с правилами, поскольку больше в ней не нуждались.

Конечно же, этот инстинкт происходил не от запоминания длинных списков правил. Скорее они могли абстрагироваться от подробных правил и сформировать вписывающиеся в методологию метафоры, которые помогали им самостоятельно прочувствовать недостающие правила.

Методология, выраженная в терминах её метафор, открыта дальнейшему интерпретированию. Вместо того, чтобы быть костьми в могиле, методология формирует скелет - концептуальный каркас - который со временем обрастает плотью. Создатель методологии может начать с единственной очень буквальной интерпретации в качестве первого набора правил. Важно то, что метафоры говорят нам, как справляться с пропусками в этих правилах. Нам не нужно тщетно искать конкретных ответов, которых нет в книгах с правилами. Вместо этого мы можем призвать на помощь основную метафору методологии.

Глава 4: Заводская Культура

Метафора о Заводском Конвейере

Управление проектами разработки приложений может быть довольно пугающим бизнесом. Всё что угодно может пойти не так. Слишком много проектов превысили бюджет, не уложились в сроки или произвели нечто низкого качества, что никому особо и не нужно. Это делает менеджеров проектов дёрганными.

Другие индустрии сталкивались с похожими проблемами в прошлом и выработали способы их преодоления. В частности, массовое производство было примером, предложив конвейерные линии в качестве наиболее рационального подхода к обеспечению регулярных, предсказуемых, своевременных поставок согласно оговорённой цене и качеству.

Принципы массового производства на конвейерных лентах были переняты и другими индустриями, где их изложили как основные принципы профессионального управления проектами, и они доказали свою высокую эффективность на деле.

Соответственно, не стоит удивляться, что многие люди, вовлечённые в разработку приложений, соблазнились Заводской Культурой и её доминирующей метафорой о Конвейерной Ленте и пользовались ею для приручения своих диких проектов.

Шоколадная Фабрика

Представьте себе чистую ухоженную фабрику, надёжно и рационально производящую коробки с шоколадными конфетами, наполненными апельсиновой начинкой. Обратите внимание на конвейер с конфетами, лежащими на его ленте в разных стадиях звершённости. Посмотрите на станки, расположенные в различных местах вдоль конвейера, и на рабочих возле них.

Теперь запустите конвейер как кино. Добавьте цвет, запахи и звуки, если получится. Заметьте, что получается, когда оператор первого станка нажимает кнопку и опускает рычаг. Станок создаёт пустые шоколадные раковины и выкладывает их на конвейерной ленте. Шоколадные раковины едут по ленте и останавливаются возле следующего станка. Оператор этого станка поворачивает рукоятку, и каждая шоколадная ракушка наполняется в точности необходимым количеством апельсиновой начинки. Шоколадки едут дальше по конвейеру и останавливаются по мановению руки оператора следующего станка, который кладёт орех на макушку каждой конфеты. Затем они едут дальше. Оператор нажимает на педаль, и последний станок ковейера берёт по десять конфет одновременно, кладёт их ровными рядами в коробки и составляет коробки в штабеля для транспортировки.

Теперь представьте, что вы менеджер такого заводского производства. Уявите себя проходящим вдоль конвейера с блакнотом в руках. Вы чувствуете удовлетворение от того, как гладко, слаженно и циклично работает конвейер. Час за часом, день за днём. Всё работает как часы.

Оппа! Внимание! Один из станков конвейера сломался. Полуготовые конфеты скапливаются на ленте. Рабочие паникуют: жидкий шоколад, апельсиновая начинка и

шоколадные раковины разливаются по конвейерной ленте на заводской пол. Ваше сердцебиение ускоряется. Вы чувствуете начало паники. Это ваша ответственность! Вы должны разобраться с поломкой! Что вы собираетесь делать? Ну же, нам нужно решение! Немедленно!

Как опытный менеджер производства, вы уже сталкивались с подобными ситуациями раньше. Вы знаете, как с ними справляться. Вы бежите к большой красной кнопке на стене, нажимаете на неё всей ладонью и тем останавливаете весь конвейер. Вы выкрикиваете приказы и запускаете стандартные процедуры по очистке, нахождению места поломки, починке и перезапуску конвейера. В течение часа производство возвращается на круги своя! Панике конец. Работа сделана замечательно!

Люксовая Часть Рынка

Прошло шесть месяцев. Вы поддерживали производственную линию в порядке, установили стандартизованные процедуры и меры безопасности. Всё идёт как по маслу, когда вы за главного. Вы заработали себе репутацию отличного менеджера производства. Вас хочет нанять более престижная шоколадная компания, специализирующаяся на люксовом шоколаде ручного изготовления.

Представьте себе дюжину художников, сидящих вокруг стола, занятых изготовлением люксовых шоколадных конфет. На полу и на столе следы брызг шоколада и апельсиновой начинки. Художники обмениваются семейными историями. Они улыбаются. Они смеются. Казалось бы весёлое место для работы. Услышьте звуки, запахи, почувствуйте атмосферу.

Одна из художниц доделала несколько шоколадных раковин и потянулась за чашей с начинкой. Осталось немного. Она встаёт и отпускает шутку - все смеются. Она идёт к холодильнику, берёт полную чашу начиночной пасты и возвращается с ней к столу. Не осталось и апельсинового масла, поэтому она идёт в кладовку и возвращается через пару минут с небольшой бутылью. Она наливает несколько капель масла в пасту. Ой, налила немного лишнего в этот раз. Эта партия будет более пикантной на вкус, чем обычно. В следующий раз надо быть поаккуратнее. С огромным терпением и мастерством она медленно замешивает апельсиновое масло в пасту.

Задержитесь на секунду, чтобы осмотреть сцену. Как менеджер производства вы отвечаете за то, как работает это заведение. Сравните эту сцену с производственным конвейером, который был под вашим руководством на предыдущей работе. Каковы ваши ощущения? О чём вы думаете?

Ещё пара художников закончили делать шоколадные ракушки и ждут, когда будет готова апельсиновая начинка. Они решили сходить в ближайшее кафе, а не просто сидеть за столом.

Ещё один художник добавляет финальные узоры на горстку конфет. Его нож для шоколада выскальзывает, посылая ещё две конфеты в корзину с браком. Выжившие конфеты он укладывает одну за другой в коробки и вручную завязывает каждую коробку ленточкой. Когда заполняются двадцать коробок, он ставит из на поднос и относит в магазин в соседнем здании. Десять минут спустя он возвращается, но останавливается на перекур, прежде чем опять взяться за шоколадные раковины.

Вы пробуете одну конфету. Она очень вкусная. Но у вас есть ощущение, что этому предприятию нужна встряска.Вы размышляете, нельзя ли сделать процесс более рациональным. Конфеты ручного изготовления стоят в десять раз дороже в производстве, чем обычные конфеты с начинкой с вашей прошлой работы. Тут много отходов, а производство медленное и ненадёжное. Работа требует много труда и подвержена настроениям очень искусных высоко оплачиваемых художников. Вы уверены, что дела можно вести куда более гладко.

К концу дня двенадцать художников седелали восемьсот семьдесят три конфеты. Меньше, чем дневная норма в тысячу конфет. Теперь вы отвечаете за производство, и с вас спросят за невыполненные нормы. Пока ещё рано делать выводы. Может завтра художники наверстают упущенное.

Через пару дней директор продаж вызывает вас к себе. У него настоящая паника. Основной конкурент, стремящийся к доминированию на люксовой части шоколадного рынка, созвал пресс конференцию. Они уменьшили зависимость от художников путём автоматизации большей части их работы. Теперь у них работают низко оплачиваемые операторы станков, производящие высококлассные конфеты на конвейере, за восьмую часть ваших затрат. При этом отходы уменьшились на семьдесят пять процентов, а объёмы производства в двдадцать раз выше, чем могут себе представить ваши художники. Если вы не ответите на это, то шансы банкротства вашей компании сильно увеличатся.

Конфеты конкурентов на вкус не отличаются от ваших, а цена в разы ниже. Вы опять видите доказательство рационализма конвейерного производства. Вы опять смотрите на своих шоколадных художников, обменивающихся шутками, тратящих четыре с половиной минуты на одну конфету, допуская ошибки и выкидывая брак.

Вам надледит с этим справиться в роли менеджера производства. Старший менеджмент возлагает на ввас надежды. Им нужен ответ! Вы либо справитесь с работой, либо вылетите с работы! Что вы предпримете?

Большая сеть гостиниц предлагает вашему конкуренту контракт на восемьдесят тысяч конфет в месяц. Постояльцы гостиниц находят индивидуально завёрнутые конфеты на столиках возле кроватей. Он пробуют их, им нравится, и они запоминают название изготовителя. В следующий раз, когда они будут в кондитерском отделе, они увидят, что ваши конфеты стоят в несколько раз больше, чем те, которые им так понравились в гостинице. Они протягивают руку за коробкой конфет. Какую они выберут?

Могут ли художники на вашем производстве соревноваться с эффективностью конвейера? Если вы не сможете сделать производство конкурентноспособным, то компания вылетит с рынка люксовых конфет, а вы окажетесь на улице. Что же вы сделаете?

Вы знаете, что нужно сделать, и вы делаете это. За три месяца вы полностью переводите производство на конвейер. Через ещё три месяца оно начинает приносить прибыль, а ещё через три ваша компания доминирует на рынке.

Фабрика по Выпуску Программ

Год спустя, имея достаточно опыта в качестве менеджера шоколадного производства, вы решаете двигаться дальше. Вам предлагают хорошо оплачиваемую работу в качестве менеджера проектов в компании, производящей программное обеспечение. Вы

принимаете это предложение. Вы почти ничего не знаете о создании программ, но вы точно знаете, что нужно для гладкой рациональной работы.

В первые несколько дней работы вы замечаете куда больший хаос, чем вы ожидали. Слишком много проектов не укладываются в сроки и в бюджет. Вам кажется, что народ вообще не может сработаться. Что ж, возможно, именно поэтому они наняли вас. Большой начальник говорит, что у вас есть шесть месяцев на то, чтобы взять проекты под контроль.

По мере вашего вливания в коллектив вы замечаете, что компания сильно зависит от настроений нескольких талантливых высоко оплачиваемых пограммистов. Похоже, что эти ребята никогда не могут дать точную оценку времени для задач. Когда они что-то производят, то часто оказывается, что половина написанного заказчику вовсе не нужна. И даже хуже - то, что заказчику всё-таки нужно, оказывается довольно ненадёжным.

Всё это кажется вам очень знакомым. Компания работает примерно так же, как та шоколадная фабрика с ручной работой до внедрения в ней вами дисциплины, предсказуемости и контроля затрат. Вспомните художников, сидащих за их столом, то и дело проливающими шоколад или начинку на пол. Вспомните, как нерационально они работали. Вспомните, как один конкурент чуть не обанкротил их. Вспомните, через сколько стресса вам пришлось пройти. Вы на своём тяжёлом опыте убедились, что компания, полагающаяся на работу художников, не может тягаться с эффективностью конвейерного производства. Вы спасли компанию от банкротства. Вы преобразовали бизнес. Вы создали одну из наиболее рациональных конвейерных линий в индустрии.

Похоже, что вы опять столкнулись с точно такой же ситуацией. Если эта компания не изменится, то может вылететь из бизнеса. Вы знаете как справиться с такой небрежностью и начать работать рационально, гладко и прибыльно. Вы и раньше это делали, так что повторить не будет проблемой. Вам дали шесть месяцев на изменение ситуации. Итак, каков ваш следующий шаг?

Весь ваш опыт говорит вам, что нужно вести проекты как конвейер на заводе. Эта теория подтвердилась на практике на шоколадных фабриках. Налаженоое конвейерное производство - это и есть та разница между предсказуемостью и хаосом. Вы справитесь, вы снизите затраты и привнесёте порядок в эти проекты. И этоименно то, что вы решили сделать.

От ручной работы к массовому производству

Ваш опыт работы на шоколадных фабриках сделал вас приверженцем Заводской Культуры, причём вы полагаете, что нет разницы, если вы производите самолёты, ложки, конфеты или программы. Проекты идут гладко, если применять основные принципы управления конвейерным производством, установленные в индустрии массового производства десятиетия назад.

До введения принципов управления конвейерным производством, во многом полагались на мастерство людей. При таком подходе продукт большей частью стоял на месте, а мастер перемещался вокруг него, измеряя и меняя инструменты и задачи. На это уходило ценное время, замедляя общий прогресс. Кроме того, для этого требовались мастера, умевшие делать много разных вещей, а значит, требовавшие высокую оплату. Эти нюансы не давали снизить себестоимость продукции. А качество конечного изделия всегда варьировалось; это зависело не только от индивидуальных талантов мастеров, но и от стабильности результатов каждого из них. Именно так работала фабрика по выпуску

конфет ручным трудом, пока вы не изменили ситуацию. И, скорее всего, компания по написанию программ работает в том же стиле.

В случае массового производства, работники стоят неподвижно в то время, как продукция движется мимо них по конвейерной линии. Спецфикации продукции известны из плана, составленного заранее вместе с разбиением целого процесса на стадии, расписанием и бюджетом для этих стадий и для всего процесса вцелом. Каждый работник выполняет задачи, определённые для его стадии производства, и передаёт результат труда следующему работнику конвейера для выполнения его задач. Завершённая продукция сходит с конвейера в срок, согласно бюджету, и соответствуя спецификациям. Менеджеры следят, чтобы работники строго придерживались плана, и наказывают тех, кто этого не делает, так как отклонение от плана угрожает успеху всего проекта.

Рационализм Конвейера

Конвейерное производство доказало свою эффективность в сравнении с ручным трудом в дешёвом, быстром и надёжном производстве больших количеств товаров. Для начала, каждый рабочий не должен быть особо опытным: есть чётко определённая роль по выполнению простых и повторяемых операций. Это удешевляет рабочую силу. К тому же они тогда работают быстрее, поскольку остаются на одном месте, а не тратят время на частую смену инструментов и операций. И, наконец, вероятность человеческой ошибки и разброса в качестве сильно уменьшаются благодаря повторяемости процесса и стабильности оборудования.

Отдельные мастера просто не могут соревноваться с производительностью конвейеров. Это уменьшило потребность в мастерах как среди менеджеров, так и среди рабочих. Рабочие превратились в операторов станков вдоль конвейера. Менеджеры сфокусировались на организационном администрировании предприятий.

Управление Проектом

Успех конвейерного производства вдохновил менеджеров многих других индустрий. Долгое время они наблюдали за провалами проектов: превышение бюджетов; затягивание сроков; упущение важных свойств продукции; или просто низкое нестабильное качество. Они искали способ контролировать эти риски и внести больше предсказуемости в свои проекты.

Они воспользовались конвейером как метафорой, чтобы понять, как нужно работать. Они привнесли научное управление, экспертов по эффективности, фирмы по инжинирингу менеджмента, чтобы помочь с определением корпоративных структур, с фиксированными ролями для рабочих и стандартными методами работы, которые обеспечат повторяемые результаты. Компании публиковали организационные диаграммы, устанавливающие иерархическую структуру руководства и удостоверяющие, что все решения принимаются или одобряются менеджерами. Менеджеры стали надзирателями. Они тыкали книгой с правилами в нос, если кто-нибудь не подчинялся.

В результате получился шаблон для руководителей проектов, чьи действия теперь рассматриваются как лучшие практики, или даже стандартом индустрии для управления любым проектом, независимо от индустрии. Сейчас существует множество курсов, которые покажут вам в точности, как повторить эти шаги на ваших собственных проектах. Появилась сертификация, которая распознаёт ваши навыки профессионального менеджера проектов.

Вот база, основанная на проверенном рационализме конвейерного производства:

Спланируйте наперёд:

- Напишите детальные спецификации продукта, включая необходимые стандарты качества
- Разбейте работу, необходимую для изготовления продукции, точно соответствующей спецификациям, на последовательность нескольких стадий
- Определите специализированные рабочие места (роли для рабочих) на каждой стадии
- Опишите оптимальную последовательность простых повторяемых действий, необходимых на каждом из таких рабочих мест
- Посчитайте время, необходимое для выполнения каждого действия, и выработайте соответствующее расписание для каждой стадии, включая время начала и окончания для всего проекта
- Определите экономические ресурсы (материалы, стоимость труда, станки, инструменты, и т.д.), необходимые для каждого действия. Просуммируйте их, чтобы определить стоимость каждой стадии и весь бюджет проекта.

Начните Производство согласно плану:

- Возьмите работников из доступного персонала и назначьте их на рабочие места, определённые в плане
- Прикажите рабочим выполнять повторяющиеся действия, предписанные на каждом месте, со скоростью, необходимой для завершения проекта вовремя и в рамках бюджета
- Нажмите кнопку Пуск, чтобы запустить производство

Мониторинг и Контроль:

- Непрерывно проверяйте:
- Рабочих на местах, чтобы убедиться, что они выполняют действия согласно плану
- Темпы производства, чтобы укладываться в расписание
- Потребление экономических ресурсов, чтобы проект укладывался в бюджет
- Конечную продукцию на предмет соответствия стандартам качества
- Там, где инспекция находит проблемы, справьтесь с ними:
- Остановите производство
- Исправьте причину проблемы, где возможно, путём:
- Починки сломанного оборудования
- Крика и запугивания рабочих
- Перезапустите производство
- Там, где причина проблемы не может быть устранена, попробуйте следующие методы:
- Увеличьте бюджет
- Предложите экономические льготы рабочим
- Добавьте рабочих к конвейеру
- Добавьте ещё один конвейер
- Увеличьте сроки проекта
- Измените спецификации продукта
- Отмените проект

Разработка Приложений - это Конвейерное Производство

К семидесятым годам индустрию программных приложений стали замечать. Другие индустрии уже давно продемонстрировали, что ручная работа, от которой до сих пор зависила индустрия приложений, приносит с собой хаос, неуверенность и риск. В то время, как конвейерное производство и контролирующее его административное управление, проявили себя как исключительно эффективные средства работы с рисками проектов и привнесли порядок, предсказуемость и контроль.

Многие менеджеры проектов по разработке приложений были вдохновлены этим успехом и пытались уменьшить свои риски и внести больше предсказуемости, надёжности и дисциплины. Как следствие, они втянулись в Заводскую Культуру и приняли конвейрное производство в качестве основной метафоры. Они создали виртуальные заводы по изготовлению программ с имитацией конвейеров, за которыми присматривают с помощью надёжных принципов управления. Довольно часто:

- Менеджеры создают план проекта, разбивающий необходимую работу на несколько стадий с чёткими временными и стоимостными рамками для каждой стадии, а, значит, и для всего проекта вцелом.
- Рабочие берутся из имеющегося персонала и назначаются последовательно на различные стадии проекта:
- Стадия требований: Спецификации продукта заранее записываются заказчиком или другим авторизованным человеком, тем самым переводя их бизнес потребности в спецификационный документ.
- Стадия анализа: Аналитик переводит спецификации продукта в полные фиксированные функциональные требования к программному приложению, производя документ с функциональными требованиями.
- Стадия дизайна: Дизайнер переводит функциональные требования в чёткую и ясную архитектуру приложения, формируя технический дизайн.
- Стадия кодирования: Программист, следуя техническому дизайну, пишет программный код.
- Стадия контроля качества: Тестеры проверяют функционал программного кода, чтобы удостовериться в том, что он соответствует всем требованиям из вышеупомянутых документов.
- На каждой стадии рабочие периодически пишут статус репорт, чтобы менеджеры могли следить за прогрессом.

Обзор Следственных Метафор

В предыдущей главе было упомянуто, что метафора о конвейере ведёт нас к многочисленным следственным метафорам. Позже мы их рассмотрим более подробно. Пока же нам хватит небольшого обзора, чтобы составидь общее впечатление о мировоззрении приверженцев Заводской Культуры.

Программы жёсткие

Программы рассматриваются как твёрдый физический объект, вроде металла, которому придают его конечную форму согласно набору процедур, производя законченный продукт. Заранее записанные детальные спецификации определяют форму, которую должна принять программа.

Дальше, производственным конвейерам необходимы физические предметы, которые будут передаваться от стадии к стадии. Разумеется, в разработке программ такого физического предмета не существует; таковым объектом будет сама программа, но она не

существует до поры до времени. Поэтому, до начала стадии кодирования её заменяет документация.

Как только завершается стадия кодирования, программа не должна меняться, разве что осуществляется периодическая поддержка.

Дизайн - это Проект

Техническая документация, произведённая дизайнерами, - это детальный проект для программистов, которого они должны очень аккуратно придерживаться. Поскольку каждый документ в производственном процессе является аккуратным переводом предыдущего документа в новый формат, то всё указанное дизайнерами в проекте может быть отслежено назад к требованиям, согласованным в контракте с заказчиками. А следовательно, есть контрактное обязательство, требующее от программистов, чтобы они в точности соблюдали детальный дизайн, представленный в проекте.

Требования Записываются

Всегда есть страх, что рабочего собъёт автобус или он уволится, забрав с собой всё его знание о проекте. Вместо того, чтобы носить знание в голове, работники обязаны описать результаты своей работы аккуратно и полностью в документации. Документация, созданная на каждой стадии проекта, показвает прогресс проекта, так что ничто не теряется при замене сотрудника. Это делает документы основным и наиболее надёжным средством коммуникации. Каждый документ полностью описывает всё, что необходимо знать рабочему на очередной стадии производства, и потому должен рассматриваться как единственный документ с требованиями, на котором рабочий на этой самой стадии будет основывать свою работу.

Менеджеры - это Командиры и Надзиратели

Если дать возможность рабочим принимать решения по ходу проекта, то никто не будет уверен, что происходит в конкретный момент на проекте. Для уменьшения этого риска используется предварительное планирование максимального количества решений. Если нужны какие-то ещё решения в течение проекта, у менеджеров есть необходимые полномочия для их принятия. И они же отвечают за то, чтобы рабочие их выполняли. О менеджерах можно думать как о командирах в армии, которые постоянно следят за рабочими, чтобы их команды правильно выполнялись.

Команды - это Рабочие

Рабочие, закреплённые за конкретными стадиями производства, берут документ, написанный в одном формате, переводят его аккуратно по расписанию в другой формат и передают рабочим следующей стадии для последующего перевода, пока последний перевод (программный код) не выйдет с конвейера по расписанию, в рамках бюджета и удовлетворяя изначальным требованиям.

Единственные таланты, необходимые рабочим, это форматы языков, на которых производятся переводы, а также скорость и аккуратность в выполнении этого перевода. Например, дизайнерам не нужно понимать концепцию бизнеса (это забота аналитика) и не нужно знать язык программирования (это забота программистов), им лишь нужно уметь качественно переводить функциональные требования в структурные проекты.

Менеджеры, втягивающиеся в метафору конвейерной линии, часто спрашивают - нельзя ли полностью автоматизировать процесс перевода. Пусть компьютер делает перевод, чтобы люди могли лишь набирать требования прямо в компьютер, а код бы выходил в качестве конечного продукта, исключая таким образом расходы на переводчиков вдоль конвейера.

Заказчики - это Покупатели

Изготовитель программ - это поставщик продукции, а заказчик - это покупатель этой продукции. Документ с требованиями образует контракт между заказчиком (чтобы они знали, что они получат) и поставщиком продукции (чтобы у них была чёткая цель). Как только этот документ подписан обеими сторонами, каждая сторона обязана соблюдать условия контракта. Поставщик обязан сделать программы точно по спецификациям в контракте и к дате, согласованной в контракте. А заказчик в свою очередь обязан заплатить по контракту за эту программу.

Содержание Фиксировано

Нет смысла начинать проект, если никто не знает его целей, так как он никуда никогда не пойдёт. Цели проекта образуют его содержание. Изменения содержания проекта не приветствуются посреди хода проекта, так как это может повлечь существенные риски. А потому содержание проекта фиксируется заранее в требованиях, а потом считается замороженным на длительность проекта.

Время - это Деньги

Поскольку почти никакого сырья в производство программ не вовлечено, основная статья расходов - это время сотрудников. Переработки ведут к превышению бюджета. Чтобы контролировать затраты, нужно заранее оптимизировать, спланировать в рассписании контролировать время, необходимое на выполнение каждой задачи

Качество Проверяется

Поскольку каждая стадия производства является всего лишь переводом документа из одного формата в другой, то довольно просто проверить качество на выходе каждой стадии. Всё, что нужно сделать менеджерам авторизованым сотрудникам, это просмотреть документы, проверить аккуратность перевода предыдущего документа и подписать, если качество удовлетворительно. Саму программу можно рассматривать как финальный перевод целой серии документов, которые ему предшествуют. Проверка качества заключается в сверке созданного функционала с требованиями контрактных обязательств, оговорённых (а также записанных и подписанных в оригинальном документе) с заказчиком.

Деньги - это Стоимость

Компании существуют, чтобы приносить прибыль. Прибыль - это цена минус себестоимость. Как только цена оговорена с заказчиком, любая экономия на производственных расходах увеличивает прибыль. Деньги, потраченные на производство, должны правильно учитываться и распределяться заранее в плане проекта. Также за ними нужно пристально следить в течение производственного цикла с помощью чётко установленных бухгалтерских правил.

Успех - это Соблюдение Норм

Этот тип работы не требует творчества, а лишь соблюдения норм. Пока работники правильно следуют плану и делают, что положено, то результаты будут предсказуемы. В таком случае проект должен чётко придерживаться плана, чтобы убедиться, что каждая стадия начинается и заканчивается вовремя и в рамках бюджета, а конечный продукт отвечает спецификациям. Когда все выполняют свои роли согласно плану, проект идёт гладко, и не будет нежелательных сюрпризов. Тогда успех будет измеряться тем, насколько всё идёт по плану.

Неудачи Наказуемы

Когда люди пропускают сроки или производят плохую продукцию или выполняют задания не так, как в плане или как велено менеджером, то такие люди подвергают опасности целый проект. Когда работники не справляются с обязанностями, они должны за это отвечать. В этом случае у менеджеров есть власть, чтобы их наказать. Наказания могут быть просто словами, а могут быть и потерей бонуса или даже потерей работы в особо серьёзных ситуациях. Осведомлённость о потенциально возможных наказаниях служит как сдерживающий фактор, заставляющий сотрудников усердно работать.

Растрата - это Небрежность

Каждая стадия производства тщательно спланирована, чтобы получить правильные результаты с минимальными отходами. Небрежность в следовании плану порождает растраты. Небрежность ведёт к переделкам, а переделки повышают расход ресурсов, подвергая риску бюджет, расписание и стандарты качества.

Прогресс - это Завершённые Задачи

Влиятельные эксперты в индустрии программных обеспечений давно говорят о том, что ошибки в требованиях или дизайне очень дороги в исправлении, если программирование уже началось. Поэтому конвейерное производство программ настаивает на том, что нельзя возвращаться на стадию уточнения требований или вносить изменения. Прогресс может двигаться только вперёд. Поэтому все задачи на каждой стадии должны быть выполнены полностью и акуратно.

Отчёты о статусе позволяют менеджерам следить за прогрессом и выполнением задач на данной стадии конвейера. Кроме того должен быть способ сигнализировать о завершённости стадии, тогда может начаться новая стадия. С вещественными предметами легко распознать, что станок закончил переработку сырого материала. В написании программ это не так просто. Один подход - это попросить аналитика или дизайнера сказать, что работа выполнена. Однако многие компании предпочитают оставлятть такие важные решения менеджерам, а потому менеджеры просматривают и подписывают документы, тем самым заявляя, что ошибок нет и можно заморозить документацию. Это официальная печать одобрения для движения дальше к следующей стадии конвейера.

Этот принцип движения только вперёд многих людей привёл к тому, что конвейерную модель описали как водопадный подход к разработке приложений, поскольку вода течёт только вниз и никогда обратно вверх.

Глава 5: Команды - это...

Что Такое Команда?

До сих пор мы говорили о взглядах, ценностях и метафорах. Мы видели, что можно убедить кого-то изменить их взгляды, но ценности и метафоры сидят куда глубже, на эмоциональном уровне. Всё это имеет значение, поскольку касается не только людей, но и отношений между ними, выполняемой работы и того, как они измеряют свой и чужой успех. То есть, ценности, взгляды и метафоры являются сердцем команд и командной работы.

Разумеется, все говорят о пользе команд и командной работы. Мы постоянно слышим, что командная работа - это хорошо. А потому, неудивительно, что компании рекламируют и афишируют её для “командных игроков”. Всё было бы хорошо, если бы это означало, что компании нанимают и развивают людей, хорошо работаюших вместе. Но из своего опыта могу сказать, что “командные игроки” - это эвфемизм для людей, которые будут выполнять приказы, спускаемые им сверху боссом. В этом случае фраза “командные игроки” означает послушание и не имеет ничего общего с командами и командной работой.

Разумеется, можно собрать группу людей и попросит их работать как одна команда, но эти люди не будут работать как команда, до тех пор пока они не станут командой. А для этого им нужно объединиться, сработаться и расцвести.

Как Команды Объединяются?

Команда начинается с немногим большего, чем несколько человек, озабоченных их собственными интересами. Чтобы объединиться как команда, должно присутствовать взаимное понимание и уважение того факта, что эти люди теперь заодно.

Как я мог убедиться, никто не может навязать это чувство единения команде, члены команды должны сами увидеть это друг в друге, а для этого им нужна возможность узнать друг друга и найти что-то общее, что поможет им отличать себя от окружающих. Если члены команды разделены и не имеют возможности развить чувство взаимной принадлежности, у них никогда не будет единства, и команда никогда не станет одним целым. Также я видел, что намеренный роспуск команд в конце проекта и возвращение их членов в “запас” разрушает любое единение команды, которое могло начать формироваться.

Один мой коллега Нил Крейвен сидел в большом офисе открытого типа, окружённый людьми, с которыми он никогда не работал. Такой тип офиса позволял экономить на аренде и иметь всех сотрудников на виду. Это никак не было связано с командами. Компания ратовала за “виртуальные команды”, в которых временные группы людей назначались на проекты на основании почти случайной их доступности из огромного географически разделённого списка сотрудников компании.

Компания проталкивала идеи выгод командной работы, но, как сказал Нил, люди были изолированы и одиноки. Очень часто члены команд никогда друг друга не видели в лицо и

никогда не работали вместе по окончании проекта. Не было ощущения принадлежности и ничего, что бы объединяло команду, кроме кратковременного назначения на проект, которым управлял обычный менеджер проектов.

Как Срабатываются Команды?

Как только у команды появляется ощущение взаимного единения, им нужно научиться работать вместе как команда. Благонамеренные усилия руководства часто терпят неудачу в попытке симулировать это. Типичные тим-билдинги привносят некую радость в коллектив, но редко имеют длительный эффект. Мотивационные постеры обычно рассматриваются как банальные декорации.

Команда образуется не под внешним воздействием, а лишь когда команда строит себя сама. Команда может сплотиться вокруг общих интересов, которыми могут быть правила, по которым договорились работать члены команды. Конечно же, поначалу может наблюдаться борьба конфликтующих взглядов, пока один из них не возьмёт верх. Основатель команды, или доминируюший член команды, может даже навязать свои собственные взгляды в команде, особенно если у него есть авторитет, но позже эти взгляды подвергнутся тесту.

В детстве я играл в ковбоев и индейцев с другими мальчишками, жившими по соседству. Мы разыгрывали великие битвы, о которых читали в приключенческих книгах. У одного мальчика, Генерала Кастера, было много игрушечных пистолетов, и он давал их другим поиграться.

В Битве Маленького Большого Рога Генерал Кастер отказался умирать. Он заявил, что одет в волшебные доспехи, отражающие пули. Когда остальные дети запротестовали, Генерал Кастер забрал все свои пистолеты и отправился домой, тем самым закончив игру.

Генерал Кастер играл по правилам, отличным от правил других мальчиков. Это не давало нам воспроизводить битвы подобающим образом. Вскоре нам это надоело и мы договорились сделать свои собственные пистолеты из палок и резинок. Генералу Кастеру пришлось играть по нашим правилам или не играть вообще.

Когда команда выработала общие взгляды, она может начинать работать над связями, помогающими команде сплотиться. Я видел несколько команд без общих взглядов, в которых почти отсутствовала связь между членами команды. Это не давало им объединиться, а потому командам не хватало внутренней силы, чтобы прожить достаточно долго.

У одной крупной компании, производящей финансовое программное обеспечение, было несколько команд в разных странах. Руководство беспокоила изолированность команд, они хотели, чтобы лидеры команд встречались раз в месяц для обмена опытом. Поскольку я был лидером одной из таких команд, мена пригласили на эти встречи.

Первая встреча была интересной - все слетелись из разных уголков мира и представляли свои проекты другим. Мы узнали, что проекты сильно различаются. Каждый проект требовал совершенно уникального набора взглядов на то, как измерять успех. Один проект считал за успех захват доли развивающегося рынка. Другой - продление высоко оплачиваемого контракта на оказание консультационных услуг одному клиенту на довольно длительный срок. Третий - поставка нового продукта, финансируемого сразу несколькими банками.

Ко второй встрече стало понятно, что хоть мы все и работали в одной компании, у нас было мало общего, что позволило бы нам иметь одинаковые взгляды.

Третьей встречи не было. Мы все вернулись к своим проектам и редко когда-либо ещё виделись.

Сработаться очень важно, так как это заставляет людей заботиться не только о своём будущем, но и о будущем команды. Когда сработавшаяся команда обнаруживает угрозу, члены команды приходят в состояние повышенной готовности. И, чтобы выйти из этого состояния, они делают всё возможное для выживания команды.

Если же угроза исходит изнутри самой команды, то я видел случаи избавления от членов команды, угрожавших сработанности коллектива.

Я как-то возглавлял команду, разрабатывавшую систему упрабления базами данных, в ОбъектВаре. Одой из объединяющих команду ценностей было предположение, что если относиться к людям с доверием, то они оправдают его добросовестным трудом. У Майкла (одного из сотрудников), очевидно, были другие ценности, включая веру в то, что ошибки приемлемы, если последствия незначительны. Это создавало некоторые конфликты в команде, но Майкл отмахивался от критики, как от слишком эмоциональной реакции других людей.

Однажды, Майкл пропустил важную встречу, что оставило довольно неприятное впечатление о всей команде. Я нашёл Майкла дома, но он предъявил какое-то обобщённое оправдание, где умудрился обвинить двух других членов команды. Быстрая проверка выявила невиновность тех двух. Команда обсудила этот случай и проголосовала за то, чтобы исключить Майкла из команды. Его ценности были разрушительны для команды, а его поведение подрывало те устои, вокруг которых сработался коллектив. Майкл был исключён из команды, а потом и уволен из компании.

Итак, в сработавшемся коллективе трения возникают очень редко, а работа идёт гладко. Однако, введите в коллектив кого-то, кто не принимает ваших ценностей, и получите кота в стае голубей.

У одного из моих клиентов была команда, которая годами успешно работала. Однако, руководство беспокоилось, что команда увязнет в старых привычках, а потому решило сделать вливание. Они наняли Жака, чьё резюме было исключительно впечатляющим. У Жака было много классных идей, которые обещали встряхнуть ситуацию и научить команду работать по-новому.

Но его встряска оказалась больше похожей на землетрясение. Жак принёс с собой менталитет жёсткого командования и контроля, что шло вразрез со взглядами команды, которая в основном полагалась на сотрудничество и консенсус.

Жак не мог понять, почему команда такая недисциплинированная, и сильно расстраивался от того, что они постоянно съезжали с колеи. Чем больше команда сопротивлялась, тем сильнее Жак пытался контролировать их. Команда не понимала, почему Жак такой тиран, и почему он пытается навязать им то, что уводило их от нормальной совместной работы.

Столкновение взглядов проявилось на результатах: проекты замедлились, мотивация упала, никто не был счастлив. Жак обвинял в этом команду, а команда обвиняла его.

Со временем, команда просто отказалась работать с Жаком. Жака уволили, и проекты стали улучшаться.

С другой стороны, я видел, как команда защищала себя так, что даже целиком покинула организацию!

Компания по разработке приложений, с которой я сотрудничал, взращивала высоко эффективную команду программистов, чья продукция помогала процветанию компании. Тем не менее, после нескольких успешных лет, пара дорогих ошибок отдела продаж привела компанию к финансовому кризису. Страх сокращений повис в воздухе. Некоторые из программистов стали искать работу в других компаниях, что стало угрожать целостности команды. Я был шокирован тем, что произошло: члены команды собрались для обсуждения кризисной ситуации, после чего все вместе уволились. Целая команда перешла работать в другую команду. Им пришлось это сделать для выживания команды.

Как Команда Расцветает?

Как только команда сплотилась, благодаря общности взглядов, и сработалась как коллектив, ей нужно пережить несколько успехов в своей среде, что бы расцвести в полном объёме. Это означает защиту себя от внешних угроз.

Довольно очевидный, но часто недооценённый факт, что у команды не просто отношения с её менеджером. Все, кто вне команды, - это часть среды, в которой существует команда. Эта среда может быть дружелюбной или враждебной по отношению к команде, или и то и другое. Чтобы среда была дружелюбной, команда должна быть полезной для среды, иначе, команда не будет представлять ценности и даже может рассматриваться как угроза для среды, а потому может оказаться под сильным давлением.

Таким образом, выживание команды зависит от удовлетворения нужд среды своей повседневной работой. Если члены команды видят, что мировоззрение команды усиливает их эффективность в повседневной работе и улучшает отношения команды со средой, то такое мировоззрение будет только усиливаться внутри команды. Повторяющийся успех ведёт к тому, что взгляды команды перерастают в их ценности, а они усиливают внутреннюю интеграцию команды, закрепляют место команды в среде и защищают команду от внешних покушений на её целостность.

Если ценности оказываются неэффективными и ослабляют внешние связи, команда будет чувствовать угрозу. В результате команде придётся пересмотреть свои ценности, и, если угроза достаточно реальна, то принять новые взгляды (ведущие к новым ценностям), которые уменьшат беспокойство команды и увеличат продуктивность команды.

При смене взглядов на более подходящие для среды команда избавляется от внешних угроз. Обратной стороной медали является то, что новые взгляды могут начать угрожать внутренним связям команды. Члены команды могут иметь разногласия относительно того, какими должны быть новые взгляды, и могут иметь разное отношение к доминирующим взглядам. Это угрожает сплочённости команды и может привести к расколу.

Хоть и редко, но мне довелось видеть, как команды, работавшие слаженно долгое время, преодолевают такие угрозы. Ответ, по крайней мере в командах, где я работал, состоит в том, что у команд есть наработанные механизмы для ситуаций, выходящих за рамки их взглядов. А именно, сработавшиеся команды становятся экспертами в заполнении

недостатков правил точно так же как и отдельные эксперты: они призывают на помощь метод развития метафор.

Метафоры, разделяемые коллективом, образуют базу его культуры. В таком случае, процветание команды означает, что команда смогла выработать культуру, чьи основные метафоры помогают команде адаптироваться и выжить в меняющейся среде.

Как только команда выработала культуру, помогающую выжить долгий срок, главной угрозой такой команде становится давление извне, заставляющее их отказаться от своей культуры в пользу другой. Инстинктивной реакцией команды будет сопротивление, которое можно сломить, лишь уменьшив их беспокойство до того уровня, на котором они согласятся сменить культуру.

Глава 6: Менеджеры - это...

Авторитет Руководства

Мы уже видели, что у проектов больше шансов на успех, если команда объединена вокруг культуры, основанной на общих метафорах. Эти метафоры подсказывают как вести себя в штатных ситуациях и как заполнять пробелы правил при незнакомых обстоятельствах. И даже если команда сплотилась вокруг общих взглядов, руководство может захотеть их изменить.

Менеджер - это некто, имеющий власть над командой. Менеджер может использовать эту власть, чтобы навязать свои собственные взгляды, которым команда инстинктивно сопротивляется. Вместе со своими взглядами менеджер одновременно навязывает и метафоры, породившие эти взгляды, а также соответствующую культуру. В результате происходит столкновение культур, приводящее к беспокойству команды и сильному её желанию обойти требования менеджера и продолжать следовать своим собственным метафорам, чтобы уменьшить беспокойство и увеличить шансы на успех проекта. С другой стороны, менеджер скорее всего будет рассматривать такое поведение как угрозу проекту и подрыв его авторитета. Менеджер тоже начнёт беспокоиться и давить на команду.

Моя команда была нанята бывшим бухгалтером, ставшим теперь старшим менеджером. Нам было поручено сесть с экспертами отдела кадров, чтобы понять их бизнес модель и написать новое программное обеспечение, поддерживающее её.

Через три месяца мы делали презентацию прогресса, и нас спросили “Когда уже будет готово?” Один из членов команды ответил “От вас зависит, когда вы решите, что приложение достаточно хорошо, чтобы отдать заказчику. А тем временем мы будем продолжать его улучшение сколько угодно долго.” Работа продолжалась месяц за месяцем. Команда выпускала новые версии приложения, но менеджер не хотел отдавать его заказчику, пока оно не будет совсем “готово”.

Команда всё больше расстраивалась от того, что менеджер не может понять - приложение никогда не будет “готово”, ибо оно постоянно улучшается со временем. А менеджер всё больше терял терпение от того, что команда не хочет закончит начатое.

Это противостояние длилось почти три года, когда, наконец, менеджер сдался под давлением заказчика и отправил ему продукт, который по его словам всё ещё не был закончен.

Культура FIFO

Я заметил, что довольно часто команды и их менеджеры являются приверженцами конфликтующих культур. Столкновения культур создают конфликты между людьми. Не удивительно, что сотрудники часто жалуются на слишком большое количество начальников.

В чём же дело? По своему опыту могу сказать, что люди не говорят, будто им не нужны начальники, скорее они хотят более дружелюбного отношения с ними. И это не из-за того,

что они избалованные дети, желающие развлекаться и не нести ни за что ответственности. Скорее от того, что избыток диктаторского управления довольно плохо отражается на результатах проекта.

Однажды у меня произошла интересная беседа с владельцем среднего размера компании, производящей программные приложения. Он хотел знать, как увеличить мораль и производительность. Поговорив с разными людьми в компании я выяснил, что владелец был больше заинтересован в послушании, чем в результатах. У многих сотрудников были классные идеи на тему улучшения культуры, чтобы поднять мораль и стать более успешными на проектах, но они боялись озвучить их из страха быть прозванными “отступниками”.

Будучи наивным я доложил владельцу компании о некоторых обнаруженных различиях культур, на что он мне резко ответил: “У нас тут культура FIFO. Люди могут либо влиться либо свалить (Fit-In or F***-Off)”. Именно этим люди и занимались.

Враждебные Менеджеры

Странно, но команды иногда выигрывают от враждебно настроенных людей. Враждебность обеспечивает команду врагами, а враги помогают команде объединиться и сфокусировать свои усилия. Враг даёт команде кого-то, с кем можно соревноваться, тем самым усиливая сплочённость команды. Конкурент на рынке - это замечательный враг, ибо не даёт команде расслабиться и заставляет их творить и адаптироваться, чтобы выжить и победить.

Однако, плохо, когда такую роль недруга играет менеджер команды.

Менеджер будет врагом, если он заставляет команду работать или организовываться так, что в теории звучит красиво, но на практике разрушительно. Примерами этого могут быть попытки навязать другие ценности, манеру поведения и организационную структуру, которые будут постоянно игнорировать культуру команды и вредить её сплочённости.

В крайних случаях это может быть умышленным саботажем, но чаще всего у менеджеров хорошие намерения, и они пытаются помочь командам быть эффективными. Типичным случаем умышленного саботажа является попытка менеджера присвоить себе всю славу за успехи команды, а вину за все ошибки свалить на команду. Такое отношение явно разовьёт эго и статус менеджера, но лишь за счёт команды.

Однажды я пошутил с командой, сказав, что моя роль как их менеджера очень простая: лично получать все похвалы за их успехи (”смотрите, как классно они сработали под моим исключительным руководством!”), и винить их во всех неудачах (”какая некомпетентная кучка лоботрясов!”). После вежливых усмешек один из членов команды сказал: “именно так работал мой предыдущий менеджер!”

Несознательный саботажник не понимает, что его действия снижают эффективность команды или даже угрожают целостности команды. Существует четыре классических случая неосознанного враждебного менеджмента, которые я наблюдал много раз:

1. Микро Менеджеры. Беспокоятся о том, что люди будут ошибаться и делать не идеальную работу. Они боятся утратить контроль над проектом и иметь недостаточное влияние на его успех. Они не способны делегировать и настаивают на том, чтобы единолично принимать все решения. Поэтому они висят над душами сотрудников,

постоянно следя за каждым их действием. Разумеется, это замедляет процесс принятия решений и затрудняет творчество. Если хотите увидеть экстремальный пример, поищите в Интернете заметки Эдварда Майка Дэвиса о Нефтяной Компании Тигр. (Edward Mike Davis - Tiger Oil COmpany)

2. Менеджеры бюрократы. Эти полностью полагаются на авторитарную иерархию и навязывают стандартизированные процессы со строгими административными правилами. Сотрудники должны слепо следовать правилам и заполнять нужные формы. Успех проекта измеряется точностью выполнения правил командами и их менеджерами. Вцелом, такой стиль управления считает, что чёткое администрирование - это уже успех, а не инструмент поддержки создания высококачественной продукции. Бюрократия увеличивается, качество проектов снижается. Никто не задаёт вопрос, который должен вертеться у всех на языке: “Если бы у нас не было этих процессов и иерархии, создали бы мы их сегодня и порекомендовали бы их как лучший способ начать работу?”

3. Менеджеры Посредники не столько говорят командам, как выполнять их работу, сколько мешают формированию хороших отношений команды со средой. Они пытаются “защитить” команду и среду друг от друга, действуя в качестве посредника, считая, что те не готовы общаться напрямую. Это создаёт препятствия передаче и чёткому пониманию нужд заказчика, ограничивает возможности обратной связи и замедляет все коммуникации. Типичная ситуация, когда менеджер постоянно преувеличивает возможности команды, что выливается в резкий переход от оптимизма к расстройству. Это не выгодно ни команде, ни заказчику и портит отношения между ними.

Некоторое время я вёл несколько команд в рекрутерской компании, имя которой умолчим. Одна из моих команд пожаловалась, что мой начальник только что нагрузил их кучей работы с нереальными сроками, к тому же относящейся к проекту, который всё равно будет отменён. Я обратился к своему начальнику, чтобы объяснить, что он обрекает команду на неудачу, чем вредит нашим отношениям с заказчиком. Его ответ был ошеломителен: “Тебя никто не спрашивает. У нас тут иерархия, а не демократия. Мы уже пообещали это клиенту, а твоя работа состоит в том, чтобы выполнять наши обещания.” Не удивительно, что и я и члены этой команды вскоре покинули компанию.

4. Нерешительные Менеджеры меняют своё мнение о приоритетах, не давая команде сделать хоть что-нибудь. Поэтому редко когда что-либо поставляется заказчику, и команда теряет свою репутацию.

Враждебные менеджеры оставляют командам три выбора:

1. Бунтовать: Сопротивляться менеджеру, чтобы защитить единство команды. Скорее всего, начальство будет рассматривать это как неповиновение и предательство. Или, как минимум, как упрямство и неблагодарность. Часто это выливается в то, что менеджер, пользуясь властью, наказывает команду, чтобы убедиться в большем повиновении в будущем.

2. Согласиться: Отказаться от надежды на единение и делать то, что сказал менеджер. В такой ситуации они переключаются с того, что нужно для успеха проекта, на удовлетворение менеджера. Теперь успех измеряется в терминах соблюдения правил, а не результатов проекта. Если проект завершится успешно, то это всего лишь побочный эффект, а не сознательно достигнутая цель команды. Даже в этом случае успех был бы куда больше, если бы команда подчинялась добровольно.

3. Разойтись: Позволить команде разделиться, дав возможность присоединиться к другим (авось, более сплочённым) командам, избавив менеджера от его враждебного отношения.

Джереми руководил продуктовой командой одного из моих клиентов. Он был умным и энергичным, но обрёк свою команду на неудачу. Он верил, что клиенты тупые, и команды не должны слушать их, ибо это неправильно.

Джереми потребовал, чтобы ему позволили принимать все решения, одобрять и даже изменять все требования заказчиков. Поскольку команда была нацелена на удовлетворение нужд заказчика, отношение Джереми к команде было деструктивным по отношению к их сплочённости.

Члены команды стали появляться на работе в странное время, чтобы избегать менеджера. Они были счастливы, когда он уходил в отпуск или заболевал, поскольку это давало им возможность работать вместе и фокусироваться на нуждах клиента. Конечно же, по возвращении, Джереми отчитывал сотрудников, заставлял их переделывать работу и назначал их на разные проекты, чтобы исключить сговор.

Старшее руководство уважало Джереми за его упорный труд и преданность, и не могло предотвратить саботирование работы команды. Поскольку команда ценилась выше, чем отдельно взятый сотрудник, Джереми попросили покинуть компанию. Это воодушевило команду, но оставило Джереми уверенным, что всё рухнет без его поддержки. Всё не рухнуло, наоборот, компания наняла нового менеджера, который создал куда более здоровые отношения с командой и помог им сплотиться и значительно увеличить эффективность.

Дружелюбные Менеджеры

Итак, враждебный менеджер может быть угрозой существованию команды, а потому обычно сталкивается с существенным сопротивлением со стороны команды. В таких ситуациях менеджер может воспользоваться своей властью, чтобы утихомирить или заставить подчинённых слушаться, но он никогда не добъётся их добровольного сотрудничества. Проблема в том, что враждебные отношения - фундаментально нерабочие отношения между менеджером и командой, ведущие команду к заниженным результатам, разочарованию в работе, а часто и к разрушению отношений и даже к разрушению команды.

Если разобраться, то враждебный менеджмент основан на недоверии. Менеджеры не доверяют сотрудникам в том, что они примут правильные решения и будут усердно работать. Поэтому менеджеры принимают все решения, приказывают работникам выполнять эти решения и внедряют меры контроля, чтобы рабочие не халтурили.

Недоверие означает подозрение. Если люди видят, что их менеджер в них не уверен и относится к ним с подозрением, то это может повредить отношениям. Сотрудники начинают относиться к намерениям менеджера с подозрением, теряют к ним доверие, а потом и преданность. В лучшем случае, результаты будут средними, поскольку взаимное недоверие поднимает стоимость, портит мораль, снижает качество принятия решений и в корне убивает желание творить.

Бьёрн и я были оба директорами. Вице-президент уволил людей из команды Бьёрна, чтобы сэкономить на затратах, а потом нагрузил его команду дополнительной работой.

Бьёрн обратился к высшему руководству за практической помощью. Ответ, наверное, был взят прямо из комиксов о Дилберте: “У меня для вас один совет: работайте умнее, а не усерднее!”

Не удивительно, что мотивация Бьёрна упала ниже плинтуса. Несколько месяцев позже, не видя просвета в таких отношениях с руководством, Бьёрн покинул компанию.

С другой стороны, дружелюбные менеджеры строят отношения, основанные на доверии, а не на власти. Конечно, доверять людям бывает рисковано, но недоверие приводит к ещё большим рискам. При снижении доверия менеджеры и команды учатся придерживать информацию, манипулировать фактами, скрывать ошибки, пытаться получить похвалу за успехи и свалить вину за неудачи на других. В результате прогресс замедляется, стоимость повышается, а эффективность вылетает в трубу. В своей книге “The Speed of Trust”, Stephen M.R. Covey называет это “налогом недоверия” противопоставляя его “дивидендам доверия”.

Когда я впервые присоединился к довольно крупной консультационной компании Силиконовой Долины, меня включили в команду под руководством Грэга. Он нам сказал, что не будет влазить в нашу работу и будет защищать нас от внешних угроз, а также помогать нам строить и поддерживать хорошие отношения с клиентом.

По крайней мере так было в теории. На практике же Грэг находился под огромным давлением со стороны его собственного руководства, которое хотело снизить затраты и увеличить прибыль. Консультантов считали немного большим, чем просто статьёй расходов, несмотря на то, что команда генерировала существенную часть дохода компании.

Ситуация достигла критической точки, где Грэг сказал, что компания переходит из консалтинга в поставку коробочных приложений, поскольку приложения “как печатание денег - не требует зарплаты”. Консультантам приказали использовать любые возможности для продажи лицензий клиентам, чтобы компания начала процветать только от доходов с лицензий и смогла избавиться от консультантов. Неудивительно, что консультанты увидели в этом враждебность со стороны руководства, которое теперь рассматривалось как противник, а не союзник. Несколько лучших консультантов ушло, а многие из оставшихся были сильно демотивированы и потеряли доверие к руководству.

На еженедельных митингах с Грэгом команда пыталась восстановить уверенность. После нескольких недель постоянных уверений в том, что компания одумалась и опять ценит консультантов выше всего, доверие начало восстанавливаться, а мотивация возрождаться. Но вскоре компания анонсировала, что продала большую часть своего консультационного бизнеса, обогатив старших менеджеров и разозлив консультантов.

Разумеется, доверие должно быть двусторонним. Когда менеджеры начинают доверять своим командам, а, что более важно, команды начинают доверять своим менеджерам, то пропадает потребность в принуждении и проистекающими из него посредственных результатах. Вместо этого можно развивать преданность и сотрудничество. При таких условиях сотрудники становятся куда более мотивированными и с большим желанием посвящают свои ценные знания и созидательность на благо проектов и команд, а, значит, и компании. То есть, доверие помогает получить от людей наилучших результатов, оставляя людей счастливыми. Доверие снижает расходы и повышает ценность продукции, тем самым увеличивая прибыль.

Прежде чем команда станет вам доверять как менеджеру, вы должны продемонстрировать команде, что бы достойны доверия. Доверие означает уверенность. Открытость и честность - это хорошее начало, но недостаточно просто быть милым с людьми. Так вы можете понравиться команде, но этим трудно пробудить их уверенность в вашей компетентности и вашей способности руководить ими.

На заре своей карьеры руководителя я изо всех сил старался понравиться команде. Я хотел быть их другом. Они всегда были рады пойти со мной на пиво, но я так никогда и не заслужил их полного доверия. Несколько лет понадобилось, чтобы понять, что нравиться ещё не значит быть опорой.

Моим оправданием было то, что начальник, немного тиран, часто ставил глупые задачи. Я редко возражал, так как, если честно, немного боялся его. Вместо этого я приходил к своей команде с улыбкой и говорил: “Ребята, нам сказали сделать это. Я знаю, что это бестолково, но мы ведь все знаем, что начальник идиот, так что давайте просто сделаем, что сможем.”

Это ужасный метод управления. Он оставлял мою команду обессиленной и без поддержки. Я не смог о них позаботиться. Я был слабым менеджером: заботился только о себе.

Чтобы заслужить уверенность, нужно продемонстрировать собственную надёжность, сфокусировавшись на общих выгодах и показывая свою преданность команде. Это означает, что нужно всегда быть лучшим в своей работе и способным помочь команде стать лучшей в их работе.

Кроме того, нужно принять личную ответственность за результаты. Результаты очень важны, так как другие будут постоянно оценивать ваши способности, основываясь на том, как вы работали раньше, как работаете теперь и что можно ожидать от вас в будущем. Вместо того, чтобы хвалить себя за успехи и обвинять команду в неудачах, нужно научиться разделять успех с командой и лично принимать вину. Пусть за себя говорят ваши результаты, а не титул.

Филипп был очень талантливым тим лидером в одной английской компании. После нескольких лет проявления своего таланта он был повышен до директора, а потом его сделали партнёром компании.

Его очень уважали за его опыт и знания, но, к сожалению, новый титул ударил ему в голову. Пользуясь возросшей властью он всё меньше стал полагаться на опыт, а всё больше на приказы и контроль. Он превратился в диктатора, а его желание помогать рядовым сотрудникам испарилось.

Постепенно его команда утратила к нему доверие. Он оказывался всё больше изолированным, а через несколько лет оказался лишь администратором, одобряющим ежемесячные расходы. И даже в этой роли он был знаменит своей придирчивостью: полдня гонялся за кем-нибудь, кто внёс шоколадку в отчёт о командировке.

Со временем, уровень взаимного доверия скатился так низко, что его присутствие в компании приносило больше вреда, чем пользы. Остальные партнёры проголосовали за его увольнение. Какова же была его реакция? “Но я же партнёр! Вы не можете меня выкинуть. Я решаю, кто может остаться, а кто уходит!”

Однажды доказав, что вам можно доверять, вы можете начать строить доверительные отношения, если сможете продолжать вызывать доверие. Сначала, скорее всего, это будут индивидуальные отношения с членами команды. По мере роста доверия эти отношения могут перейти на всю команду.

Вобщем, если вы хотите добиться хороших результатов в короткие сроки, нужно перестать быть подозрительным, надеяться на подковёрные уловки, думая, что люди лишь статья расходов. При этом нужно начать строить, поддерживать и развивать доверительные отношения со своей командой. Тогда менеджер уже не будет человеком, которого команда боится, а будет тем, кто помогает команде добиться наилучших результатов.

Менеджер Рефери

Это вспомогательные (поддерживающие-supportive) отношения между менеджером и его командой.

В конце 80-х и все 90-е года несколько книг и статей заявляло, что наиболее эффективный подход к руководству следующий:

- Нанять лучших людей
- Дать им чёткие цели
- Уйти с дороги, пока они всё делают замечательно

Это явно была реакция на гнетущий недружелюбный менеджмент, замедлявший проекты и мешающий инновациям. Это же привело к одному из наиболее популярных терминов управленцев того времени: наделение властью (empowerment). Этот тип отношений основан на проявлении доверия, что противопоставляется враждебному руководству, основанному на подозрениях.

Менеджер и команда договариваются о том, что они могут ожидать друг от друга, а потом доверяют друг другу в том, что сдержат слово. Затем, если каждая сторона удовлетворяет ожидания на приемлемом уровне, то уровень доверия в отношениях растёт, и отношения требуют меньше формальностей.

Наделение властью должно помочь пробиться сквозь бесполезный управленческий интерфейс и дать команде больше власти в принятии решений о том, как делать работу более эффективно. Основополагающая идея состоит в том, что команда по необходимости сама найдёт способы приспособиться к среде. Команда изменит свои взгляды и ценности, а, следовательно, и свою внутреннюю структуру и способы работы, чтобы выжить. Менеджер сопротивляется соблазну использовать власть, чтобы навязать собственные взгляды и ценности и вмешаться во внутреннюю кухню команды. Вместо этого менеджер остаётся восприимчив к нуждам команды и способности объединиться и процветать самостоятельно. Менеджер, который не может сопротивляться вмешательству в этот процесс, рискует стать саботажником, какими бы благими ни были его намерения.

Это звучит здорово, так как это освобождает менеджеров от большой части ответственности, а команды освобождает от нежелательного вмешательства. Однако, на практике это часто означало, что на команды взваливали много ответственности и не предоставляли почти никакой поддержки от руководства, когда им нужна помощь.

Возможно, что эта нехватка практической помощи происходила из убеждения, что отсутствие руководства лучше, чем враждебное руководство. Из-за этого многие команды запутывались; сталкиваясь с препятствиями им не хватало сил самостоятельно преодолеть их, и не было возможности менять цели и перенаправлять ход проекта, чтобы приспособиться к меняющимся обстоятельствам.

Я на себе прочувствовал наделение властью, когда я возглавил команду в ОбъектВаре. Исполнительный директор Ричард поручил нам скопировать сложную программу конкурентов. Сперва всё шло хорошо, но со временем до нас дошли слухи, что всё большее число заказчиков считают, что мы просто делаем продукт вроде “и я туда же”, без каких-либо инноваций, которые могли бы их заинтересовать. Когда мы рассказали об этом Ричарду, он ответил: “просто сделайте его и дайте мне знать о готовности, а я позабочусь о продажах”. После трёх лет изоляции мы сделали совершенную копию продукта конкурентов. К моменту поставки продукта лишь немногие интересовались им, и совсем единицы были готовы купить.

Конечно же, команды не выигрывают от вмешательства враждебного руководства, но команды сталкиваются с препятствиями, которые не могут преодолеть самостоятельно - в этом им необходима помощь руководства с необходимой властью. Это называется менеджер-рефери, что представляет собой более ответственную форму наделения властью, при которой руководство следует всем трём правилам описанным выше, но и уделяет достаточно внимания тому, чтобы среда была безопасна для команды. Менеджер должен использовать свою власть и политическое влияние, чтобы те, кто вне команды, играли честно и дали команде шансы на успех. Например, он должен справляться с бюрократией, выбивать достаточный бюджет и отбиваться от конкурентов и саботажников. Таким образом, вместо того, чтобы направлять свою власть внутрь на контроль и командование, менеджер-рефери направляет свои силы наружу, чтобы поддержать команду, удаляя препятствия с их пути.

Группа компаний из Люксембурга Альфа наняла меня для управления некоторыми их проектными командами. В компании считалось, что менеджер не начальник, а члены команды - не подчинённые. Менеджер существует не для того, чтобы его боялись, а для того, чтобы его уважали и доверяли ему. Альфа видит роль менеджера в удалении препятствий, на которые натыкаются команды в ходе проекта, и в постоянной помощи командам стать более эффективными в работе.

Пока я там работал, я заметил, что у многих клиентов Альфы были свои огромные отделы разработки приложений, но они всё равно не справлялись с некоторыми своими проектами и просили Альфу спасти их. Когда я пытался понять причины, то обнаруживал, что в тех командах был типичный враждебный менеджмент. По-моему, Альфа преуспевала не потому, что нанимала более умных людей, а потому, что устанавливала более эффективные отношения между менеджментом и командами. Расчищая дорогу своим командам менеджеры из Альфы помогали им делать работу быстрее меньшим числом людей.

Менеджер-Лидер

Титул менеджера даётся свыше и придаёт официальную власть над командой. А позиция лидера обычно даётся командой. Она отражает их доверие человеку в том, что у него есть способность и желание помочь команде процветать.

Нельзя навязать лидера команде. Если высшее руководство назначают лидера в команду, ему всё равно придётся заслужить отношение к нему, как к лидеру. Если лидер окажется неэффективным, то команда его отвергнет, а вмешательство руководства в этот процесс может сделать из лидера саботажника.

Лидер помогает команде изнутри. Лидерство означает помощь команде в нахождении общих взглядов, вокруг которых можно сплотиться, и помощь в повторении успехов в работе. Успех тут очень важен. Лидер, который просто помогает команде делать всё, что они хотят, вовсе не ведёт их. Лидер должен помочь команде стать самокритичной, и постоянно следить и переосмыслять, а где необходимо, полностью менять их взгляды, чтобы добиваться результатов, имеющих смысл для окружающей среды. Пока среда не удовлетворена, на команду будет оказываться растущее давление. Когда лидер поможет команде преуспевать в работе, среда будет поддерживать команду, а команда будет процветать.

Наиболее эффективный метод способствования лидерству в моём понимании - это когда высшее руководство внедряет карьерный рост, поощряющий лидерство, вместо навязывания старшинства согласно иерархии профессионального менеджмента, где отчёты идут наверх, а приказы вниз. Лидерство означает не контроль над командой, а больше ответственности перед ними. Когда существует лишь карьерный путь профессионального менеджмента, то и продвигаться люди могут лишь по линии менеджмента, поэтому они стремятся впечатлить своих менеджеров, будто они сами хорошие менеджеры. В результате они могут потерять связь с командой, которой должны помогать в производстве качественной продукции, и заразиться эгоистичным индивидуализмом, что подходит им самим, но вредит общей эффективности команды.

Много лет назад я работал на AT&T в исследовательской лаборатории. Мне объяснили, что во многих организациях только самый младший персонал делает непосредственно исследования и разработку продуктов, поскольку все старшие ставятся перед выбором: застояться или перейти в профессиональный менеджмент. Таким компаниям сильно не хватает опытных людей, поскольку нет карьерной линии, которая могла бы их воспитать. В результате у них продукция среднего качества, и периодически для вдохновения и помощи приходится прибегать к услугам дорогих консультантов.

У нас же лидером исследовательской группы был старший инженер консультант, он же исследователь. Эта позиция подчёркивала его обширный опыт и способности в исследованиях и приравнивалась к старшей менеджерской позиции по рангу и зарплате. Вместо того, чтобы погрязнуть в бюрократии, вроде учёта затрат, эта лидерская линия карьеры позволяла опытным людям проводить большую часть их времени над улучшением качества исследования.

Таким образом, в AT&T лучшие люди могли преследовать лидерскую карьеру и продолжать заниматься исследованиями, а не переходить в чистый менеджмент. Не удивительно, что их лаборатории были лучшими в мире, если лучшие люди занимались исследованиями, а не административными задачами.

Власть

Враждебные менеджеры полагаются на жёсткую власть - кнут и пряник - потому что это единственная власть, которая у них есть над командой. Менеджер-рефери тоже полагается на них, но чтобы контролировать среду от имени команды. Менеджер-лидер не полагается

на жёсткую власть - вместо этого он пользуется более мягкими методами, помогающими команде контролировать себя.

Разницу между жёсткими и мягкими видами власти можно увидеть в пяти формах организационной власти, идентифицированных в 1954 году двумя социальными психологами French и Raven.

Первые три формы власти являются жёсткими. Они активно используются менеджерами в иерархии командования и контроля. Жёсткая власть основана на той идее, что работники должны подчиняться приказам, спущенным сверху менеджерами, контролирующими их зарплату и возможности продвижения по службе:

Законная власть: Это когда статус и авторитет основаны на вашем звании. Команды вынуждены делать что-то, потому что их босс так сказал: Менеджер отдаёт приказы, а подчинённые выполняют.

Власть пряника: Тут менеджер предлагает награду, желанную для команды. Менеджер выдвигает требования, а команда следует указаниям, чтобы повысить свои шансы на бонус, повышение зарплаты или продвижение в карьере.

Принудительная власть: Менеджер раздаёт приказы, а члены команды могут быть наказаны за неподчинение. Менеджеры запугивают людей понижением статуса, заморозкой зарплаты или даже потерей работы, если они не будут делать то, что им сказано.

У жёсткой власти всё ещё имеется существенная польза в случае поддерживаемых команд, так как она делает среду безопаснее для команды, что мы видели в отношениях с менеджером-рефери. Однако, преобладание жёсткой власти для контроля команд сильно снизилось за последние десятилетия.

Теперь всё меньше полагаются на ручной труд, для которого жёсткая власть неплохо подходила. В наше время творческие умы рабочих стали основными средствами производства вместо заводов и земель. В частности, в индустрии разработки приложений большая часть работы состоит из интеллектуальных усилий и требует глубоких технических познаний, высоких творческих способностей, а также способности быстро учиться и адаптироваться. От сотрудника зависит, сколько своих умственных способностей он вложит в свою работу.

Если работники умственного труда захотят покинуть компанию, они заберут с собой и средства производства. Поэтому наиболее способные работники ценятся больше всего. Компании соревнуются друг с другом , чтобы привлечь и удержать лучших из лучших. Поэтому часто сотрудники начинают думать “работодатели нуждаются в нас больше, чем мы в них”, и часто они правы.

Для некоторых авторитарных менеджеров это может быть горькой пилюлей. И тем не менее эти менеджеры не могут использовать свою власть, чтобы заставить работников умственного труда работать упорнее, и уж точно нельзя заставить быть изобретательнее. Приказы, подкреплённые угрозами, обречены на плохое исполнение; они не могут надеяться на энтузиазм сотрудников, творчество и преданность, которые так необходимы, чтобы обойти конкурентов. Из-за этого баланс власти сместился к более мягким типам власти, образующим основу лидерства:

Власть эксперта: Для решения специфических проблем всегда нужны наиболее талантливые и опытные люди. Иерархия тут не имеет значения. Чаще всего, профессиональный менеджер менее всех осведомлён о том, как решить данную задачу, и вынужден полагаться на опыт команды. В отличие от авторитарного менеджера лидер знает, как выполнять работу своих подопечных, и лидером стал, благодаря своим заслугам. Лидер понимает работу своей команды, замечает, когда команда в затруднении, может помочь им в трудную минуту и может научить и направить их в нужное русло. Лидер постоянно стремится расширить свои знания и развить способности, относящиеся к работе и к лидерству.

Власть уважения: В этом случае уважение к человеку проявляется за его заслуги и достоинства, а не за титул. Лидер, являющийся кумиром, разделяющий и уважающий ценности команды, быстро добъётся их преданности и желания сотрудничать. Команда будет знать, что лидер всем сердцем предан своей команде, и не захотят его подвести.

Балансирующая Власть

Когда команды говорят “нам нужно меньше менеджмента и больше лидерства”, они имеют ввиду, что враждебный тип менеджмента может усилить личность менеджера, но ослабит организацию вцелом.

Желание получить больше лидерства означает, что нужно поддержать команду, а не исключить менеджмент. Для этого необходимы две поддерживающие роли - менеджер-рефери и менеджер-лидер, которые в совокупности поддерживают баланс авторитета с другими формами власти. Если полагаться только на лидера, то появляется риск угроз со стороны среды, с которыми не всегда могут справиться более мягкие виды власти. Положившись лишь на рефери команда остаётся без возможности сплотиться, а потому работа будет не так эффективна. Вместе же лидер и рефери могут помочь команде выработать достаточную внутреннюю силу и защиту от среды, чтобы успешно работать как единое целое.

Увы, в организациях с преобладанием авторитарного приказного менеджмента будет очень трудно перейти от диктаторства к балансу власти рефери и лидеров. Высшее руководство нескольких компаний спрашивало меня: “Мы согласны, что нам это нужно, но на практике, что мы будем делать со всеми менеджерами, которые у нас есть?”

Разумеется, компания столкнётся с большим сопротивлением со стороны авторитарных менеджеров. Они почувствуют растущую угрозу и станут цепляться или даже увеличивать свою власть в организации.

Нельзя ожидать, что команды и их лидеры сами справятся с этим, потому что мягкая власть всегда проиграет политическую битву с авторитарной жёсткой властью. Единственный вариант, который сработал на моей практике, - это когда высшее руководство, будучи заинтересовано в изменениях, помогло преодолеть трудности, пользуясь своим авторитетом.

В то же время старшее руководство должно постоянно убеждать профессиональных менеджеров в том, что лидеры не подрывают их авторитет, а работают параллельно. Кому-то это понравится, ибо откроются новые возможности для способных людей; особенно тем, кто был разочарован необходимостью подниматься по менеджерской лестнице, отрываясь от технической работы, которая им так нравится. Я даже видел, как такие менеджеры переключались на роли лидеров, и часто с хорошими последствиями,

поскольку теперь они знают работу и нужды менеджмента и продолжают работать в своей технической среде. В таких ситуациях те, кто умеет быть лидером, а не диктатором, получит себе силу всей своей команды.

Те, кто решит остаться профессиональным менеджером, а не лидером, может предпочесть стать менеджером-рефери. Это, конечно, уменьшает их прямую власть над командой, но фокусирует их на поддержке команд и лидеров, используя своё политическое влияние для удаления препятствий с дороги производительности команды. Хоть некоторым это может быть не очень удобно, кому-то же, кому не хватает лидерских задатков, это поможет сохранить свой статус менеджера и сделать существенный вклад в развитие компании.

В случаях, где менеджеры не хотят или не могут стать лидерами или рефери, важно понимать, что пока мы рассматривали только властные отношения между командами и менеджерами. Но ведь существуют и другие важные роли, не связанные с властью; например, административный менеджмент, к которому в своей карьере приходят некоторые менеджеры.

Что такое Менеджер?

Со временем команда будет судить навязанные ценности и менеджера на предмет того, как эффективно они помогают команде выполнять задания и поддерживать хорошие внутренние и внешние отношения. Если команда работает успешно, то она примет ценности менеджера и признает его лидером команды. В противном случае, команда будет беспокоиться о своём выживании, откажется от менеджера как от лидера, сплотится вокруг взглядов и ценностей, которые более эффективны по их мнению, и найдёт себе другого лидера (обычно кого-то своего уровня).

Разумеется, властолюбивый авторитарный менеджер не отдаст свою власть просто так, а попытается использовать свой авторитет для восстановления контроля. Даже если менеджер выиграет этот бой - это будет ненастоящая победа. Выиграв несколько битв менеджер может серьёзно повредить отношениям с теми, на кого он больше всего полагается, - с командой - а, значит, в итоге он проиграет войну. Установив враждебные отношения с командой, с помощью власти можно добиться подчинения, но никак не преданности.

Стандартное мнение многих враждебных менеджеров: “босс лучше знает” и “я нужен подчинённым больше, чем они мне”. Прав менеджер или нет, но такое отношение ведёт к конфронтационным отношениям с командой, где ему приходится полагаться на принуждение, чтобы добиться повиновения.

Альтернативой враждебному менеджменту является никак не отсутствие менеджмента, а скорее поддерживающий менеджмент, помогающий команде добиться успеха, а не втягивающий всех в расточительную борьбу за власть.

Поддерживающий менеджмент бывает двух типов: Рефери, который обеспечивает безопасную среду для своей команды, и Лидер, помогающий команде быть эффективной в имеющейся среде. Они оба необходимы для слаженной работы команды. Когда команда не тратит время и энергию на борьбу с враждебным менеджером, а, наоборот, получает помощь от опытного рефери и эффективного лидера, то вся энергия команды будет направлена на генерирование идей, помощь компании в разработке классной продукции и выбивании опоры из-под ног конкурентов.

Враждебного менеджера команда будет считать некомпетентным и будет хотеть, чтобы тот не вмешивался в работу, а значит вместо сотрудничества получится сопротивление. А потому, не удивительно, что команды не хотят работать с такими менеджерами.

Менеджер, поддерживающий команду, будет награждён полным и добровольным сотрудничеством. Потому-то такой вид руководства является наиболее выгодным для команды: менеджер-лидер может помочь команде выработать способ объединения. Аналогично, менеджер-рефери выгоден команде, потому что защищает её от внешних угроз.

Глава 7: Корпоративная Культура

Столкновения Культур

В корпоративной культуре всегда есть доминирующая метафора, определяющая то, как люди смотрят и реагируют на разные ситуации. Культура, основанная на конвейерной метафоре, вряд ли наймёт человека, который всем сердцем верит в необходимость удовлетворения клиента. Компания, в которой доминирует метафора предварительного моделирования, не сможет найти место для фаната конвейерного производства.

Вы можете биться головой о стену из-за того, что люди не хотят меняться. Вы знаете, что для них лучше, но они вас разочаровывают. Ведь для них разочарованием является то, что вы пытаетесь заставить их свернуть с прямой проторенной дороги на опасную неизведанную тропинку.

Люди спорят на разных уровнях метафор. Нет смысла спорить о том, почему время это качество, с кем-то, кто не верит, что разработка приложений - это архитектурный дизайн. Такой человек отвергнет вашу методологию или культуру, если она не совпадает с его доминирующей метафорой. Если же его доминирующая метафора совпадает с вашей, то он может не соглашаться с вами на более низком уровне, но по крайней мере у вас есть точки соприкосновения культур, на которых можно основывать обсуждения.

Когда вы меняете методологию, вы заодно навязываете культуру, к которой она относится. Если эта культура конфликтует с доминирующей в данное время культурой, то вы столкнётесь с сопротивлением, поскольку все смысловые интерпретации тоже будут конфликтовать. Следовательно, смена методологии - это не только навязывание новых правил, но и смена культуры на поддержание метафор, составляющих новую методологию.

Навязывание команде методологии, которая отражает другое мировоззрение, создаст конфликт по всем направлениям, и, разумеется, столкнётся с заметным сопротивлением.

Как мы увидим позже, изменение культуры не означает изменение поведения. Если вы это сделаете, то превалирующие культурные метафоры либо потянут за собой поведение, либо создадут такой внутренний конфликт, что прогресс просто остановится. Чтобы культурные изменения продержались, нужно изменить целый ряд культурных предположений. А это означает изменение метафор, лежащих в основе мировоззрения доминирующей культуры, на новые.

Сконструированный Мысленный Образ

Когда нас захватит водоворот метафор, то часто хочется посвятить других в свои замечательные открытия. Мы проводим заседания, делаем презентации и пишем статьи и книги о методологиях, наполненными нашими мыслями. К сожалению, они оказываются не настолько впечатляющими, как нам хотелось бы.

Метафоры не так полезны, когда они и их следствия записаны, а скорее, когда они в вашем разуме, впитаны как многогранные мысленные модели, из которых вытекают

дальнейшие умозаключения. Психологи обнаружили, что наиболее эффективные метафоры довольно скупы на словах, но имеют сильное образное значение.

На одной презентации выступающий сказал, что “гибкая разработка приложений (agile software development) включает в себя координирование действий разработчиков так, чтобы завершить пункты из списка требований к продукту, которые были приоритезированы в терминах того, как заказчик обозначил их важность”. Так продолжалось некоторое время, и я явно заметил в аудитории несколько человек, потерявших интерес. Слишком много было слов, слишком уныло. Куда лучше было бы со слайдом из трёх слов, но с более действенным мысленным образом: “непрерывное удовлетворение заказчика”, с паузой после него, чтобы аудитория подумала о смысловых интерпретациях самостоятельно, прежде чем докладчик продолжит.

Сильные мысленные образы очень важны для быстрой и действенной работы метафор.

Психологи неоднократно продемонстрировали, что слабые метафоры разум воспринимает медленно и неохотно. Это больше напоминает заучивание наизусть алфавита. Поскольку образность алфавита довольно низка, то его довольно трудно запомнить с первого раза. Да и потом он легко вспоминается лишь в заученной последовательности. Повторить алфавит задом наперёд куда сложнее. Также психологи показали, что метафоры с сильными образами запоминаются быстро и надолго, осмысляются глубже, вспоминаются легче и в разных трактовках и поддерживают большое число возможных смысловых следствий.

Но ещё важно то, что эффективность каждого мысленного образа очень обособлена. Нет смысла навязывать ваши собственные мысленные образы метафоры кому-то, для кого это пустой звук.

Это может объяснить, почему эксперты по метафорам обнаружили к своему удивлению, что рисунки не помогают доносить смысл метафор до других. Похоже, что процесс рисования сам по себе является процессом обучения, а если не пройти через этот процесс самому, то рисунок может принести больше вреда, чем пользы. Есть даже исследование, доказывающее, что рисунки часто являются контрпродуктивными и приводят людей к интерпретациям, о которых вы и не думали.

Так что не навязывайте свои художества людям, если они вас об этом не просят. Рисунок, насыщенный мысленными образами, может показаться кому-то бессмысленным. Я заметил, что лучшее время для рисования - это когда вы стоите с кем-то рядом у доски и прорабатываете метафоры вместе. Вы оба будете схватывать на лету все возникающие образы, рисуя, чиркая, стирая и перерисовывая до тех пор, пока метафоры не перейдут, куда надо - в головы, а не на бумагу или доску.

Итак, эффективная метафора должна быть описана так, чтобы в головах аудиенции началось построение личного яркого, запоминающегося мысленного образа, построенного на фоне индивидуальных опыта и знаний каждого слушателя - сильный личный образ, стимулирующий воображение.

Изменение Культур

Теперь представьте, что вы хотите изменить имеющуюся корпоративную культуру, например, с метафоры о конвейере на метафору о построении моделей. Проблем будет куда больше, чем при найме человека с конфликтующей культурой. Если сегодня все в

компании живут и дышат метафорой о конвейере, то неизбежно столкновение культур с каждым.

Вам не избежать того факта, что изменить культуру очень трудно. Нужно всех в компании перенастроить на новую метафору. Сказав это мы можем чётко определить шаги и многие ошибки, которые можно допустить и которых нужно остерегаться. Следующая глава в точности излагает, что нужно делать для изменения корпоративной культуры, чтобы вся организация последовала за вами.

Глава 8: Изменение Корпоративной Культуры

Тяжёлые Изменения

Компании могут изменить свою культуру. Я видел, как это происходило. Но, несмотря на заявления некоторых, не так легко изменить корпоративную культуру. Это означает изменение доминирующей метафоры компании, а это означает изменение не только поведения; это ещё и изменение взглядов и даже ценностей. Для этого организация должна хотеть, быть готова и способна подвергнуться существенным преобразованиям. Это требует серьёзной решимости и часто влечёт долгое и болезненное привыкание.

Многие компание просто не готовы или не хотят или неспособны пережить всё то, что подразумевает под собой изменение культуры. Многие говорят, что хотят изменить культуру, но на поверку оказывается, что они хотят результатов без усилий. Им хочется побыстречку сделать лишь кое-что. В лучшем случае, они добиваются небольших местных результатов, но это часто происходит против основной линии компании, а потому результаты почти всегда разочаровывают. А со временем они возвращаются туда, откуда начали.

Мне позвонил исполнительный директор одного очень известного доткома. Назовём его Питер. Он был в городе и хотел позвать меня на обед. За мексиканским стейком с красным вином Питер сказал мне, что он нервничает. «Энтони, наш главный конкурент захватил весь наш бизнес за последние два года. Мы теперь лишь пытаемся угнаться, а не лидируем. Чем больше мы стараемся, тем больше отстаём. Что они делают такого, что мы не делаем?»

Я хорошо знал конкурента. Я провёт некоторое время, исследуя их способ работы, а потому мог ответить Питеру сразу: «Они используют метолику Lean», - начал я, и стал объяснять более подробно. Питер остановил меня через минуту: «Замечательно. Мы тоже хотим этим заняться. Мы хотим быть быстрее всех и исключить растраты. Но мы не можем делать это так, как ты описываешь. Мы должны сохранить все хорошие практики, которые мы уже используем.»

Через несколько дней он связал меня с Дэйвом, своим коллегой. «Мы сделаем это!» воскликнул Дэйв, «Но нужно быть практичными. Мы не можем подорвать дисциплину. Мы не будем делать ‘официальный Lean’. Вместо этого мы будем более агрессивными по отношению к крайним срокам.»

До меня дошло, что Питер и Дэйв боятся изменений. Я же больше боялся, что они вообще не собираются ничего менять. Всё-таки конкурент выбивал из них дух. Инстинктивный страх Питера и Дэйва почти наверняка проистекал из их глубоко укоренившейся метафоры конвейерной линии. Они свято верили в определённый процесс со строгим контролем со стороны руководства, который они считали абсолютно правильным. А ослабление дисциплины рассматривалось как начало хаоса, опасного для компании. Прежде чем начинать изменения, Питеру и Дэйву нужно было самим поверить в них, а это означало бв принятие того факта, что изменения будут сложными, болезненными и, да, пугающими.

Я часто объясняю клиентам, что культура, подкреплённая нежеланием пройти через болезненные изменения, как резинка. Вы можете растягивать её, пока она не порвётся от натяжения, или вы можете сдаться, отпустить её и смотреть, как она возвращается к первоначальной форме.

Дёрганья просто не дадут тех результатов, которые нужны компаниям. Изменение корпоративной культуры с одной метафоры на другую - это огромное преобразование. Как любое большое изменение, оно не произойдёт случайно и не будет лёгким. Для этого необходимо спланированное усилие. Пока я видел успех лишь там, где присутствовало желание людей в организации - от верхов до низов - изменить культуру во что бы то ни стало.

Некая компания наняла меня, чтобы изменить культуру с конвейерной на сервисную. Компании повезло: старшее руководство осознавало наличие проблемы, как и большинство сотрудников. Они не жаловались на плохое следование правилам - они видели потребность в изменении правил.

Когда я прибыл, то увидел, что аналитики сидят в противоположном конце здания от программистов и обмениваются документами туда-сюда. Все выглядели удручённо, и работа шла медленно.

Один из аналитиков пожаловался мне, что программисты то ли не могут, то ли не хотят придерживаться спецификаций. Я пошёл в соседний отдел, где оба программиста излили мне душу, рассказав, что аналитики подсовывают спецификации им под дверь, избегая общения. И те и другие казались нормальными ребятами, но почему-то застряли в этом бою. Аналитики свято верили в конвейерное производство, а программисты были сторонниками построения моделей. У них просто отсутствовала общая идея, вокруг которой можно было бы сплотиться. А потому они разделились на мы-и-они.

Первым делом я посадил всех в одной комнате. В течение первой пары недель дважды в день мы обсуждали текущие проекты и отмечали важные вещи на доске. Я пытался постепенно изменить образ мышления с «я» на «мы», чтобы все были вовлечены, информированы и задействованы.

В течение следующих нескольких месяцев мы постепенно сместили фокус с производства на отслеживание меняющихся нужд заказчика. Это включало в себя изменение как поведения, так и языка. Людям нужно было научиться перестать спрашивать, что нам нужно делать, и начать спрашивать, что больше всего необходимо заказчику. Им пришлось научиться смотреть наружу.

Один аналитик не мог смириться с такими изменениями. Он не хотел или не мог отказаться от конвейерного производства, согласно которому его спецификации говорили программистам, что нужно делать. Ему казалось, что совместная работа лишь трата времени. А потому, по обоюдному согласию, он покинул компанию. Большинство же поначалу испытали некий дискомфорт, но со временем приняли изменения.

По мере изменения языка и поведения люди поменяли и взгляды. Вся компания постепенно изменила культуру с заводской на сервисную. С изменением корпоративной культуры изменилась и мораль. Через несколько месяцев один сотрудник сказал мне, что впервые за несколько лет с удовольствием идёт на работу. В компании установилась более приятная атмосфера, проекты стали завершаться успешно, увеличилась производительность, и компания выросла.

Три Важных Шага

Для изменения корпоративной культуры недостаточно одного желания. Потребуются три вполне чётких шага:

- Приготовиться к изменениям: подготовить организацию, чтобы инициатива была не просто пустым лозунгом
- Провести изменения: протолкнуть изменения во что бы то ни стало
- Закрепить изменения: убедиться, что новая культура пропитала всю компанию, и никто не пытается вернуться к старым привычкам.

Все три шага важны. Я видел много компаний полных энтузиазма, готовых измениться, начинающих со второго шага, забывая первый и третий. Это всегда ошибка.

Пропущенный первый шаг подготовки компании почти гарантирует, что новая культура даже не начнёт приживаться.

Пропущенный третий шаг позволяет новой культуре забыться. Чтобы новая культура встроилась достаточно глубоко в структуру компании и не дала старой вползти обратно, нужно немалое время.

Восемь Классических Ошибок

John Kotter из Harvard Business School возможно самый большой в мире эксперт по изменениям. Коттер провёл года, работая с компаниями, проходящими глобальные трансформации, и нашёл восемь классических ошибок, допускаемых компаниями, каждая из которых может нарушить ход изменений.

Книга Коттера Leading Change описывает эти восемь ошибок и настаивает на том, что каждой из них нужно уделить внимание по очереди - и в приведённом порядке - если хотите, чтобы изменения были успешны. Если поддаться давлению и перескочить через любую из них или перейти от одной к другой слишком рано, то изменения могут не закрепиться.

Вот восемь классических ошибок:

- Слишком много самоуверенности
- Неспособность создать достаточно сильную ведущую коалицию
- Недооценённая сила видения
- Недонесение видения
- Препятствия на пути нового видения
- Неспособность добиться краткосрочных побед
- Преждевременное утверждение победы
- Незакрепление изменений в корпоративной культуре

В моей работе с клиентами я сталкивался со всеми восемью ошибками и сам видел последствия каждой из них. За прошедшие несколько лет я убедился, что фокусирование на этих классических ошибках даёт компаниям замечательный план для изменения корпоративной культуры с одной метафоры на другую.

Давайте посмотрим на восемь классических ошибок, сгруппированых в три важных шага, необходимых для действенного культурного изменения.

Шаг 1: Подготовка к Изменениям

Помните, культуры конфликтуют, а доминирующие взгляды нынешней культуры будут сопротивляться взглядам той, которую хотите ввести. ДЛя уменьшения сопротивления необходимы серьёзные приготовления. Только когда организация готова изменить культуру, тогда она будет способна изменить её. Первые четыре ошибки являются симптомами организации, неготовой к осуществлению изменений. Такие организации нужно немного подогреть.

Ошибка 1: Слишком много самоуверенности

Самая большая ошибка - это делать изменения с места в карьер, не создав чувства необходимости в компании. Если большинству кажется, что нет особой необходимости в смене культуры, или менеджеры плодят фальшивое чувство безопасности, то не будет достаточной энергии для осуществления изменений.

Три года назад компания, разрабатывающая программные приложения, попросила меня помочь “перенести бизнес в современный мир”. Они быстро теряли заказчиков, а прибыли уменьшались. Покопавшись немного, я сделал несколько рекомендаций, чтобы перевести компанию на культуру удовлетворения заказчика, и все с энтузиазмом закивали и согласились с проведением изменений.

Однако сперва ничего не случилось. Я немного подтолкнул их. Один из партнёров разослал письмо, зазывая добровольцев, заинтересованных в участии. Пара человек проявила интерес, но скоро отказались. Я попинал народ ещё немного. Без результата. Наконец, один из партнёров отвёл меня в сторонку и объяснил: “За последние двадцать лет мы не раз сталкивались с проблемами и как-то их всегда преодолевали. Скорее всего через пару месяцев всё вернётся на круги своя и ситуация устаканится”. Без ощущения срочной необходимости не было и рвения к переменам.

Два с половиной года спустя ситуация стала критичной: компания была близка к банкротству. Они наняли новое высшее руководство с указанием делать всё необходимое для спасения компании. Это включало найм двух моих сотрудников на полную ставку для внедрения изменений, рекомендованных мной почти три года назад.

Ситуация исправилась, но почти в последний момент. С существенно большим чувством необходимости и меньшей самоуверенностью компания могла избежать ненужных головных болей, сэкономить время и быть сейчас в куда лучшем финансовом положении.

Между прочим, чувство срочной необходимости не то же самое, что беспокойство; это сильное желание пойти дальше пустых разговоров и начать действовать.

Мой бывший коллега Родриго работал в компании, чей директор трубил о неминуемом изменении бизнеса с диктаторского на партнёрский. Компания теперь искала инноваций.

Звучало это здорово, а энтузиазм и мораль персонала были на высоте как никогда. Пока они не увидели исполнение видения: коробка для предложений, в которую они могли кидать свои инновационные идеи. Был назначен приз за лучшую идею месяца. Родриго выиграл пару раз. Ни одна из победивших идей так и не была воплощена в реальность. Вскоре Родриго перестал напрягаться, как и все остальные. А смысл?

Со временем коробка для предложений была убрана, что и стало концом великого плана инноваций.

Ошибка 2: Неспособность создать достаточно сильную ведущую коалицию

Существенные культурные изменения невозможны без активной поддержки от нескольких людей на самом верхнем уровне руководства. Отдельные личности, безвластные комитеты и наёмные команды не будут эффективны. Для преодоления главных источников сопротивления, ставящих под угрозу эффективные изменения, нужна сильная властная команда старших руководителей.

Во главе этой команды не должны быть менеджеры, свято верящие в приказы и контроль. Иначе им будет очень трудно менять культуру. Такие менеджеры рассматривают призывы к изменениям как предательство. Они не изобретают, а администрируют. У них есть привычка навязывать статус кво. А когда они меняют что-то, то делают это диктаторскими методами.

Нельзя просто приказать людям измениться. Люди могут выполнять приказы, но это не значит, что они в них верят. Они скорее будут играть роль, а не жить этими приказами. Более чем когда-либо добровольное сотрудничество низов влияет на успех изменений. Чтобы это произошло, людям нужно лидерство, а не менеджмент.

Лидерство легко не даётся. Warren Bennis сравнил лидерство со стаей кошек: людей нельзя подталкивать, их нужно мягко тянуть. Лидеры ищут инноваций, а не администрирования и контроля. Они всегда ищут способы долгосрочного улучшения ситуации. Статус кво их не удовлетворяет, а призывы к изменениям - это хороший повод для исследований. Кроме того, лидеры обычно умеют вдохновлять, вести примером, а не приказывать.

Один из моих бывших сотрудников хотел сменить конвейерное производство с жёстким менеджментом на более мягкий стиль сотрудничества. Менеджеры прикинулись слишком занятыми, а потому назначили двух новых сотрудников руководить инициативой. Эти сотрудники были беспомощны: они ездили по офисам компании с презентациями, призывая людей к изменениям, но от них обычно отмахивались. Когда они предоставили финальный отчёт руководству, то за свои усилия получили корпоративный эквивалент похлопывания по спине: ужин со своими семьями за счёт компании. Малейшие признаки изменений испарились, а сотрудники были переназначены на другие задачи.

Ошибка 3: Недооценённая сила видения

Культурные изменения должны продвигаться с помощью разумного, правильно изложенного, вдохновляющего видения будущего. Это необходимо для ясности, эффективного принятия решений и всеобщего принятия общих направления и целей. Слишком часто компании теряют видение и оказываются поглощёнными детальными сложными планами, а процесс изменений вязнет в бестолковых дебатах и бюрократии.

В главном европейском офисе американской компании в воздухе висело возбуждение. Один из больших начальников прилетел из США, чтобы озвучить видение компании на следующий год.

Я стоял в сторонке и наблюдал за тем, как докладчик начал свою презентацию воодушевлённой аудитории из нескольких сотен человек. Звучало это примерно так: “Я

знаю, вам кажется, что у нас нет долгосрочных планов, но это не так. Мы точно знаем, куда должна идти эта компания. Мы знаем о всех решениях, которые нужно принять. Многие из них уже приняты, и мы составили план принятия оставшихся.” После короткой паузы: “Есть вопросы?”

Первый вопрос был очевиден: “Можете сказать, что это за решения?” На что начальник ответил: “О каждом решении вы узнаете в нужное время. По мере принятия решений мы будем передавать их вашим менеджерам. Ваша работа будет состоять в их выполнении. А теперь, господа, пицца и напитки ждут вас в конце зала.”

Как кто-то может вдохновиться этим?

Пока я ел пиццу и пил колу, я услышал разговоры сотрудников: “Это презентация не видения, а ерунды”, “Он имел в виду “У нас есть планы, но они слишком секретны, чтобы делиться с вами никчёмами”", “Что-то мне домой захотелось!”

Ошибка 4: Недонесение видения

Видение нельзя навязать. Нельзя заставить людей поверить в него, игнорируя их мнение и отмахиваясь от их вопросов. Одна поговорка говорит: “Человек, убеждённый против его воли, остаётся при своём мнении.”

Чтобы завоевать сердца и разумы, важна двусторонняя коммуникация. Не только словесная, но и в виде непрерывного потока действий. Старшие менеджеры должны подавать людям пример. Если они говорят одно, а делают другое, то процесс потеряет силу.

Много лет назад, почти сразу после окончания университета я работал в компании, заявившей всем клиентам “мы ручаемся будущим компании за постоянное стремление к качеству”. Была сформирована новая команда слежения за качеством, чтобы ездить по миру и распространять новость. Сотрудникам раздали наклейки и значки с надписями “Качество - Это Главная Работа”. Качество было всем. По крайней мере на словах. Реальных последствий не было. Это была пустая маркетинговая кампания.

Пару месяцев спустя отдел маркетинга разослал сообщение с информацией, что осталось несколько сотен зонтиков с надписью “Качество На Первом Месте”. Эти зонтики были задуманы как подарки клиентам, но почти никто их не брал. А потому, чтобы уменьшить растраты, каждого из сотрудников обязали купить по одному. Какое оскорбление!

Ещё несколько дней спустя было разослано второе письмо, заявляющее, что поскольку мало кто захотел покупать зонтики, руководство сделало вывод - сотрудники не верят в кампанию борьбы за качество. Ещё бы!

Шаг 2: Проведение Изменений

Если организация готова, то можно начинать изменения. Вопрос только - хочет ли организация изменений? Следующие три ошибки могут навредить эффективности изменений.

Ошибка 5: Препятствия на пути нового видения

Поскольку изменения касаются всех, то и препятствия могут возникнуть где угодно. Может оказаться, что организационная структура мешает изменениям, или система пересмотра зарплат наказывает людей за то, что они изменяют способ работы, а могут и начальники навязывать старые манеры поведения, или недостаточный тренинг не даёт воспринять новшества. Если в организации есть что-то, что не даёт сотрудникам следовать за новым видением, то они будут чувствовать себя бессильными. Нужна всецелая поддержка как словом, так и делом от высшего руководства, чтобы справляться с такими помехами. Это означает переопределение организационной структуры, выработка новых методов оценки труда, обучение сотрудников, и, вообще, делать всё, что необходимо для принятия компанией новой формы.

Два сотрудника моей команды работали на телекоммуникационном проекте в некой компании. Это был очень важный проект, так как для этой компании телекоммуникации были новым бизнесом. Потребности конечных пользователей были неясны, а потому требования и дизайн можно было доработать лишь со временем.

Сверху пришла информация, что этот проект положит начало переходу компании к Agile методикам.

Через несколько дней моих ребят попросили предоставит отчёт о положении дел. Они честно отчитались в проделанной работе и указали, какие детали требуют разъяснений. Увы, их упрекнули в том, что они пустились в дизайн и анализ, не составив детального плана проекта.

Ошибка 6: Неспособность добиться краткосрочных побед

Преобразование организационной культуры занимает время. Без небольших, но постоянных результатов на ранних стадиях процесс может сойти с рельс. Чем гоняться за журавлём в небе и надеяться на результаты, которые могут проявиться со временем, руководство должно всячески способствовать изменениям, которых можно добиться в ближайшие дни.

Пусть результаты говорят сами за себя. Когда люди видят ощутимые результаты, и что все их празднуют, они понимают, что усилия того стоят, процесс получает доверие и набирает обороты.

Однако важно, чтобы результаты и празднования были настоящими. Пыль в глазах может деморализовать тех, кто узнает о фальшивости результатов, и дать ложное ощущение спокойствия тем, кто ещё в неведении.

Несколько лет назад я сидел в одном офисе с тремя людьми, почти два года пытавшимися разработать приложение, которое бы производило сложные финансовые анализы. Глава отдела находился под большим давлением, ибо от него требовали результатов. Он объявил, что теперь компания будет делать ежемесячные релизы, а первый будет в пятницу. Та команда из трёх человек запротестовала, так как у них не было ничего полезного для релиза. “Не беда,” - ответил глава отдела, - “Это будет всего лишь демо-версия. Я найду данные за день до этого, а вам нужно будет лишь сымитировать вычисления.” Команда ужаснулась, но согласилась.

Настал день демонстрации - глава отдела привёл с собой главу компании. Он толкнул небольшую речь о том, какая это важная веха. Команду попросили начать демонстрацию,

которая, разумеется, показала результаты, запрограммированные заранее. Глава отдела сказал: “Через месяц вы увидите ещё больший прогресс!”

Глава компании была безмерно счастлива, так как долго уже ждала хоть каких-то результатов. “У меня сюрприз,” - анонсировала она, и в комнату ввезли огромный праздничный торт и две бутылки шампанского. Она похлопала сотрудников по плечам, пожала им руки и покинула комнату очень довольной.

Настроение троицы упало ниже плинтуса. Они не только праздновали фальшивые результаты, но и подписались под тем же на следующий месяц. Через неделю один из них начал поиски новой работы и покинул компанию ещё пару недель спустя. Второй попросился на другой проект. А третий остался один на проекте, который был закрыт через пару месяцев после горячих дебатов между главой отдела и руководством компании.

Ошибка 7: Преждевременное утверждение победы

Если ранние успехи ведут к преждевременному завершению процесса изменений, то результаты будут неустойчивы. Поспешное объявление победы не даёт новой корпоративной культуре укорениться достаточно глубоко. Привыкание к новым манерам и взглядам может занять несколько лет постоянных усилий, пока они не впитаются во все части компании так, что можно будет объявить об окончательной смене культуры.

Я сам допускал эту ошибку несколько раз. Годами я с сожалением наблюдал, как за несколько месяцев после моего ухода от клиента улетучиваются сделанные мной изменения. Клиенты обычно были довольны моей работой, но сами не справлялись с поддержкой новых процессов.

Со временем до меня дошло, что мы слишком рано заявляли о победе. Мы пропускали важный шаг, на котором должны были убедиться, что изменения укоренились в корпоративной культуре. Теперь я уделяю этому куда больше внимания, чтобы клиенты могли становиться самодостаточными.

По-началу, новые взгляды будут лишь в головах людей. Они будут играть в новую культуру, а не жить ею; они будут делать то, что им сказали, а не то, что считают за потребное. Нельзя считать, что культура прижилась, пока эти новые взгляды не перейдут с уровня разума на уровень эмоций. То есть, основополагающая метафора культуры должна перейти головы в сердце. Только тогда взгляды превратятся в ценности, и лишь тогда люди станут инстинктивно придерживаться новой культуры и преуспевать в ней.

Без постоянных усилий и достаточного времени этот переход от разума к эмоциям не случится. Новые взгляды на уровне разума со временем уступят место старым эмоциональным взглядам и старой культуре со всеми её взглядами и практиками, которые вползут следом.

Классический пример этой ошибки - это проект Chrysler Comprehensive Compensation (C3), где Кент Бекк руководил разработкой новой системы начисления зарплат. Проект был большим успехом. Он стал плакатом для новой методики Кента eXtreme Programming (XP), приведшим к ошеломительно быстрому принятию XP на вооружение во многих проектах и организациях.

Тем не менее, через несколько месяцев после ухода Кента проект C3 был забыт и выброшен. Компания Daimler Chrysler отказалась от XP и вернулась к старым методам работы.

Кент ушёл слишком рано, а проект и компания стали слишком зависимы от его энергии и харизмы. Для самодостаточности им нужно было позволить культурному изменению проникнуть глубоко по всей организации прежде, чем отпускать Кента. Как это обычно бывает, резинка вернула себе свою изначальную форму.

Шаг 3: Закрепить изменения

Если избежать всех семи предыдущих ошибок, то можно добиться перехода нынешнего персонала и руководства на новую культуру. Но бессмысленно предпринимать долгую болезненную культурную трансформацию, если она не переживёт следующего поколения.

Ошибка 8: Незакрепление изменений в корпоративной культуре

Как только организация успешно преобразовала свою культуру, нужно убедиться, что она сохранится от одного поколения к следующему. В любое время в компанию могут прийти новые лидеры и новые сотрудники со своими культурами. Нужно, чтобы они впитали нашу культуру, а не мы их.

Неразборчивость в найме может подвергнуть риску новую корпоративную культуру. При найме новых сотрудников важно иметь новую культуру в виду. Неправильные решения о найме, особенно на высшем уровне руководства, могут привести к появлению сильных личностей, неспособных или нежелающих встроиться в “то, как мы тут ведём дела”.

Иногда люди просто не будут вписываться. В таких случаях лучше просто отпустить их, а не позволять подрывать корпоративную культуру. А уж если наняли правильных людей, то нужно чтобы нынешнее поколение жило и дышало новой культурой и передавало её новому поколению и следующим.

Приложение A: Научная Культура

Введение

До того, как Заводская, Дизайнерская и Сервисная Культуры стали доминировать, была Научная Культура. Она началась в пятидесятых, когда программисты носили белые лабораторные халаты и должны были использовать научные принципы, чтобы результаты были точными и предсказуемыми. Она так никогда и не прижилась в индустрии, поскольку, несмотря на обещания, разработка приложений на практике оказалась куда менее научной. В некоторой степени Научная Культура была довольно сильна на компьютерных факультетах университетов, хотя даже там пошла на убыль.

Дийкстра

Наверное самым активным сторонником Научной Культуры был поздний Эдсгер В. Дийкстра. Многие его работы теперь доступны на его сайте в техасском университете. Их стоит почитать, так как они проливают свет на мотивацию, базовые предположения и подъём и падение Научной Культуры для разработки приложений. Я тут процитирую кое-что из его работ и использую имена, которые использовал он (например EWD268).

1950-е: Семена Идеи

В 1950-х Дийкстра написал программы для компьютерного оборудования, которое ещё не было построено. У него были лишь спецификации железа, и он не мог протестировать программу. Поэтому Дийкстра обратился к математике, карандашу и бумаге, чтобы доказать правильность своей программы.

1960-е: Открытие

Дийкстра был доволен результатом, поняв, что математическое доказательство лучше тестирования. Это привело его к теперь уже знаменитой его фразе в 1969 “Тестирование программ может быть использовано, чтобы показать ошибки, но не их отсутствие! Поэтому состоятельность программы должна быть доказана на основе текста программы” (см. EWD268)

1970-е: Доказательства, доказательства и ещё раз доказательства

Затем Дийкстра посвятил остаток своей жизни развитию и евангелизированию Научной Культуры, которая охватила разработку приложений как математическая деятельность. В 1970-х он ратовал за математическое доказательство корректности программ. Позже это было подкреплено его утверждением, что на самом деле доказательство должно предшествовать программе, формируя её спецификацию.

На самом деле, Дийкстра был против использования метафор для описания или представления разработки приложений. Компьютеры представляют такую “радикальную новинку”, считал он, что “метафоры и аналогии слишком плоские” и не представляют ценности (см. EWD1036). Наоборот, разработка приложений ни на что не похожа и должна рассматриваться обособленно.

Дийкстра был уверен, что разработка приложений состоит совсем не из написания программ для компьютеров. Нет-нет, это лишь побочный эффект. “Типичная ошибка считать, что задача программистов состоит в производстве программ” (см. EWD316)

Скорее “Это внутренний долг каждого занятого в программировании - писать доказательные алгоритмы” (см. EWD316). Основная задача в разработке приложений - это создание математических доказательств того, что ваши программы будут работать правильно, если позже вы надумаете их запустить на компьютере (см. EWD1036).

Конечно, это возвращает нас к собственной метафоре Дийкстры о том, что программирование - это математическая деятельность. Разумеется, Дийкстра не рассматривал это как метафору, глубоко укоренившуюся в Научной Культуре, частью которой он являлся. Для него это был факт. И в этом ничего необычного. Я видел много людей, которые придерживались разных метафор так же искренне, как и Дийкстра, и каждый был готов интерпретировать свои метафоры как непреложный факт. Только другим людям нужны метафоры, потому что они не могут или не хотят видеть абсолютную истину, которую вы видите так ясно. И уж точно они так же думают о вас.

1970-е и 1980-е: Ох, уж эти любители!

Объяснение Дийкстры ранних неудач Научной Культуры было не в том, что программисты в белых халатах слишком сильно пытались быть научными, а в том, что они были недостаточно научны! “Часть вины должна лечь на ранних учёных, вовлечённых в программирование … Большинство из них не перенесли свои научные стандарты качества в программирование, которое стало их основным занятием … Поколение “научных” пользователей подошло к программированию, как к решению кроссвордов, созданных производителем компьютеров, а не как к деятельности, стоящей научной мысли.” (см. EWD924)

В 70-х и 80-х Дийкстра постоянно утверждал, что творчество, инновация и проницательность не имеют отношения к разработке приложений. “Меня сильно удручило упорство целой индустрии в том, что якобы достаточно здравомыслия, когда я знаю, что необходимо научное мышление.” (см. EWD917). Компьютерную индустрию нужно было спасти от самой себя, а погружение компьютерной науки в математику было единственным возможным спасением. Дийкстра решил взять на себя миссию по преобразованию компьютерной науки из ремесла в науку.

Однако, к разочарованию Дийкстры большинство людей в индустрии не проявили интерес к его евангелическому утверждению. Наоборот, они покупали “шарлатанские микстуры”. В конце 70-х он выразил особое возмущение всё большим использованием диаграмм и графических заметок, которые он считал “костылём” для “мыслителей-любителей”. Диаграммы, по его мнению - это “для необразованных и от необразованных” (см. EWD696). “Обществу нужна компетенция и знахарство” стенал он (см. EWD966).

К середине 80-х он начал считать, что индустрия просто не была заинтересована в улучшениях: “Весь их выбор это некомпетентность и нечестность. И это ужасно. Но, пожалуйста, не обвиняйте нас, когда компьютерная индустрия рухнет.” (см. EWD917)

Если бы только индустрия его слушала, ведь “Академически образованный учёный компьютерщик должен уметь программировать хотя бы в несколько раз лучше, чем средний программист или хакер без формального обучения” (см. EWD1157) Увы,

“систематическая дисквалификация компетентности … это изобретение менеджеров, за что они и должны нести всю ответственность.” (см. EWD966)

В частности, менеджеры виноваты в своей очевидной миссии по уменьшению зависимости от талантливых работников с верой, что “для промышленного предприятия вполне подойдут второсортные умы” (см. EWD966). “Любовь организаций к посредственности” особенно видна в менеджменте, поскольку “лучшие профессионалы совершенно не намерены переходить на менеджерские позиции” (см. EWD966).

К концу 80-х, кроме менеджеров, Дийкстра стал обвинять и программистов: “Настоящие программисты не осмысливают свои программы, ибо это не круто. Они предпочитают подмену интеллектуального удовлетворения от неполного понимания, что они делают в своей наглой безответственности, и от следующего за этим азарта от поиска ошибок, которых они не должны были допустить в первую очередь.” (см. EWD1012)

1990-е: Теряя надежду

В начале 90-х Дийкстра почти совсем потерял надежду на переубеждение людей индустрии. Он валил это всё на их неполноценность: “Посредственности не доверяют исключительному человеку, поскольку они не понимают его и боятся, потому что он может делать вещи не подвластные их пониманию” (см. EWD1165)

Он хотел, чтобы наука защищалась. “В течение 40 лет компьютерная индустрия совершенно игнорировала находки компьютерной науки … Есть большая разница между тем, о чём общество просит, и тем, что ему на самом деле нужно … Задача лучших университетов сказать индустрии то, что она не хочет слышать, … чтобы убедиться, что укус академического овода действительно болит” (см. EWD1165)

Тем не менее, он стал ощущать, что всё было безнадёжно: “Бои идут между менеджерами/счетоводами с одной стороны и учёными/технологами с другой … Учёные всё чаще проигрывают, … потому что основной их интерес - это наука, … в отличие от менеджеров, для которых эти бои смысл жизни.” (см. EWD1165)

К середине 90-х он отозвался о “жестоком повороте истории”, когда общество 20-го столетия избегало науку. “Но мы не можем обвинять университеты”. Вина лежит на индустрии из-за того, что она давила на университеты с требованием “не увлекаться … научным образованием, а ограничиться профессиональной подготовкой” (см. EWD1209). К концу своей жизни Дийкстра обеспокоился будущим университетского образования и умолял “пока компьютерной науке не позволяют спасти компьютерную индустрию, нужно хотя бы не позволить компьютерной индустрии убить компьютерную науку” (см. EWD1284)