Словарь ненормативной лексики руководителя

Игорь Ашманов

Артем Попов. © "Ашманов и Партнеры"

Этот небольшой словарик является иллюстрацией к Правилам-2. Здесь собраны типовые высказывания руководителей и/или хозяев компании, которые при настойчивом их повторении приводят к трудностям в управлении проектом и персоналом.

Ссылки по теме: "Правила Ашманова": часть первая

«Правила Ашманова»

Игорь Ашманов

Артем Попов. © "Ашманов и Партнеры"

Внимание, вышло продолжение: Правила Ашманова. Часть 2. Об управлении проектами

Данный свод высказываний предназначен для руководителей, которым волей судеб пришлось заниматься новым для себя делом - управлять тем или иным "программистским" проектом (созданием информационной системы предприятия, разработкой сайта, и т. п.).

Опытному человеку сказанное ниже может показаться набором простых и давно известных истин. Я и не собираюсь претендовать на авторство всех приведенных ниже правил.

Однако начинающие менеджеры программных проектов зачастую не знают простейших вещей - например, того простого факта, что нельзя верить срокам, называемым программистами. Воинские уставы и Правила дорожного движения также выглядят просто, но они "писаны кровью".

Много раз видели мы срыв сроков, провал проектов. Видели бизнесменов, с готовностью вкладывавших деньги в новую технологию, поражающую воображение - без понимания рынка, бизнес-планов и даже примерных результатов и сроков работ. Я и сам совершал множество подобных ошибок за 15 лет работы в индустрии производства программного обеспечения.

Вот эти простейшие вещи и собраны здесь в виде свода правил. Вот самое первое из них:

Первое правило Ашманова. Не бывает технических проблем. Бывают только человеческие, то есть организационные.

Я не даю здесь технических советов относительно управления проектами, правил планирования и документирования, процедур тестирования и выпуска. Обо всем этом написаны горы специальной литературы, в том числе классическая книга Фредерика Брукса "Мифический человеко-месяц".

Однако должностные инструкции и правильные процедуры - далеко не всё. При запуске проекта руководитель в первую очередь вступает в **человеческие отношения** с коллегами, исполнителями, подчиненными. Эти отношения сложны, непривычны и часто могут просто поставить в тупик, если не знать всего нескольких простых правил.

Об управлении программистами

Я лично очень уважаю и люблю разработчиков программного обеспечения - программистов, однако в обращении с ними нужно соблюдать осторожность и определенные правила.

Управление программистами - не магия. Управлять программным проектом может даже гуманитарий. Но для этого обязательно нужно вообще уметь управлять людьми и проектами. Как и в любой отрасли, менеджеру достаточно знать некоторые особенности технологического процесса и не поддаваться "мифам индустрии". Всё остальное зависит от обычного умения менеджера наладить работу.

Мифы. Управление программистами имеет особенности, осложненные мифами и иллюзиями вокруг программирования. Эти мифы охотно поддерживаются разработчиками и продавцами компьютерных услуг. Основной причиной мифов является противоречие между очевидной интеллектуальностью и сложностью работы с одной стороны, и совершенно обычными свойствами персонала и проектов - с другой стороны. Независимость от мифов приходит с опытом и знанием.

Распространенные мифы о разработке программного обеспечения

Миф об уникальной специфике программного обеспечения. Производство программного обеспечения не является особым бизнесом, что бы там ни говорили сами разработчики или продавцы информационных систем. Оно не более особенное, чем пищевая промышленность или косметология. Законы развития и окупаемости проектов при разработке ПО, интернет-сайтов и корпоративных информационных систем - те же самые, что и везде.

Поэтому разговоры об уникальности разработчиков ПО, специфике управления программистским проектом, особых путях бизнеса - всегда очень подозрительны.

Миф о невероятной сложности программирования. На самом деле процесс создания современной программы не сложнее и не более творческий, чем процесс создания современного автомобиля, рекламного ролика или лекарства. В этих индустриях давно наведен порядок, не мешающий творчеству.

Миф о величии программиста. Разработчик программного обеспечения - не оракул, прорицания которого бизнесмен должен слушать с восхищением. Действительно, внешне любой, даже средний программист выглядит, как взрослый, умный и ответственный человек - он высокообразован, занимается сложной умственной работой, владеет терминологией, умеет поставить задачу и обосновать запрос на выделение ресурсов. При этом на деле он постоянно проявляет себя как малолетний двоечник - не говорит всей правды, ошибается со сроками, срывает проекты, сворачивает с прямого пути и развлекается за счет работодателя.

Если в проекте разработчики играют первую скрипку, вероятность краха проекта - высока. Верный признак такого положения - когда менеджер высокого уровня, непрограммист, с заметной неуверенностью рассказывает о том, что всё идет нормально и текущий этап проекта представляет собой важные внутренние технические улучшения, которые на следующих этапах позволят осуществить прорыв.

Нужно помнить, что разработчик ПО - это инженер, а в бизнесе высоких технологий выигрывают не инженеры, а бизнесмены и менеджеры. Как и везде.

Миф о магической силе технологии. Новая, сложная, впечатляющая технология не обязательно приведет к успеху. К счастью, сегодня это уже не нужно никому специально доказывать. Всякая технология может стать успешным продуктом, а может просто привести к растрате денег инвестора. Источники успеха проекта всегда лежат вне сферы технологий - в сфере бизнеса.

Кроме того, нужно помнить, что действительно уникальные технологии возникают очень редко, и с большой вероятностью в настоящий момент точно такая технология уже обсуждается, разрабатывается или даже продается где-то еще.

Правила, которые полезно знать менеджеру

Правило 2. Технический жаргон ничего не значит.

Программисты используют весьма развитый технический жаргон, в том числе на совещаниях и в докладных записках. Известно, что любой жаргон служит узнаванию своих и запутыванию чужих. Жаргон программистов - не исключение, а враждебным чужаком для программиста часто служит его начальство.

На самом деле технический жаргон в большинстве случаев не нужен и не несет дополнительного знания. Всё, что нужно знать о проекте, может быть выражено обычным деловым языком - языком бизнес-плана, функционального описания, сетевого графика, рекламного проспекта.

Правило 3. Разработчики всегда называют неверные сроки.

Нельзя верить срокам, которые называют программисты. Обычно их следует умножать на Пи. Иногда (редко) - делить на Пи. Выбор правильного действия руководителя над называемыми сроками зависит от личности разработчика. Это знание приходит к менеджеру только после нескольких экспериментов именно с этим разработчиком.

Правило 4. Разработчику свойственен врожденный оптимизм.

Средний разработчик всегда **добросовестно** заблуждается относительно трудоемкости задачи, обычно в меньшую сторону, поэтому его нельзя уличить или разоблачить. Более того, он никогда не учится на ошибках и постоянно повторяет одну и ту же ошибку занижения сроков, поэтому его бесполезно стыдить или воспитывать. Категорически нельзя рассчитывать, что уж на этот раз битый жизнью разработчик наконец-то называет реальные сроки.

Типичный признак необъезженного разработчика-оптимиста - самоуверенность и горячность, стремление пойти и сделать, а не сесть и запланировать.

Правило 5. Программист испытывает страсть к обобщению.

Программист всегда всеми силами стремится сделать работу наиболее общим способом, чтобы потом только настраивать и прилаживать готовую систему. В этом - суть программирования и его сила.

И в этом же - серьезная угроза бизнесу. Если дать разработчику волю, разработка общей платформы отнимет 100% времени и денег, и продукт никогда не выйдет на рынок.

Поэтому программирование в наиболее общем виде нужно категорически запрещать. Верным признаком страсти к обобщению является планирование создания мощных ядер и наиболее общих платформ со сроками исполнения больше года.

Баланс между обобщением и текущими требованиями рынка достигается опытом и соображениями бизнеса. Программистам доверять здесь абсолютно бессмысленно, как бессмысленно обсуждать с пьяницей семейный бюджет.

Правило 6. Нельзя делать "по-хорошему".

Всякий программист свою страсть к обобщению оправдывает похвальным желанием наконец-то всё сделать по-хорошему. Точно так же системный администратор всегда просит денег на самую лучшую технику и самое дорогое программное обеспечение от Oracle. Делать по-хорошему - теоретически неправильно и практически вредно для бизнеса. Нужно делать так, чтобы всё работало, удовлетворяло клиентов (ровно на уровне цены продукта) и бизнес развивался.

Верный признак работы в стиле "по-хорошему" - упорная работа с "ядром", задержки с реализацией конкретной запланированной функциональности и срыв сроков выхода продукта.

Правило 7. Приминание травы может отнять любое количество времени.

Программист всегда стремится удовлетворить свою потребность в вооруженности - максимально обустроить рабочее место, то есть создать инструментарий, установить самое последнее программное обеспечение, самую современную технику. Если дать ему волю, он потратит на это 100% рабочего времени, причем сумеет доказать начальству необходимость такой работы.

Верный признак такого перманентного обустройства - полуразобранные компьютеры на рабочих местах и необычные, нестандартные программы на этих компьютерах.

Правило 8. Разработчик не интересуется бизнесом, он - типичный автор.

Разработчик в среднем не стремится помочь организовать и развить бизнес, именно поэтому он не любит тестирования, считает пользователей идиотами, поэтому его трудно заставить документировать и поддерживать уже сделанные системы и программы.

Истинные личные мотивы большинства разработчиков - **авторские**, то есть включают интересную работу, хорошие гонорары и известность. Низовой разработчик может не иметь даже и авторских амбиций, и интересоваться только работой и зарплатой. Успех продукта или компании в смысле роста прибыли интересует разработчика только опосредованно.

Это нормально, так как авторские мотивы - очень мощные и их можно правильно использовать с большой пользой для компании.

Выводы

Приходится работать с тем, что есть. Из сказанного вытекает практическая невозможность перевоспитания разработчиков, системных администраторов и средних технических менеджеров в деловом духе. Этого и не требуется. Вместо перевоспитания нужно понимание мотивов и привычек, и работающие процедуры планирования, исполнения и контроля, сохраняющие свободу мышления и оставляющие простор для творчества.

Схемы мотивации также должны учитывать авторские мотивы и врожденный оптимизм разработчика (например, из этого общего принципа легко выводится недопустимость штрафов за срыв сроков).

Приложение

Словарь ненормативной лексики программиста

Учитесь понимать программистов. "Ну, не знаю, у меня на машине всё работает!" - "Планировать разработку бессмысленно, жизнь всё равно богаче!" - "К пятнице готово не будет, но в понедельник точно. Или во вторник!" - "Программа хорошо документирована на языке С".

Вышло продолжение: Правила Ашманова. Часть 2. Об управлении проектами

«Правила Ашманова»—2. Управление проектами

Игорь Ашманов

Артем Попов. © "Ашманов и Партнеры"

От автора

На первую часть "Правил" поступило много откликов, в том числе критических. Насколько можно судить, больше всего обиделись на мою статью некоторые программисты. Например, были такие

возражения:

- В жизни всё не так, и автор абсолютно некомпетентен.
- Бывает куча технических проблем, а не только организационных. А если менеджер этого не понимает, его нужно гнать в шею.
- Программисты вовсе не такие плохие. Вот наши знакомые программисты не срывают сроков, живут нуждами бизнеса и так далее, вообще абсолютно сознательные и продвинутые личности. А если и бывают неправильные программисты, то их нужно просто заменить и нанять более правильных.
- Наоборот, это менеджеры плохие, некомпетентные, и от их неумного и неумелого руководства страдают умные программисты. Менеджеры, описанные в статье, просто никуда не годны и их нужно уволить.

Ну что тут скажешь! В целом в вышесказанном практически всё правда... для отдельных случаев. Но мои оппоненты ломятся в открытую дверь - ведь я писал не критику на программистов, а разбор острых ситуаций в проекте.

Да, некомпетентный менеджер может отравить жизнь гораздо сильнее, чем программист (именно об этом - вторая статья ниже).

Да, хорошо, если можно просто нанять "внятного" программиста. Вообще хорошо тем разработчикам, которые попали в хорошую компанию или смогли сформировать ее вокруг себя! Оглянешься - благодать! Менеджеры технически грамотны, программисты ответственны, зарплаты большие, планирование четкое, есть простор для творчества, персонал уважают и ценят, продукт продается - что еще нужно, чтобы достойно встретить старость...

Такое и в жизни бывает, особенно в успешных "компьютерных" компаниях, производящих собственные программные продукты. Жаль только, что такой идеал встречается редко. Рассматривать такие случаи не имеет смысла. А средний случай - совсем другой.

Последние полтора года в России происходит массовый поворот мелкого и среднего бизнеса к компьютерам и Интернету. Огромное количество компаний "реального сектора", то есть не компьютерных и не интернетовских, начинают делать себе сайты, строить информационные системы и нанимать для этого программистов.

Этот процесс лавинообразно втягивает в управление компьютерными проектами всё больше и больше менеджеров, никогда этого ранее не делавших и даже не имеющих технического образования. Руководить разработкой складской системы или сайта приходится продавцам, маркетологам, часто - высшим менеджерам и первым лицам компаний. И взять других менеджеров просто неоткуда. В стране наблюдается острейшая нехватка средних менеджеров -

руководителей проектов.

И программисты им также часто достаются не самые опытные и не самые профессиональные. Молодые, недавние студенты. Зрелых и профессиональных мало, и они чаще работают в тех самых "компьютерных" компаниях или уехали на заработки за рубеж.

Таким образом, разработкой часто руководит менеджер, делающий это впервые, и для его программистов это тоже всего лишь первый или второй серьезный проект в жизни. Как сложатся их отношения, выйдет ли из этого что-нибудь - предсказать трудно.

Но можно дать несколько полезных советов - может, хоть один да пригодится в реальной обстановке.

Содержание

Введение - правила одной строкой

При росте объема проектов и количества используемых человеческих ресурсов управление ими становится всё более похоже на сложную инженерную задачу (*которая сама по себе постепенно становится проектом - более высокого уровня*). Для управления большими проектами нужны знание общей теории, детальные бизнес-процедуры, мощные программные средства управления проектами, высококвалифицированные специалисты по управлению. Если читателю хочется стать профессионалом в науке об управлении проектами, нужно читать соответствующую литературу или поступить на специальные курсы.

Однако чаще всего в проекте возникают проблемы не из-за объема или сложности задач, а из-за неумелого управления и запутанных человеческих отношений. В этом случае формальные методы не смогут помочь.

В данной статье я хочу коснуться человеческих проблем, возникающих при управлении относительно небольшим проектом, где применять всю мощь науки просто некогда или незачем (нерентабельно).

Как известно, знание немногих принципов часто заменяет знание многих фактов. Вот выработанные мною для себя самые общие принципы управления проектами:

А. Кадры решают всё.

Б. Ключ к успеху проекта - передача ответственности участникам проекта.

В. Ключевой момент переключения ответственности - принятие решения.

Ниже я подробно поясню, что означают эти принципы и какие правила поведения менеджера из них следуют.

А как менеджеру правильно запустить проект по разработке, спланировать работы, организовать разработчиков, распознать опасные сбои и тенденции в ходе работ, аккуратно завершить работу? Для этого также достаточно знать всего несколько основных, вполне очевидных правил.

Точнее, знать недостаточно (тем более, что они просты и очевидны) - их нужно исполнять. Поразителен тот факт, что многие менеджеры их таки не исполняют. В своей частной жизни они легко подчиняются подобным правилам - свой автомобиль они не забывают заправлять бензином, чинить, мыть, как-то планируют маршрут, по завершении поездки паркуют и выключают автомобиль, ставят его на сигнализацию и т. п. Но почему-то, придя на работу, начинают думать, что программистский проект может выполниться, как по волшебству, сам собой, при нарушении последовательности действий, невыполнении технологических требований, без подготовки, планов и ресурсов...

Вот эти правила.

I. Как правильно запустить проект и закончить его

- Решение о запуске проекта должны принимать ответственные лица.
- Для начала проекта нужно исполнить процедуру запуска проектов.
- Без энтузиаста любой проект мертв.
- Проект нельзя обсуждать без обоснования.
- Не бывает проекта без руководителя.
- Руководитель должен быть один.
- Проект нельзя запускать без плана, написанного лично руководителем проекта.
- Проект нельзя запускать без ресурсов.
- В начале проекта всегда бывает естественное торможение. Чтобы его преодолеть, нужны терпение и настойчивость.
- Вполнакала проекты не делаются.
- Только коррекцией в контрольных точках можно удержать проект от срыва.
- По окончании проекта нужно правильно переключить ответственность.
- После завершения проекта нужно запустить следующий "бессрочный" проект - проект поддержки.

II. Неправильный проект и его лечение

- Риск неудачи проекта есть всегда.
- Важно вовремя узнать "паттерн" неудачного проекта - симптомы надвигающейся болезни.
- Когда в проекте что-то идет не так, начальство предпринимает титанические бесполезные усилия по наведению порядка.
- Бессмысленно муштровать исполнителей, нужно муштровать менеджеров.
- Наведение порядка в проекте всегда должно начинаться с головы.
- После разбора полетов нужно списать все долги и запустить проект заново.

Разберем эти принципы и правила подробнее...

I. Правильный проект: запуск, ведение, завершение

Основные принципы устройства коротких проектов

В качестве примера коротких проектов по разработке ПО можно привести разработку сайта, небольшие внутрифирменные информационные системы для собственных нужд, первые версии программ для внешних потребителей, разработку shareware и так далее.

Такой короткий проект обычно отличается от длинного и большого тем, что:

- проект часто возникает в организации, не занимающейся разработкой ПО профессионально;
- руководитель проекта имеет обычно небольшой опыт разработки, занят и другой, основной деятельностью;
- ресурсы сильно ограниченны: сроки небольшие, людей мало - каждый разработчик и менеджер проекта на счету.

В условиях сильно ограниченных ресурсов нет возможности делать всё по-хорошему: тщательно планировать, использовать специальные средства планирования и управления, методично нанимать нужных специалистов, закладывать достаточные сроки тестирования и документирования.

В небольшом проекте не получается рассматривать персонал как атомы, движение которых подчиняется макро-законам. (Я, впрочем, считаю, что и в большом проекте тоже нельзя.)

В таких условиях **люди** становятся ключевым фактором и никакие формальные "техники" управления проектом не смогут заменить работы с людьми. В условиях дефицита ресурсов приходится иметь дело с теми, кто есть под рукой, и именно отношения с этими людьми будут определять успех проекта.

Отсюда вытекает принцип номер один:

Принцип А. Кадры решают всё.

Автор этого принципа широко известен.

Люди вовлечены в проект настолько, насколько они чувствуют свою **ответственность** за него, поэтому второй основной принцип гласит:

Принцип Б. Ключ к успеху проекта - передача ответственности участникам проекта.

(Предвидя возможные возражения разработчиков, скажу, что энтузиазм и авторские творческие порывы я тоже отношу к чувству ответственности.)

Для того, чтобы возложить ответственность за что-либо на кого-либо, нужно это сделать в явном виде. Назовем это переключением ответственности.

Точками переключения ответственности служат **решения**. Отсюда - третий принцип:

Принцип В. Ключевой момент переключения ответственности - принятие решения.

Например, принятие решения о запуске проекта переключает ответственность за проект на руководителя проекта, а ответственность за выделяемые ресурсы - на его нанимателя.

Руководитель проекта в свою очередь начинает транслировать часть ответственности дальше вниз, принимая решения о подборе и расстановке разработчиков.

Если люди, структура ответственности и решения в проекте выбраны правильно, то и проект будет правильным.

Правильный проект

В понятие "правильного проекта" я включаю самые простые вещи: необходимый минимум действий - не технических, а относящихся к ответственности и взаимоотношениям людей в проекте. А именно:

- правильный запуск проекта;
- правильный контроль;
- правильное окончание или остановка проекта.

Ключевым этапом является запуск проекта, поскольку именно на этом этапе обычно закладываются основные ошибки проекта.

От идеи до запуска проекта

Для начала нового проекта нужны две вещи: начальная **идея** и принятие **решения** об инициации проекта.

Возникновение идей - вещь интимная и иррациональная, составляет предмет философии и психологии, и здесь мы его касаться не будем. Создание творческой обстановки, стимулирующей энтузиазм и появление свежих идей - отдельная большая тема.

Рассмотрим принятие решения о запуске, потому что, казалось бы, это вещь совершенно рациональная.

Действительно, принятие решения о запуске проекта - это принятие решения об инвестициях ресурсов (денег, времени, труда) в условиях некоторого риска. Следовательно, это прерогатива владельцев ресурсов, то есть хозяев компании или тех, кому они делегировали право принятия подобных решений (гендиректора, начальника департамента и проч.).

Эти **ответственные** лица должны оценить привлекательность проекта и принять решение рискнуть ресурсами. Таким образом, получаем правило:

Правило 1. Решение о запуске проекта должны принимать ответственные лица компании.

Это правило может показаться трюизмом, банальностью, но я встречал много случаев, когда о существовании проекта гендиректор компании узнавал через полгода после его начала - обычно в момент возникновения проблем или вообще случайно.

Выяснялось, что решение о проекте и, следовательно, инвестициях и риске принимал некто безответственный - кто-то, не несущий ответственности за проекты и расходы - кто-то, с кого даже нельзя толком спросить (разработчик, продавец). То есть имело место пагубное разделение полномочий и ответственности.

Хорошо еще, если вообще кто-то это решение принимал! Бывает, что запуск проекта обходится вообще без принятия решений. Многие проекты в компании растут, как бурьян.

Кто-то высказал классную идею, хозяин/руководитель на бегу согласился, что она классная, кто-то там же в коридоре или курилке походя взялся кое-что сделать, и вот проект как бы начался. Для него нет коммерческих оснований, руководителя, исполнителей, то есть проект по сути нежизнеспособен - но включен в планы компании, является частью ее ментального пространства.

- Я слышал, у вас делается лазерный партнерский медиапортал на воздушной подушке?
- Ну да, мы вообще-то это делаем, скоро будет. Там фишка в том, что у нас практически всё есть - и подушка, и лазеры, только собрать вместе и втюхать партнерам.
- Круто. А когда выпускаете?
- Сейчас не помню точно. Надо спросить у этого, как его... Кто ж там у нас... Ну, в общем, там в разработке знают. Короче, через несколько месяцев. Ну, к сентябрю.

Мертворожденные проекты

Так в компании образуется множество несуществующих, мертворожденных проектов, которые, несмотря на свое несуществование, числятся в активе, попадают в планы, пожирают деньги и силы (см. ниже раздел про **виртуальную реальность**). Начальство считает эти проекты существующими, на их выпуск рассчитывают продавцы, про них рассказывают партнерам и прессе, назначают планы выхода, включают в планы

продаж. Виртуальный проект ведет себя почти как настоящий, только никогда не заканчивается выводом на рынок.

Я не раз видел ситуацию, когда в компании было больше проектов, чем людей (считая уборщицу и секретаря). Ясно, что большинство из них не могли быть живыми. При трезвом рассмотрении половину можно было вычеркнуть сразу, потому что никто в них не был заинтересован, а еще треть также была виртуальной, но оказывала ожесточенное сопротивление, потому что грела чье-то самолюбие или служила чьим-то интересам.

Для того, чтобы не пропускать в ментальное пространство компании подобную нежить, нужна надежная и простая процедура рождения, то есть **процедура запуска проектов**. Она и послужит фильтром для мертворожденных или случайных идей.

Процедура запуска

Отсюда следует второе правило:

Правило 2. Для начала проекта нужно исполнить процедуру запуска проектов.

Процедура запуска весьма проста и включает следующие основные этапы:

- **принятие решения** об инициации проекта **ответственным** лицом,
- написание **обоснования** проекта и оценки затрат на проект,
- назначение **руководителя**,
- принятие **решения о запуске** проекта,
- написание руководителем **плана** работ и **принятие** его начальством,
- выделение ресурсов.

Это совсем небольшой набор вполне очевидных действий, что-то вроде общей гигиены - чистки зубов, мытья рук перед едой и т. п. И настолько же полезный. К сожалению, эта простейшая проектная гигиена многим просто неизвестна.

Нужно обратить внимание на тот важный факт, что при запуске проекта решение принимается **дважды**: первое решение - о том, чтобы рассмотреть и проанализировать идею, и второе решение - о запуске проекта.

На самом деле здесь есть и завуалированное третье решение - принятие плана.

Энтузиасты

Конечно, все эти перечисленные выше события не смогут произойти, если ребенок не будет зачат, то есть если у проекта не будет отца - энтузиаста, предложившего саму идею. Имеет место следующее правило:

Правило 3. Без энтузиаста любой проект мертв.

Энтузиаст должен быть готов преодолеть небольшие барьеры, которые ставит перед проектом процедура запуска - убеждение соратников, написание обоснования, получение одобрения руководства на запуск проекта.

Наличие невысоких формальных барьеров - необходимости написания бумаг и проч. - с одной стороны, и атмосферы поощрения энтузиастов - с другой, создают в компании здоровую систему инноваций и фильтрации мертворожденных проектов.

Обоснование

Естественно, если в организации формальные барьеры слишком высоки (бумаг требуется слишком много, к боссу на прием не пробиться, требуются визы всех директоров и т. п.), то организация будет терять инновационные идеи и расхолаживать энтузиастов.

Но небольшие, невысокие барьеры нужны, потому что если энтузиазм инициаторов и готовность начальства настолько низки, что нет сил даже написать обоснование или назначить руководителя, то проект - не нужен.

Обоснование - первый из таких барьеров.

Обоснование может быть написано тем самым энтузиастом или заказано коммерческому отделу, оно может состоять не из десяти, а всего из двух страниц, но без него нельзя. Если по проекту не удается (или некому) даже написать обоснование, его не нужно начинать. Введем очередное правило:

Правило 4. Проект нельзя обсуждать без обоснования.

Замечу, что обоснование может написать и разработчик, если он является инициатором или сторонником идеи. Не страшно, что он "не понимает в коммерции". Если идея интересная, нужно придать ему коммерсанта для прояснения затрат, рыночных условий, будущей окупаемости.

Заметим, что важен сам факт **наличия** трезвого обоснования необходимости проекта и анализа рисков и расходов, а не доказательство прибыльности проекта. Основания для запуска проекта могут быть различными - это ведь не обязательно прибыль.

Обоснование может быть и таким: "да, сайт заведомо не принесет нам денег, но без сайта уже неприлично, у всех конкурентов есть" или "эта система не позволит больше зарабатывать, но наведет хоть какой-то порядок здесь, здесь и здесь при таких-то расходах в месяц на поддержание" или "диск с демо-версиями будет иметь чисто имиджевый эффект и обойдется во столько-то".

Главное - нужна трезвая оценка затрат и рисков. Руководство должно принимать решение, то есть идти на расходы и риск с открытыми глазами.

К сожалению, очень часто на этой стадии обходятся заменяющими рассуждениями вроде "да это нам почти ничего не стоит", "у нас и так почти всё есть, осталось немного", "сделаем по ходу, параллельно", "да мне соседский парень за 100 долларов всё сляпает", призванными просто замылить вопрос обоснования.

Про *виртуальные* обоснования будущей рентабельности типа: "объем рынка равен полутора миллиардам долларов, если мы с нашим продуктом займем 0,5%, или даже лучше 1,5%, то это будут громадные деньги..." - даже говорить не хочется.

Руководитель

Далее нужно назначить руководителя. Это может быть то самый энтузиаст или другой менеджер, но руководитель должен быть. Если никто не несет **персональную** ответственность за проект, то проекта всё равно что нет. Каждый движущийся по дороге автомобиль должен быть снабжен шофером.

Итак, еще одно очень простое правило:

Правило 5. Не бывает проекта без руководителя.

Хотя руководитель не обязательно должен быть тем самым энтузиастом, который инициировал проект, определенный энтузиазм по отношению к проекту он должен испытывать. Движущие силы его для нас сейчас неважны (карьера, интерес к идее проекта, стремление показать себя, проч.), но руководитель должен иметь **решимость выполнить проект**. И эта решимость должна идти изнутри, а не сверху.

Руководитель проекта должен быть назначен как можно раньше.

Руководитель должен быть один

Здесь нужно обязательно отметить, что много ответственных не бывает. Много бывает только безответственных.

Очень часто можно видеть назначение сразу нескольких ответственных за проект - дело в том, что это самый легкий способ для большого босса быстро решить проблему назначения, не решая ее.

— Руководитель проекта нужен. Кого назначим?
— Да вот хоть Володя с Максом - один продавец, другой технарь. Как раз разберутся. Да, и пусть Пупкин им поможет с маркетингом.

Это верный путь к провалу или созданию "виртуального" проекта-зомби, который как бы движется. А назначенные Володя, Макс и Пупкин думают, что работа как-то там ведется другими двумя товарищами. У семи нянек...

Правило 6. Руководитель должен быть один.

Следует добавить также, что номинальный руководитель (назначенный из уважения или на безрыбье), скажем, большой босс, который в основном ездит по заграницам и переговорам в роли как бы руководителя проекта - еще хуже, чем отсутствующий руководитель или упомянутые семь нянек. Виртуальное руководство приводит к виртуальным результатам.

План и ответственность

Основная функция руководителя - планирование, потому что, во-первых, планирование начинается до всякого управления, а во-вторых, план есть квинтэссенция ответственности.

Ведь план - это не программа действий и событий наподобие программы ТВ, а письменное **обязательство** руководителя, подобное расписке при получении денег в долг.

Аналогия здесь не поверхностная, а глубокая - ведь руководитель проекта берет у компании ресурсы в долг, а отдать должен результатами проекта.

Таким образом, если план - это вопрос обязательств и ответственности, то руководитель без плана - лицо **безответственное.**

Это простое рассуждение объясняет, почему так трудно бывает добиться планов от среднего менеджера. С этим сталкивался любой, кто запускал проекты и назначал руководителей.

Работает менеджер много, энергично, письма шлет ежечасно, а вот планов почему-то не пишет. Казалось бы, ну напиши программу действий, утверди у начальства и действуй, что тут сложного? Но ведь на самом деле это вопрос не программы действий, а ответственности!

Следовательно, здесь требуется моральный выбор и принятие рискованного решения - взятие на себя ответственности. И менеджер, часто бессознательно, уклоняется от этой ответственности.

То есть много работать он готов, клясться в верности проекту готов, делать разные обязательные вещи - тоже, а составить план и принять на себя ответственность - нет.

Интересно, что на старте проекта планы пишутся все-таки достаточно легко. И понятно, почему - проект новый, сроки длинные, до конца проекта далеко, ответственностью пока и не пахнет. Можно не ломать голову, цифры брать приблизительные, никто проверить не сможет, а накажут за срыв или нет - так до этого ж еще полгода! Как-нибудь разрулим это дело.

Именно поэтому начальные планы так неточны.

А вот в середине проекта, уже после срыва первых сроков средний менеджер обычно тихо, но упорно сопротивляется всяким попыткам получить от него следующие - уточненные - планы. Потому что теперь возможное наступление ответственности - близко и явно видно. Поротые на летучке в прошлый понедельник места еще болят.

— Дай уточненный план выпуска бета-версии, наконец! Когда вы *действительно* ее сделаете, а не так, как в прошлый раз!
— Да я прямо сейчас могу сказать, без писанины - там почти всё готово, скоро выпустим.
— Ну так запиши это кратко на бумаге и поставь даты! Когда точно?
— Да за выходные выпустим. В понедельник точно выпущу.
— Вот и пришли мне письмо с этой датой, и там подробно напиши, что именно будет выпущено. Функциональность и недоделки, которые потом исправите.
— Хорошо, в начале той недели во всем разберусь и пришлю план.
— Как это - план в начале недели?! В понедельник ведь уже должна быть бета, ты же только что сказал!!! План мне нужен сегодня!
— Ну, хорошо, хорошо. Только я сегодня кровь из носу должен собрать промежуточную сборку и отдать тестировщикам (исправить ошибку, съездить к заказчику, написать задание для программистов). А план завтра напишу...

Далее - по индукции. Недели проходят, а уточненного плана всё нет... И бета-версия (сайт, продукт, каталог, сборник) всё в том же положении - почти готова.

Итак, если руководитель назначен, нужно срочно сделать его полноценным руководителем, то есть возложить на него ответственность. А именно - потребовать от него план.

Правило 7. Проект нельзя запускать без плана, написанного лично руководителем проекта.

Поскольку ключевой момент переключения ответственности - решение, то **план должен быть принят начальством**. В этот момент ответственность за исполнение переключается на руководителя проекта, ответственность за обеспечение ресурсами - на начальство, принявшее план.

Конечно, есть и другой метод - спуск плана сверху. Начальством задан срок и требования к проекту, дальше - проси ресурсов (а вот дадут ли их и сколько - это уж как повезет) и работай хоть до четырех утра; по крайней мере, уходи домой позже босса.

К сожалению, в такой ситуации из руководителя проекта по существу не получится ответственного лица. Как он может быть по-настоящему ответственным за чужие планы и обещания? Фактически руководитель проекта из свободного человека, обдуманно взявшего на себя обязательства, превращается в задерганного исполнителя, испытывающего чувство безнадежности, потому что ни успех, ни провал от него почти не зависят. И даже щедрые премии в конце проекта самой ситуации со сдвигом сферы ответственности не меняют.

Несколько раз мне приходилось наблюдать подобные ситуации в разработке ПО, и они почти всегда кончались развалом проектной команды и срывом проекта.

Метод спуска планов сверху наиболее сродни сталинской модели индустриализации, когда и жесткие планы, и суровая ответственность наступают неотвратимо, как дождь или снег. И кнут, и пряник в руках у главного босса в этой модели должны быть почти такими же весомыми, как жизнь и смерть. Да, и нужно еще одно условие: исполнитель не должен иметь возможности сбежать. К счастью, в наших условиях свободного рынка труда эта модель не работает.

Простые и эффективные советы, как именно писать планы про разработке программного обеспечения, даются в статье Джоэла Спольски "Painless Software Schedules" (есть русский перевод "Планирование программного обеспечения малой кровью"), так что я позволю себе не углубляться здесь в технические и организационные подробности.

Ресурсы

Правило 8. Проект нельзя запускать без ресурсов.

Ну, это же вообще очевидно, скажете вы. Так, да не так. По разным причинам боссы всячески тянут с выделением ресурсов на уже утвержденный проект.

В принципе, на словах признается необходимость выделения денег, техники, людей, помещений. Но либо ресурсов пока нет, либо босс не решается отнять их у других своих подчиненных, зная, что будут неприятные сцены, либо просто не доходят руки.

Возникает дурная петля выпрашивания: вроде бы менеджер проекта поставил босса в известность о том, что ресурсы нужны и какие именно, и тот согласился. Планы и заявки утверждены.

Но в действительности ресурсов не дали: в бухгалтерии нет подписанной заявки на деньги, помещение не освободили, технику не купили, людей не отпускают из других проектов, кадровики дополнительных людей не ищут - не было приказа.

Менеджер снова приходит к боссу, а тот смотрит на него с надеждой - а может, так справишься? С деньгами в магазин и дурак сходит, а вот ты попробуй без денег!

Заново происходит торговля, препирательства. Менеджер снова выбивает уже однажды оговоренные ресурсы. Что-то наконец выделяют, чего-то опять не дают. И еще и еще раз, и так по кругу.

Заметим, что всё это время часы проекта тикают - ведь план утвержден и время уже пошло!

Иногда руководитель проекта уступает - устает, не может противостоять авторитету босса (а ведь в некоторых компаниях часами насидишься в приемной), нервничает из-за сроков - и начинает работать с усеченными ресурсами. Это чрезвычайно опасно, так как в итоге приводит к срыву сроков и наказанию невиновных.

Я могу здесь посоветовать только одно - постараться не принимать действительность такой, как она есть, и не работать с урезанными ресурсами.

Напротив, в случае затяжек с выделением ресурсов нужно стараться совершенно формально переносить сроки в плане: нет ресурсов - подождем, но запишем это в план. Обычно боссу в момент очередной торговли о ресурсах на это возразить нечего.

Такой формальный подход может не помочь с ресурсами (особенно если страшный секрет начальства состоит в том, что их на самом деле нет), но, по крайней мере, снимет с менеджера ответственность за чужие проблемы.

Трение покоя

В начале проекта всегда происходит торможение. Оно распространяется не только на выделение ресурсов начальством, но и на саму рабочую группу, и на помощь смежников.

Для неопытного руководителя это может стать неожиданной и непонятной проблемой - с большой вероятностью вначале все попытки запустить ход проекта и получить первые результаты будут оканчиваться ничем - несмотря на то, что все правильные действия сделаны: все люди проинструктированы, планы написаны, решение принято, первые собрания рабочей группы проведены.

А работа не движется. Почему-то все ждут действий остальных, никак не приступят к работе по своей части проекта, группа не общается между собой, все контакты нужно инициировать сверху.

Это - нормально.

Чтобы зажечь людей идеей проекта, нужны время и энергия. Здесь очень подходит аналогия с костром: костер нельзя зажечь из больших поленьев и моментально (разве что с помощью бензина) - нужно начинать с небольших веточек, и вначале огонь может постоянно гаснуть. А сильное пламя разгорается далеко не сразу.

Мне эта аналогия много раз приходила в голову, когда после очередного собрания рабочей группы и заинтересованного обсуждения проекта огонь в нем вроде бы

занимался, а потом я видел, что люди, как поленья, развалились в стороны и лежали холодные.

Снова складываешь костер, разводишь огонь - собираешь людей на очередную летучку, придумываешь им небольшие, понятные дела по проекту, сам пишешь и распространяешь установочные бумаги по отдельным частям проекта. И вот недели две спустя с удовольствием замечаешь, что вроде бы пошло - вдруг двое-трое сами пошли в курилку кое-что обсудить, потом в почте начали мелькать письма между разработчиками, наконец группа самопроизвольно, без подталкивания и без присутствия начальства собралась поговорить по техническим вопросам, дальше пошли снизу вверх требования ресурсов и планы, разработчики стали говорить "не мешай, отойди" - значит, наконец началось самостоятельное горение проекта.

Правило 9. В начале проекта всегда бывает естественное торможение. Чтобы его преодолеть, нужны терпение и настойчивость.

Нужно заметить, что о феномене торможения нужно помнить при составлении планов - почти гарантированно в начале нового проекта две-три недели уйдут на разогрев.

Конечно, в проектном деле существуют и аналогии бензина для разжигания костра - можно запустить важный проект быстро, если получить карт-бланш и собрать звездную команду на всё готовое (при отсутствии ограничений на ресурсы: технику, деньги, отвлечение людей из других проектов). Такое бывает, и в случае успеха такие проекты входят в легенды.

Впрочем, какой-то период торможения всё равно будет, только покороче.

Температура горения

Я глубоко убежден, что сделать проект с командой, работающей на полставки и *думающей* о проекте "на полставки" - нельзя. Здесь вопрос даже не в количестве рабочего времени в неделю, а в эмоциональной вовлеченности. Кто-то в проекте должен занять проектом свою голову и сердце на 100%. Лучше, чтобы это был руководитель проекта, но может быть и его заместитель, или главный разработчик.

Только в этом случае можно обеспечить необходимый энтузиазм, а также вовремя заметить нелады в проекте и справиться с неизбежными, но неожиданными проблемами.

Возвращаясь к аналогии с костром, можно сказать, что огонь в костре должен быть настоящий - температура не может быть ниже 451 градуса по Фаренгейту. Иначе горения не получится, будет только тление и дым.

Правило 10. Вполнакала проекты не делаются.

(Точной формулировкой этого правила я обязан Илье Ставискому, владельцу компании "Информатик".)

Доверие, оно же контроль

В ходе проекта также несколько раз приходится принимать решения. Это должно происходить в **контрольных точках**. Контрольные точки должны быть сразу внесены в планы проекта.

Контрольные точки служат для того, чтобы оценить, идет ли проект как запланировано, или имеются проблемы. Дальше требуется принять решение - продолжить работу по плану или вносить коррективы (в планы, в объем выделенных ресурсов, структуру руководства).

Чтобы руководитель проекта и его начальство имели возможность принять такое решение, их нужно обеспечить информацией - отчетами о выполнении. Также заранее должны быть определены **критерии** для принятия решения - как минимум, точно описаны ожидаемые в данной точке результаты, чтобы можно было легко определить, достигнуты они или нет.

Понятно, что контрольные точки должны располагаться на стыках важных этапов проекта. Как правило, сюда входят написание и приемка ТЗ, выпуск прототипа, выпуск бета-версии, окончание тестирования и техническая приемка, завершение подготовки к выводу на рынок или публикации.

Принятие решений в контрольных точках должно быть **явным** - в виде актов, электронных писем или устного признания начальством того факта, что проект идет по плану (или что обнаружены проблемы и срыв сроков, и будут сделаны такие-то коррективы).

К сожалению, очень многие менеджеры либо пропускают контрольные точки либо смотрят на их сдвиг сквозь пальцы. Естественно, в таком случае отставание от графика начинает незаметно накапливаться, и к сроку завершения проекта возникают неприятные сюрпризы.

Правило 11. Только коррекцией в контрольных точках можно удержать проект от срыва.

Корректное завершение проекта - передача ответственности

Проект обязательно должен получить корректное завершение.

Прежде всего это касается не конкретных действий (тестирование, документирование, приемка, консервация и архивирование сами собой разумеются), а вопросов ответственности и принятия решений.

Поскольку основной вопрос проекта - это не ресурсы или количество работы, а планы и ответственность, то и главный вопрос завершения проекта - новое надлежащее переключение ответственности.

Правило 12. По окончании проекта нужно правильно переключить ответственность.

Принятие решения о завершении проекта

Начальством должно быть в явной форме признано (решено), что планы выполнены и проект завершен. Если есть недоработки или предложения по необходимым

улучшениям, должно быть принято еще одно решение - продолжить проект для завершения доработок.

Передача проекта

Теперь завершенный проект должен быть официально передан от разработчиков в департамент продаж, департамент поддержки и им подобные. "Передан" означает официальную передачу ответственности за проект.

Новая рабочая группа

В этот момент может быть назначен новый руководитель (ответственный), должен быть пересмотрен состав рабочей группы - сформирована группа поддержки проекта, написан план поддержки. Фактически, снова произведена процедура запуска проекта - теперь уже *проекта поддержки*.

Размывание ответственности на стыке

Снова может показаться, что сказаны вещи совсем уж очевидные. Однако огромное количество проектов никогда не заканчивается таким правильным образом. Разработчиков зачастую никто не распускает, проект "висит" на них; официальной передачи другим департаментам не происходит - тем просто подбрасывают задачи сделать то одно, то другое по выводу на рынок чужого для них проекта.

В результате ответственность совсем размывается, выпуск проекта (вывод продукта на рынок, публикация сайта в Сети, запуск в эксплуатацию) топчется на месте, планов по выпуску не пишут.

Маркетоиды уверены, что проект еще "вылизывают" и ответственность на разработчиках, а разработчики - что его давно готовят к продаже, и ответственность с них снята.

Технический руководитель не знает (так как от него теперь это не зависит), может ли он уже перебросить людей на другие работы и каких именно, и делает это "по факту", по мере необходимости.

В подобных случаях довольно полезны внутренние акты приемки и твердое официальное оповещение боссов и смежных отделов о прекращении работ по проекту, за исключением таких-то доработок по списку. Они визуализируют основной факт - переключение ответственности.

Проекты никогда не заканчиваются

Давным-давно я слышал от одного монтажника-высотника, что на каждой большой стройке они оставляют по человеку (который разбивается). В разработке ПО ситуация чем-то схожая - руководитель проекта должен помнить о том, что если проект важный и развивающийся, кто-то из рабочей группы так и останется постоянно привязанным к нему - всегда будет требоваться поддержка и развитие.

Незаметная рутинная возня с уже выпущенным проектом будет постоянно отнимать время и силы. Обычно этот факт выпускается из виду и поддержка не

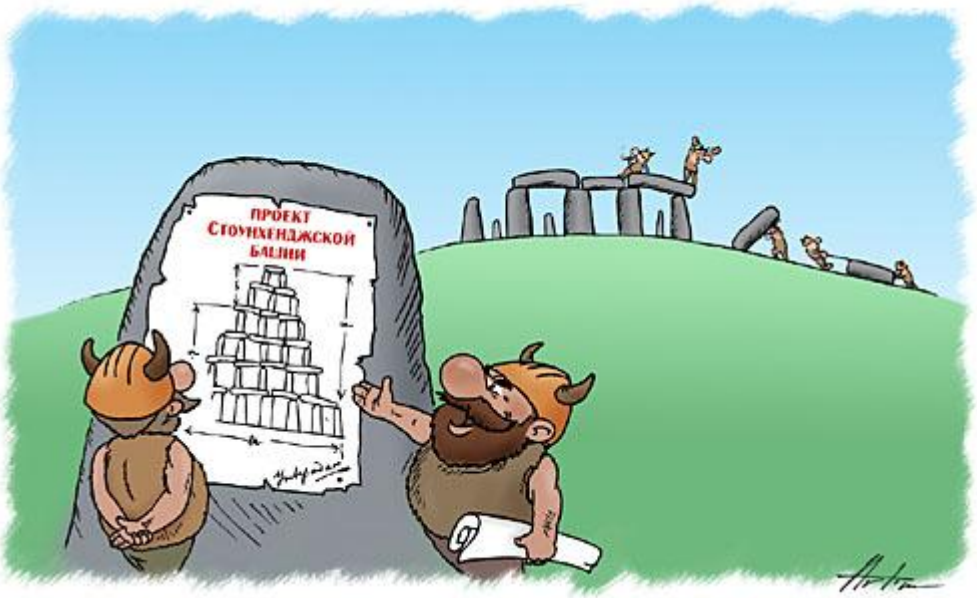
учитывается в планах. Если выпуск новых версий все-таки можно оформить, как новый проект, и получить под него ресурсы и сроки, то "визуализировать" для верхов поддержку не так просто.

Часто разработчиков заставляют вести поддержку выпущенных проектов, упирая на *чувство* вины и ответственности - "ну это ж твои ошибки, так исправляй". А между тем стоит выпустить три-четыре важных проекта, и все основные разработчики оказываются заняты этой незаметной и непрестижной деятельностью. Призов за нее не дают, а работа по исправлению ошибок - довольно скучная и крайне неблагодарная. Такая ситуация потенциально конфликтна, поскольку ведет к недовольству и перегрузке персонала и неожиданным срывам проектов.

Итак, получаем последнее правило в данной части:

Правило 13. После завершения проекта нужно запустить следующий "бессрочный" проект - проект по поддержке.

Проект по поддержке должен иметь все признаки проекта: ответственный руководитель, планы, ресурсы и т. п. Отличия срочных проектов от бессрочных - тема для отдельного разговора.



Артем Попов. © "Ашманов и Партнеры"

II. Неправильный проект и его лечение

Риск провалить проект если и зависит от квалификации руководителя, то не прямо. Точно так же рост мастерства управления автомобилем не всегда снижает риск аварии, а иногда даже повышает. Снижает этот риск не умение рулить, а осторожность и аккуратность водителя, но и она гарантий всё равно не дает - в конце концов есть еще окружающая обстановка и разные случайности. То же и с проектами. Поэтому менеджеру проекта нужно помнить следующее грустное правило:

Правило 14. Риск неудачи проекта есть всегда.

Если исключить проекты, рухнувшие из-за постановки абсолютно недостижимой цели, прекращения финансирования, банкротства компании или закрытые волевым решением руководства из каких-то высших маркетинговых соображений, то остальные риски возникают обычно из-за неправильного ведения проекта.

Негативные тенденции накапливаются и в определенный момент превращаются в болезнь. Как распознать их заранее?

Признаки неправильного проекта

Опыт проваленных проектов ничем не заменишь. Дело в том, что у неправильного хода проекта есть свои несомненные признаки. Если менеджер проекта или его босс уже имели опыт провальных проектов, они с легкостью узнают нехорошие симптомы.

Правило 15. Важно вовремя узнать "паттерн" неудачного проекта - симптомы надвигающейся болезни.

Как распознать заранее будущие проблемы проекта? Есть несколько типичных признаков.

Виртуальная реальность проекта

Очень часто проекта на самом деле не существует. Нет планов работ, практически нет руководителя, фиксации промежуточных состояний. Никто не оценивал денежные перспективы, объем рынка, трудоемкость. Однако ответственные лица изредка собираются на совещания, обсуждают классную идею, и компания числит проект в списке выполняемых. Программисты даже что-то там иногда программируют, деньги тратятся.

Относительно благополучный проект также может быть не целиком виртуальным, а частично, в некоторых своих кусочках. Чаще всего превращение благополучного проекта в проваленный - это и есть процесс постепенной "виртуализации" проекта.

Распознать такой виртуальный проект можно в первую очередь по сложившейся вокруг него мифологии.

Мифы и отсутствие информации

Мифы вокруг проекта - это набор общих высказываний и мнений о проекте, ложность которых постороннему ясна с первого взгляда.

Где руководитель? Например, может считаться, что у проекта есть руководитель, а на самом деле фактически его нет.

— Да есть как бы руководитель проекта, Петров. А, он не руководитель? А кто? Ну хорошо - как бы ответственный за проект, только он сейчас в командировке... Но его можно будет спросить, когда он приедет.

Когда он приезжает, выясняется, что за проект на самом деле отвечает кто-то другой, а Петров не в курсе. Точнее, перед поездкой вместо Петрова назначили другого, но тот пока всё еще занят в другом проекте.

Отсутствие ответов на самые простые вопросы. Никто не может ответить на вопросы, которые должны были бы быть прояснены в самом начале.

Например, считается, что по завершении проекта будет получена очень крутая штука, но никто не знает о том, что ровно это почти у всех есть:

— А на сайтах конкурентов что написано? Какие они основные функции своего продукта продвигают?
— Да, кстати, хорошая мысль! Ух ты! Сейчас скажу Сенькину, чтобы напряг там своих аналитиков - пусть посмотрят.

А где раньше был Сенькин и его аналитики? Да и сам босс тоже?

Коммерческая неясность. Зачем нужен проект и сколько на нем можно заработать - точно неизвестно. Когда выводим на рынок (запускаем, публикуем) - тоже.

Мутное состояние проекта. Диагноз подкрепляет неясность с состоянием проекта, когда никто толком ничего не знает. Боссу то и дело приходится вызывать то одного, то другого менеджера, чтобы разобраться в статусе проекта, все кивают на других, кто-то в командировке или болен, а кто-то вообще уволился.

Недостаток письменной информации. Отсутствие или негодность планов и других необходимых бумаг также очень подозрительны. Типичными признаками полуживого, полувиртуального проекта, вяло растущего на грядке, без присмотра, являются:

- План без фамилий исполнителей, с устаревшими и не обновленными датами, с пропусками таких ключевых этапов, как тестирование и выпуск, обрывающийся на бета-версии.
- Десятистраничное "Техническое задание" годичной давности, содержащее перемешку русских и английских кусков, незаметно переходящее в описание функций на языке C++, запинающееся к концу и обрывающееся на полуслове.
- "Атомарные" недельные отчеты из полутора десятков пунктов в стиле "крутили гайку номер 8 ключом на 13 два рабочих дня", которые абсолютно ничего не говорят ни о месте и назначении гайки, ни о цели кручения, ни о состоянии проекта в целом.
- Устаревший на два-три месяца план-график работ над столом руководителя.

Бесконечные совещания и "мозговые штурмы"

Особенно показательны многолюдные совещания, к которым никто не готовится и на которых не принимается решений по проекту. По неизвестной причине босс считает, что для исправления ситуации в проекте нужно собрать вече из пятнадцати сотрудников, часть из которых вообще не в курсе проекта.

"Чтобы разобраться - нужно собраться." То есть вызвать всех ответственных и устроить разбор полетов. Приглашается масса народу. Как ни странно, руководитель компании к такому собранию обычно не готовится интеллектуально - не пишет повестку собраний, не отбирает немногих ответственных лиц для участия, не задает очередность выступлений, не требует заранее отчетов, не ходит заранее "в народ", чтобы получить независимые мнения.

Однако, к собранию он обычно подготавливается эмоционально - накапливает запас начальственной воли и гнева.

Не готовятся обычно и подчиненные, уставшие от проекта и заваленные другими делами. Только самые хитрые и хладнокровные из них заготавливают список причин и оправданий - "отмазок".

На совещании разбор полетов довольно быстро переходит в кашу, разбивается на несколько независимых дискуссий, слышатся угрозы и упреки начальства, поток технических разъяснений, исторических экскурсов "как всё было". Естественно, рефреном со всех концов стола звучит "ресурсов же не хватало, мы же говорили!". Начальник надувается (ведь это претензия к нему, он же якобы не дал ресурсов).

В середине заседания возникает вдруг конструктивное течение - возникает вопрос "ну хорошо, что делаем дальше?", высказываются конструктивные предложения. Потом опять всё само собой тонет в пучине обсуждения вопроса "кто виноват и как же так вышло?".

Заканчивается собрание ничем уже поздним вечером - все расходятся уставшие, возбужденные и обиженные, руководитель вдогонку устало выкрикивает последние угрозы разобраться и наказать, и дает нечаянно задержавшимся в переговорной сотрудникам приказы дать бумаги с анализом ситуации и предложениями.

Как ни удивительно, иногда после собрания вдруг спонтанно возникает второе собрание в узком кругу задержавшихся, гораздо более конструктивное и результативное.

В результате собрания у босса все-таки появляется б*о*льшая ясность, однако конструктивных решений не возникает.

Постоянный перенос сроков

Ну, здесь комментарии не требуются. Это всегда плохой признак. Однако указывать он может на разные проблемы, от чисто технических и преодолимых препятствий до полного развала управления в проекте. Тут нужно разбираться.

Очень часто перенос сроков является признаком очень опасной ситуации - петли загруженности.

Дурная петля загруженности

В разработке ПО очень часто воспроизводится известный случай с пилой:

— Почему вы пилите тупой пилой, так ведь очень долго и трудно? - Да некогда точить, пилить надо!

Разработчик, да и руководитель проекта очень часто попадают в эту ловушку, когда кажется, что окончание проекта - за ближайшим поворотом. Поэтому разбираться, планировать и оценивать состояние проекта некогда. И они легко объясняют это начальству.

Обычно эта дурная петля возникает в условиях "небольшого срыва" сроков. Действительно, некогда и незачем писать оценки и планы, когда продукт выйдет вот-вот... Некогда оценить сроки и **размер** срыва.

Слово "небольшого" написано в кавычках не зря - я многократно видел ситуации, когда после принудительной остановки гонки высшим менеджером и трезвой оценки в тишине состояния проекта вдруг оказывалось, что срок на самом деле сорван не на дни, а на месяцы, и во многих частях прокта еще конь не валялся.

А иногда срок оказывается сорванным **навсегда**. Либо проект просто невыполним, либо инвесторы/хозяева не готовы платить *такую* цену или ждать *столько*.

Причем для получения этой ужасной правды руководителю проекта, как правило, достаточно дня спокойного размышления и написания электронного письма на страницу - то есть он мог бы это понять и раньше.

Более того, хоть это и выглядит дико в глазах вышестоящего руководителя, но обычно после получения подобной оценки состояния проекта, выданной наконец под давлением, менеджер проекта начинает утверждать, что полная неготовность проекта, огромный объем недоделок и необходимость потратить еще месяцы - совершенно очевидны, закономерны и давно известны. На вопрос "а что ж ты раньше-то молчал!" он обычно отвечает - "да я же говорил, а вы мне руки выкручивали". Босс-то помнит, что на самом деле менеджер проекта ничего не говорил, но доказать это уже невозможно.

Дело тут в том, что разработчики давно уже в глубине души сознавали, что всё плохо, но не признавались в этом даже сами себе, при том, что сумасшедший объем стоящих в очереди проблем позволял забыться "в борьбе за урожай".

Распознать петлю загруженности и скрытый срыв легко - после первого-второго переноса сроков или при первых признаках аврала нужно запросить от руководителя состояние проекта и уточненный план.

Разорвать петлю гораздо труднее - для этого требуется принятие решения об изменении хода проекта (добавлении ресурсов, усечении требований, переносе сроков), а это всегда нелегко.

Порочная логика разработки

Очень популярен перенос сроков и списывания долгов по причине спонтанного повышения требований к проекту. Наращивание функциональности очень легко находит сторонников и среди разработчиков, и среди менеджмента, потому что ведь так логично и увлекательно добавлять новые возможности.

— Смотрите, как логично всё получается - сначала простая версия, потом тяжелая, потом смежные продукты, потом новое ядро, потом продукты для всех сегментов, потом глобальные версии, потом локальные продукты для Венеры и Марса, потом...

Очень часто такая логика используется и для переноса срока выпуска проекта:

— Если мы еще добавим такую фичу, будет очень круто, ни у кого нет, все клиенты просят. А ведь нужен всего лишний месяц.
- Ну давай перенесем выпуск на май, только опиши подробнее новую функциональность.

И часто этот ход используется для переноса срока уже проваленного проекта.

Жизнь в виртуальной реальности

Наиболее запущенная стадия неправильного развития проекта - полный переход его существования в виртуальную реальность. Вот основные признаки ее.

Отсутствие обратной связи

Основной принцип нормальной работы проекта или компании - наличие обратной связи с окружающим миром. Отсутствие такой связи с неизбежностью приводит к возникновению виртуальной реальности.

Отсутствие обратной связи обычно проявляется для менеджера в том, что за провалы практически не наказывают, только журят, причем можно легко отбрехаться: написал бумагу с оправданиями или невнятный "атомарный" отчет о том, что разработаны такие-то мелкие модули с непроизносимыми названиями, сослался на трудности, нечувствительно перенес сроки, и начальство опять на время успокоилось.

Резонанс распущенности

Естественно, вместо полезной петли обратной связи, вознаграждающей за результаты и наказывающей за провалы и ошибки, возникает дурная, виртуальная петля обратной связи, твердо подкрепляющая желание жить "на халяву".

Некоторые менеджеры быстро выучивают способы получать реальные блага за виртуальные ценности. Нужны вам отчеты - будут вам отчеты. Всё легче, чем за программистами гоняться. Маховик виртуальности начинает раскручиваться в уме менеджера.

Я наблюдал случаи полного отрыва от реальной жизни, когда менеджер, только что проваливший важный проект или упустивший большой контракт, приходил к начальству с требованием повышения оклада (или даже получения опциона). Как это может быть? Да просто давно собирался потребовать, и не видит связи оклада с реальной жизнью.

При этом начальство не находило в себе сил выгнать его из кабинета, а начинало долго и мучительно препираться, еще и чувствуя неловкость.

Извне это выглядело просто абсурдно, и в какой-то момент я вспомнил, что именно это мне напоминает - компьютерную игру. Прыгнул не туда или застрелили тебя - вот тебе, пожалуйста - новая "жизнь". А если даже "жизни" иссякли и игра закончилась, то достаточно просто запустить ее заново.

Много жизней иметь вредно

Практически каждый из нас может вспомнить случаи, когда некий менеджер шел от одного провала к другому, круто поднимаясь по карьерной лестнице, так что каждый следующий проваленный проект был крупнее предыдущего.

Если бы в компании обеспечивалась обратная связь карьерного продвижения с результатами, путь этого вундеркинда по уровням игры был бы прерван в самом начале. Я уверен, что ему самому это было бы только полезно.

Торговля словами

В виртуальной реальности слова заменяют почти всё. Их можно обменять на премию, перенос сроков и прощение за провал, внеочередной отпуск в самое горячее время. Нужно только знать пароли к разным уровням игры.

Пустопорожние отчеты, дружеское просиживание в кабинетах начальства, питье пива с влиятельными людьми компании, отписки по электронной почте - всё это позволяет не заниматься основным делом и нивелировать результаты провалов. Естественно, торговля словами довольно скоро перетекает в паразитную социальную активность - интриги и сплетни, подсиживание коллег. Подробнее об этом роде деятельности см. книгу Скотта Адамса "Совершенно секретное руководство по менеджменту".

Награждение непричастных

Применение компанией премий и бонусов "по площадям" также усиливает сдвиг в сторону виртуальности.

Если квартальную премию платят всегда, то никто не ценит ее и она никого не стимулирует. Получается, что настоящими деньгами платят за виртуальные заслуги

В виртуальности бывают дикие случаи, когда увольняемому за развал работы сотруднику выплачивают очередную премию вместе с выходным пособием!

Возникновение "карманов" безделья

Часто в компании возникают своего рода "карманы" - тихие уголки, где можно месяцами сидеть, ничего не делая, где нет ни работы, ни ответственности.

Обычно начальство помнит уголком сознания, что в этом кармане что-то такое должно было быть сделано, и надо бы проверить. Однако, ценность предписанной карману деятельности невысока, бюджет небольшой, и разобраться руки никак не доходят. Да, есть ощущение легкого неблагополучия в том углу, но оно особенно никого не "напрягает". И без того настоящих проблем хватает.

Очень распространенный пример - отдел веб-дизайна в торговой фирме. Сайт вроде есть, есть программисты и даже маркетинговый начальник. Сайтом все, в общем, недовольны: он давно устарел, медленно загружается, никак не помогает бизнесу, от вебмастера не допросишься обновить простейший прайс-лист. Но компания не видит большого смысла в Интернете вообще, и руки навести порядок на сайте никак не дойдут.

А часто вообще никто не может сказать, хорошо ли работают в таком кармане, потому что посмотреть на результаты "глазами" вообще нельзя, а планов давно уже нет - то есть нет критериев для оценки работы. Например, как оценить качество техподдержки, особенно если нет охоты связываться?

Метастазы виртуальности

Области виртуальной реальности имеют свойство расширяться, захватывая всё большую часть проекта или всей компании. Действительно, думают те, кто наблюдает жизнь в виртуальной реальности своих соседей по проекту или по общему залу, - они там

не работают, срывают сроки, вообще вытворяют бог знает что, и ничего им не делается. Ну, а нам что - больше всех надо?

К сожалению, к легкой жизни в виртуальности быстро привыкают. Поэтому зачастую разогнать туман виртуальности можно только отрезвляющими суровыми мерами - увольнениями, лишениями премий, выговорами.

Однако, здесь важно не поддаться соблазнам **виртуального** же наведения порядка, которые стоят на пути любого начальника, не привыкшего долго обдумывать свои законодательные инициативы.

Соблазны наведения порядка

Как известно еще со времен Брежнева и Андропова, основными пятью признаками наведения порядка являются: шумиха, неразбериха, поиски виновных, наказание невиновных и награждение непричастных.

Когда даже самому высокому начальству становится ясно, что в проекте что-то идет не так (скажем, срок трехмесячного проекта сорван месяцев на шесть, или деньги на проект совсем кончились, а продукта нет), начинается наведение порядка со всеми описанными основными признаками.

Правило 16. Когда в проекте что-то идет не так, начальство предпринимает титанические бесполезные усилия по наведению порядка.

Вот основные негодные способы наведения порядка, с которыми сталкивался практически всякий, кто работал в неудачных проектах или несчастливых компаниях.

Контроль посещаемости

Первое, что приходит в голову боссу при личном обдумывании причин торможения или провала - что работники просто бездельничают.

— Менеджер проекта *сам программист* (вариант - *сам не программист*), вот и распустил их. Ясно, что нужно их просто тщательнее контролировать. Я сам этим займусь.

Начинаются мероприятия по контролю посещаемости. Охранник на входе получает приказ всех записывать, сисадминам чрез голову технического директора дают команду подобрать и установить систему контроля рабочих мест. Появляется предписанный шаблон объяснительной записки, босс или кадровик грозно встречает всех поутру при входе, проводится промывание мозгов опоздавшим и т. п.

Персонал ропщет в столовой, самые отчаянные угрожают в курилке увольнением. Ползут слухи об увольнениях. Программисты начинают демонстративно уходить домой в шесть вечера. Научиться приходить в десять утра у многих так и не получается.

Волна контроля посещаемости иссякает примерно через одну-две недели, максимум месяц.

Контроль количества работы

Далее светлая мысль начальства достигает самой сути: нужно контролировать даже не рабочее время, а количество работы! Возникает смелый план заставить программистов вечером *каждого* дня (вариант - по *пятницам*) записывать, что сделано, и тут же "одной кнопкой" отправлять запись в некую общую систему.

Обычно, к счастью, этот *ежедневный* вариант контроля не удается даже внедрить. Сопротивление разумной части коллектива оказывается слишком велико.

Пятничные отчеты начинают писаться (руководитель проекта упросил всех поддаться), но их перестают читать наверху через пару недель, а писать внизу - через месяц-полтора.

Правила внутреннего распорядка

Административное рвение обязательно приводит к появлению громадных и противоречивых документов, регламентирующих поведение сотрудников. (Обычно это творчество кадрового отдела. Поразительно, что отдел кадров обычно существует в абсолютной изоляции от компании, просто в башне из слоновой кости.)

Документы рассылаются по электронной почте с требованием расписаться и на полдня выключают персонал из трудового процесса. Почему-то в конце концов их никто так и не подписывает и инициатива сама собой глохнет.

Планы и отчетность

Планомания быстро достигает своего апогея. Принуждаемые сверху, менеджеры проектов всё время пишут отчеты и планы. К сожалению, эти планы обычно - атомарные, как описано выше, и фабрикуются путем копирования и вставки из прошлого отчета. Эти отчеты никто не читает, потому что человеку это невозможно. Но на какое-то время сам факт регулярного писания отчетов успокаивает начальство.

Премии и штрафы

Естественно, наводя порядок, нельзя пройти мимо материального стимулирования. В первую очередь босс держит в уме штрафы, но готов разговаривать и о премиях. Заслуженных, конечно.

Начальство подробно читает штатное расписание с карандашом в руке и детально выясняет, кто чем занимается в проекте. Продумываются варианты **аттестации** сотрудников и система доносительства их друг на друга. Технический прогресс проникает и в эту сферу - всё чаще придумываются варианты анкет и оценок соратников на базе интранета.

Начинают появляться варианты нового расчета зарплаты: например, предлагается разбить зарплату на три части - базовый минимум, потом довесок за исполнение проекта в срок, потом - премия за досрочное исполнение. (В первой части я уже объяснял, почему штрафовать программистов абсолютно бессмысленно.) Еще одна очень популярная бесперспективная идея - начать платить программистам процент с продаж вместо части

оклада или вообще разрешить им чем-то там торговать, привлекать заказы, а менеджеру отдать проект на хозрасчет.

Опять ползут слухи об увольнениях. Кое-кого даже увольняют или понижают в зарплате. И опять волна наведения порядка спадает через некоторое время - почему-то период релаксации всех нововведений составляет 1-2 месяца.

Комсомольские накрутки и политинформации

Все описанные выше набеги на персонал требуют мощного идеологического сопровождения. Для этого собираются собрания для повышения боевого духа.

Всё это в деталях описано в великолепных комиксах Скотта Адамса про бедного разработчика Дилберта и его глупого босса (http://www.dilbert.com), так что опустим подробности.

Обычно в коллективах разработчиков такие собрания не достигают своей цели, разработчики начинают пересмеиваться, задавать руководителям противные технические вопросы (а вот почему бумаги в принтере нет, а почему материнскую плату в сервер целую неделю невозможно купить и поменять и т. п.) и неудобные политические вопросы (про зарплаты), а потом на полную катушку оттягиваются на счет начальства в курилке.

Закручивание гаек не работает

К сожалению, все эти простые домашние средства, якобы повышающие потенцию коллектива - всякий раз плод не более чем двадцатиминутных размышлений босса. От этих знахарских примочек проект, естественно, не выздоравливает, хотя напуганный персонал работает еще больше, да и время идет своим чередом и что-то там мигающее на экране уже можно показать боссу.

На самом деле, конечно, проект действительно тормозится, а персонал опаздывает на работу и играет в компьютерные игры. Но вовсе не потому, что его не контролируют, а потому, что им плохо управляют.

Правило 17. Бессмысленно муштровать исполнителей, нужно муштровать менеджеров.

Это правило легко доказать следующим рассуждением:

а) Плохой исполнитель и хороший менеджер. Если в проекте оказываются вместе внятный и ответственный руководитель и неквалифицированный или ленивый исполнитель, первый моментально постарается уволить или удалить второго - чтобы не нести за него ответственности. И это правильно.

Поэтому ситуация с хорошим менеджером и плохим исполнителем - неустойчива и в жизни встречается редко, а именно в случаях кумовства, отсутствия полномочий самому выбирать исполнителей, застревания в проекте подружек шефа, которых не уволишь, и так далее.

б) Хорошие исполнители и плохой менеджер. А вот обратная ситуация - плохой менеджер с хорошими исполнителями - может сохраняться достаточно долго.

Исполнители-то работают, дают какие-то результаты. А менеджер в основном ходит к боссу и "кормит" его разными словосочетаниями.

Более того, в такой ситуации менеджер часто втихую (в кабинете босса) поддерживает проекты начальства по исправлению нравов (ведь это снимает с него ответственность, перенося ее на "плохих" исполнителей).

(Я, правда, знаю несколько случаев бунта исполнителей, после которых негодного менеджера снимали с проекта. Особенностью этих случаев было то, что исполнители не просили прибавок зарплаты или послаблений, а болели за ход проекта, что и нашло понимание у начальства.)

Таким образом, "лечить" исполнителей либо через голову среднего менеджера, либо при его поддержке (что еще более отвратительно) в большинстве случаев теоретически неправильно и практически вредно.

Правило 18. Наведение порядка в проекте всегда должно начинаться с головы.

То есть - касаться *существенных* составляющих проекта, то есть идеи и обоснования проекта, личности и квалификации руководителя, состава и согласованности рабочей группы, планов, контрольных точек.

Как вылечить неудачный проект

Дать рецепт для дистанционного лечения невозможно, поскольку каждый больной уникален (по слухам, слово "галиматья" происходит от фамилии французского врача XIX века, лечившего по переписке). Можно дать только общие советы на основе прошлого опыта.

Основной принцип лечения - распределить ответственность за проект заново и обеспечить обратную связь с результатами.

Вот какие основные этапы с необходимостью возникают при лечении проекта:

Разбор полетов и инвентаризация

Разбор ситуации нужно проводить в узком кругу, начиная с *индивидуальных* бесед с менеджером проекта и основными разработчиками. Каждый раз такая беседа должна заканчиваться просьбой написать документ о состоянии проекта или обсуждавшейся его части.

Признание наличия проблем

Рабочая группа должна *явно* признать наличие проблем и срыва в ходе проекта. Начальство также должно это сделать. Если же одна из сторон будет считать, что всё нормально или проблемы невелики, исправить ничего не получится. Основной принцип ответственности - обратная связь с окружающим миром. Если проект сорван, обратная связь должна быть ощущаема всеми участниками.

Признание ошибок и вины

Поскольку проект требует нового распределения ответственности, вина за провал должна быть явно признана. Это не вопрос наказания виновных, а вопрос их готовности к переменам.

Если руководитель проекта или часть рабочей группы будет исповедовать принцип "виноват не я, а все прочие", то внутри проекта в будущем может возникнуть тайная или явная оппозиция, возлагающая вину на других и продолжающая делать те же ошибки.

Очень полезно, чтобы начальник, занимающийся реструктуризацией проекта, также явно признал часть вины за начальством (за то, что не давали ресурсов, не следили, не помогали).

Создание новой команды

Поскольку уже *признано*, что старая команда справлялась плохо, нужно создать новую команду. Это могут быть ровно те же люди, но расставленные как-то по-другому, с другими полномочиями и ответственностью.

Главное в новой команде - новые взаимоотношения, поскольку старые явно не сработали. Можно заменить начальника и/или исполнителей, а можно и дать им новый шанс, но только если они приведут какие-то доказательства, что это имеет смысл. Под доказательствами я понимаю убедительные аргументы в виде признания вины (или хотя бы совершенных ошибок), предложения по улучшению (например, в виде новых внятных планов).

Наиболее типовые фразы, которые менеджеру придется услышать, заменяя того или иного члена команды: "да он один сейчас владеет ключевой информацией по проекту", "он с самого начала в проекте", "да это я всё придумал", "давай дадим ему еще раз попробовать", "просто он раньше был занят", "раньше не хватало ресурсов, а теперь он справится"... Как правило, такое говорится из боязни перемен или расположения к заменяемому.

Лечение больных самолюбий

Естественно, признание провала проекта, признание ошибок большинству разработчиков легко не дается. Тому, кто исправляет ситуацию в проекте, придется пережить мучительные дни, состоящие из разговоров "за жизнь", выслушивания претензий и угроз уволиться, тушения истерик, и так далее. Вечером, возможно, придется пить пиво с разработчиками и бесконечно обсуждать ситуацию.

Здесь придется заниматься психотерапией. С одним существенным отличием: для вас хорошее самочувствие пациентов **не** является главной целью. Оно желательно, но ваша цель - завершение проекта.

В конце концов, срыв проекта - событие реального мира, следовательно, и сопутствующие эмоции неизбежно должны быть настоящими, чтобы обеспечить хоть какую-то обратную связь.

Кто старое помянет...

После оргвыводов, раздачи тумаков, штрафов или порицаний, перемещения менеджера и исполнителей, новых назначений нужно как можно быстрее объявить новой/старой команде, что вопрос с провалом проекта - закрыт. Старое забыто, пора работать.

Принцип списания всех убытков и долгов очень важен для того, чтобы запустить новый проект. Если об этом не объявить сразу, старые обиды и амбиции заразят новый проект еще в утробе.

Если же добиться лояльности к переменам не удается (например, старый менеджер проекта, несмотря на уговоры, не может смириться со своим новым положением и постоянно борется с вновь назначенным менеджером), то всех возмутителей спокойствия нужно обязательно переводить в другие проекты. Итак, имеем последнее правило:

Правило 19. После разбора полетов нужно списать все долги и запустить проект заново.

Перезапуск проекта

Здесь всё в целом просто для того, кто уже научился правильно запускать проекты - нужно в полном объеме повторить процедуру запуска нового проекта. Нужны новые решения о запуске, новые обоснования, новые планы, новые сроки.

Вместо заключения: о чем не сказано в статье

О материальном стимулировании

Это слишком большая тема, чтобы обсуждать ее походя. Кроме того, я считаю, что основное в управлении - правильное распределение ответственности и принятие решений. Стимулирование может только несколько украсить ситуацию.

Впрочем, могу еще заметить, что выплата разработчикам авторских процентов или проектных премий (за результат) еще никому не мешали, а вот штрафы - решительно вредны.

О специальных средствах управления проектами

Если в офшорном программировании и системной интеграции использование систем управления проектами (от MS Project до Rational Unified Process) может быть просто необходимым условием получения заказа, то в малых и коротких проектах они, на мой взгляд, необязательны, хотя и могут быть полезны. Подробное обоснование этой точки зрения - тема для отдельной статьи.

Читателю может оказаться полезной уже упоминавшаяся статья Джоэла Спольски "Painless Software Schedules", подробно рассказывающая про эффективное использование Microsoft Excel для планирования разработки ПО и неудобство использования для тех же целей пакета для управления проектами Microsoft Project.

О стандартах качества

Применять ли стандарты качества наподобие ISO или CMM? Коротко выскажу свое мнение - тоже без развернутой аргументации. Опять-таки, для фирм, разрабатывающих ПО или информационные системы по заказу, а также для отделов разработки больших частных или государственных корпораций использование систем качества может быть не только необходимо, но и просто предписано правилами.

Однако в коротких проектах накладные расходы на поддержание системы качества могут съедать до 15-20% бюджетных средств и времени. Для короткого проекта это недопустимо.

Мне кажется, универсальные системы качества наподобие CMM и ISO являются в основном страховкой для заказчика ПО, то есть рассчитаны в основном на заказные проекты и возможность их поддержки и развития после сдачи подрядчиком.

А вот для фирм, разрабатывающих собственные программные продукты, системы качества могут оказаться вообще смертельным лекарством, потому что могут омертвить интимный процесс угадывания нужд потребителей и быстрого вывода продуктов на рынок (и подменить его работой на отдел качества). Какой смысл испытывать гордость за правильное оформление работ и возможность повторного использования кода, если продукт опоздал выйти на рынок?

Неспроста наиболее крупные и успешные разработчики ПО используют собственные, приспособленные к внутренним задачам системы качества.

Поэтому **лично я считаю**, что в коротких и небольших проектах, а также при разработке собственных программных продуктов применение общих стандартов качества не оправданно.

III. Приложение

Словарь ненормативной лексики руководителя

Здесь собраны типовые высказывания руководителей и/или хозяев компании, которые при настойчивом их повторении приводят к трудностям в управлении проектом и персоналом.

Игорь Ашманов

Чтобы проект удался, надо знать и выполнять ряд простых правил. Однако многие менеджеры надеются, что пронесет и так. Правильные проекты, и что для этого нужно. Неправильные проекты, и как их лечить.

Игорь Ашманов

"Правила Ашманова" - часть вторая: управление проектами

Организовать проект - сейчас сделаем!

- За проект отвечают Пронькин, Тюлькин и Васькин.

Комментарий: так не бывает. Шофер у автомобиля должен быть один. Три или пять "ответственных" обязательно развалят проект.

- Что-то вы много работы насчитали. Выдумываете сложности. Нужно просто напрячься, недельку посидеть ночами и всё сделать! С программистами всегда так. Ты поговори там с ребятами.

Комментарий: так говорит менеджер, не понимающий сути разработки ПО (которая "не резиновая") и не контролирующий ситуацию в проекте (который скорее всего уже сорван).

Кстати, это точная цитата "из жизни".

- Загруженность Петрова по нашему проекту 57%, а Сидорова - 15%.

Комментарий: обычно подобные дробные значения загруженности используют в системах управления проектами (или просто в Microsoft Excel) для "точного подсчета" ресурсов.

Я как-то не очень верю в подобные расчеты. Жизнь устроена так, что чаще всего 15% на самом деле равняются 0%, а 57% превращаются в 100% или 120%. Лучше всего, когда под проект выделена полностью занятая в нем рабочая группа, которая изредка дергает другие обслуживающие отделы.

- Надо, чтобы каждый перед уходом отмечал, сколько он сделал по проекту. Тогда мы будем точно знать состояние дел.

Комментарий: проверено - не работает. Как минимум, не работает как средство исправления ситуации в неблагополучном проекте. И как средство контроля помогает весьма умеренно. Ситуацию в проекте должен знать и сообщать его руководитель, а не автоматическая система контроля.

- Программистов нужно штрафовать за срыв сроков, чтобы чувствовали ответственность.

Комментарий: не поможет, они просто разбегутся. Проблемы со сроками возникают от плохого управления, так что штрафовать нужно менеджеров.

Кроме того, всегда соблюдать все сроки в разработке ПО практически нельзя. Вместо этого нужно иметь разумную систему диагностирования проблем на ранних этапах и процедуру переноса сроков.

- Почему вы всё время требуете новую технику? Неужели нельзя ускорить программу? Между прочим, вас наняли как раз потому, что вы опытный специалист по программированию и должны сэкономить на технике.

Комментарий: звучит логично, если бы не тот факт, что босс постарается денег на технику не дать в любом случае. Удастся ускорить или нет - техники не будет. Типичный случай попытки не кормить лошадь - а вдруг пообвыкнет? Тоже, между прочим, цитата из реальной жизни.

Что-то вялый у нас руководитель проекта...

- Я разослал всем проект/план/ТЗ, но никто не ответил. Что я, за ними гоняться должен? Они же взрослые люди. Какие ко мне претензии?

Комментарий: в общем, должен гоняться, конечно. Иначе проект закиснет. От руководителя требуется решать любые проблемы проекта, в том числе перебарывать отсутствие интереса к проекту у смежников и рабочей группы.

- Я считаю, каждый должен заниматься своим делом, в своей зоне ответственности. Я не должен делать работу за них.

Комментарий: это типичная фраза из арсенала так называемой "спихотехники". Внешне звучит совершенно правильно, прямо из западного курса MBA, но по сути означает снятие с себя ответственности за ход проекта в целом и отказ от усилий по коммуникации с другими участниками.

Точный перевод на русский язык: "я прокукарекал, а там хоть и не рассветай".

- Состояние проекта - ну чего там писать! Я сейчас всё на словах скажу. Работа ведется, всё нормально.

Комментарий: постоянное откладывание менеджером проекта рапорта о состоянии дел - самый грозный признак приближающегося или уже наступившего срыва проекта.

- Да он нормальный руководитель, только вот план никак не напишет.

Комментарий: основная физиологическая функция менеджера - это планирование своей и чужой работы. Если менеджер никак не напишет план - это не менеджер. Либо он уклоняется от взятия на себя ответственности, либо вообще не способен планировать работу.

- Да, с проектами и руководством людьми у него не очень. Но вообще польза от него бывает, вот на той неделе написал пару бумаг/провел пару встреч на неплохом уровне.

Комментарий: здесь явно обсуждается виртуальный сотрудник. Возможно, действительно как-то этого менеджера использовать можно (на побегушках), только его позиция в компании и зарплата предполагают гораздо более внушительные результаты. А порученец на позиции среднего менеджера - не нужен.

Кадры решают всё?

- Это ключевой человек. Если он уйдет, всё рухнет.

Комментарий: такое бывает, но к счастью, довольно редко. Часто после ухода "ключевого" менеджера всё "устаканивается" на удивление быстро - либо потому, что "ключевой человек" уже давно имитировал деятельность, ушел неспроста и рушиться просто нечему, либо потому что его незаменимость просто была преувеличена. Конечно, главный вопрос - как распознать, какой случай имеет место сейчас...

Период вступления в управление проектом нового менеджера подобен небольшому землетрясению в проекте, но главное - чтобы новый руководитель был действительно "правильный".

- У нас работают такие творческие люди, что их не загонишь в рамки планов и прочей бюрократии.

Комментарий: наиболее типовая мантра в программистских компаниях. Так обычно успокаивает себя руководитель, не сумевший навести порядок и наладить промышленное производство ПО.

- Нам нужны люди, но такие своеобразные/творческие, что кадровые агентства здесь не помогут.

Комментарий: напротив, опыт показывает, что кадровые агентства как раз помогают - быстро находят очень квалифицированных людей. Правда, это дорого. В противном же случае "творческих людей" берут по знакомству и потом никто не решается твердо спросить их про плоды их творчества. К тому же круг знакомых как источник кадров быстро иссякает.

- Да он хороший, командный парень. Ну, работу, конечно провалил, да я его вообще за это пинаю каждый день. В пятницу мы с ним в боулинге выпьем пива и серьезно поговорим.

Комментарий: боюсь, менеджер тут путает комфортные личные отношения и "командность" подчиненного с практической пользой для компании. Явно имеются признаки виртуальной реальности, когда питье пива с шефом заменяет реальные результаты. Можно прогнозировать, что очередной дружеский разговор ничему не поможет.

Отчего всё не так? Продолжать или бросить?

- В принципе, к ним есть какие-то нарекания, но вообще что-то там они делают в своем проекте. Надо как-нибудь разобраться, как там дела.

Комментарий: очень похоже на теплый "карман" виртуального существования. Боюсь, сил разобраться у босса не будет еще долго.

- Давайте соберем всех и наконец всё решим по проекту.

Комментарий: собрания типа новгородского вече - типичный признак "виртуального", несуществующего проекта. Собираются на них без подготовки, обсуждение идет зигзагами. Большая часть собравшихся не включается в обсуждение или не берет на себя ответственность. Кончаются они чувством раздражения и еще большей неясностью. Эффективные решения в группе 5-10 человек, мало заинтересованных проектом, - не принимаются.

- Ну послушайте, мы уже потратили на проект кучу денег, давайте уж доведем его до конца...

Комментарий: это плохая мотивировка. Доведение до конца ненужного или неудачного проекта обычно съедает еще больше денег, сопровождается скандалами и увольнениями.

- Как же закрыть проект, ведь обещаний в прессе, начальству, партнерам мы уже дали вон сколько... Я не готов принять такое решение.

Комментарий: подобная нерешительность руководителя - путь к долгой и мучительной агонии проекта и еще большей потере лица в итоге. Впрочем, если тянуть достаточно долго, даже партнеры и журналисты могут забыть об обещанном выпуске проекта.

- Проект уже навяз на зубах, мы всем обещали, пресса ждет, давайте выпускать как есть.

Комментарий: выглядит как волевое решение. Руководитель доволен, что наконец поступил по-мужски (вариант: как настоящий бизнесмен). Однако это имитация решения: если проект в плохом состоянии, *выпускать его категорически нельзя* - выпуск дрянного продукта будет виден всем, будет испорчена репутация многих участников проекта и самой компании, на следующий же день среди высшего менеджмента начнутся разборки и т. п.

А вот закрыть проект или перезапустить его, честно запланировав нормальный выпуск (и признав срыв сроков), - можно. И это по-мужски.

- Конечно, можно выпустить проект в срок, но с ошибками. А я считаю, что нужно выпускать, когда получится, но зато качественно.

Комментарий: характерное предложение руководителя проекта при имеющемся уже срыве сроков, фактически - "отмазка". Безусловно, качественный продукт лучше. Но только в том случае, когда срок "качественного" выпуска также точно зафиксирован. В противном случае эта нечаянная любовь к качеству - просто прикрытие срыва проекта и переноса сроков.

- А куда мы денем руководителя проекта - мы ведь брали его именно под этот проект?

Комментарий: но ведь вы не брались обеспечивать его семью до конца жизни? Сильный человек найдет свое место в компании или за ее пределами; слабый ей в принципе не нужен. В любом случае это не аргумент для продления агонии проекта.

Игорь Ашманов

Словарь ненормативной лексики программиста

Игорь Ашманов

Артем Попов. © "Ашманов и Партнеры"

Этот небольшой словарик - приложение к моей статье "Правила Ашманова". Там я говорю об особенностях управления программистами, а здесь собрал "практический материал" - высказывания, которые менеджер не должен понимать буквально.

Попробуйте узнать в них реальные ситуации из жизни своего проекта. Если вы заметили частое употребление сотрудниками приведенных ниже выражений, в проекте нужно срочно наводить порядок.

- Ну, не знаю, у меня на машине всё работает.

Комментарий: это неправда. То есть, конечно, что-то работает - после серии магических пассов, недоступных пользователю и тестировщикам.

- Да у вас просто "Винды" кривые.

Комментарий: это не имеет никакого отношения к делу. У большинства пользователей в мире "Винды" - кривые, а прикладные программы все-таки работают.

- Попробуйте перезапуститься. Думаю, всё заработает.

Комментарий: это маловероятно, хотя возможно. Но программа должна работать и без перезапуска.

- Как дела в проекте? Работа ведется!

Комментарий: "Работаем" - обычный ответ разработчика на вопросы менеджера. Помогает "отбить" две трети, а то и четыре пятых запросов о ходе проекта. Сам по себе этот ответ - не криминал, и на самом деле в разработке бывают периоды упорной работы "от забора до обеда", когда результатов не видно. Но частое повторение этой формулы подозрительно - она может служить и для сокрытия уже обнаружившихся проблем со сроками и трудоемкостью, которые разработчик надеется решить сам, не доводя до начальства.

- Я уже неделю ночами работаю, а вы меня укоряете за срыв срока.

Комментарий: ночная работа - это вовсе не доблесть. Скорее всего, просто у

программиста сложился такой режим (что часто бывает), а в сутки всё равно выходит 8-10 рабочих часов. Даже если и была бы переработка, то это недостаток организации работ.

- Нельзя подпускать к проекту этих маркетоидов, которые ничего не понимают в технологиях.

***Комментарий*:** маркетоиды не дают программировать всякие интересные штуки и вносят слишком много приземленных коммерческих требований.

- Эти менеджеры опять начнут совещаться, а мне работать нужно.

***Комментарий*:** действительно, часто совещания не имеют смысла, но совсем без них нельзя. А программисты с удовольствием участвуют в одних совещаниях, где идут обсуждения вообще и придумываются всякие классные идеи, и не любят другие - те, на которых наступает слишком большая ясность относительно состояния дел и выполнения планов.

- Чего там планировать, я быстрее сделаю и всё уже будет работать.

***Комментарий*:** это неправда. Скорее всего, будет сделано не совсем то и неработающее. А срок доводки окажется длиной в целый проект.

- Планировать разработку бессмысленно, жизнь всё равно богаче. Программные проекты всегда срывают сроки, потому что это сложное и творческое дело, вроде научных исследований.

***Комментарий*:** это миф. При правильном проектировании и планировании сроки разработки ПО возможно выдержать и это нужно делать.

- Нанимать персонал должен только технический менеджер проекта, потому что ему потом с ними работать.

***Комментарий*:** это часто приводит к неумолимому срабатыванию Закона Паркинсона - найму по знакомству ненужных, слабых или неконтролируемых сотрудников. Нанимать разработчиков должен высший менеджмент и по возможности через кадровое агентство, а технический менеджер - накладывать вето при необходимости.

- Если всё сделать общим образом, мы получим не только решение частной задачи, но и готовый программный продукт, который будем продавать другим, и таким образом всё окупим.

***Комментарий*:** это просто приятные фантазии. Разработка готового продукта стоит примерно в три раза дороже программы для собственных нужд (см. "Мифический человеко-месяц" Фредерика Брукса). Кроме того, никто ведь не изучал рынок на предмет выяснения, а нужен ли такой продукт, и сколько у него сильных конкурентов.

- К пятнице готово не будет, но в понедельник - точно. Или во вторник.

***Комментарий*:** скорее всего и во вторник ничего не будет. В лучшем случае будет не готовая версия, а нечто для показа из рук с объяснениями на пальцах, как всё будет потом.

- К сроку готово не будет, потому что сгорел жесткий диск и пропала работа за неделю (месяц).

***Комментарий*:** скорее всего, это неправда. Диск действительно сгорел, но причина срыва сроков не в этом. Кроме того, если бы работа ежедневно архивировалась, проблемы бы в любом случае не возникло.

- Срок сорван - а что вы хотели? С самого начала было ясно, что ресурсов не хватает.

***Комментарий*:** это точно неправда. В начале проекта никто не поднял тревоги, что мало ресурсов. И в середине проекта - тоже. Это просто самая распространенная "отмазка".

- Программа хорошо документирована на языке Си.

***Комментарий*:** программистская шутка "для своих", отражающая тот печальный факт, что никто не писал комментариев и документации к программам и не будет писать, если не заставить твердой рукой.

Игорь Ашманов