Адаптивное управление проектом разработки ПО

Архипенков Сергей
http://www.arkhipenkov.ru
email: sergey@arkhipenkov.ru

Аннотация

Только треть программных проектов завершается успешно. Главными причинами большинства неудач в разработке ПО являются не столько сложность и масштабность разрабатываемых продуктов, сколько применение неадекватных методов управления программными проектами. Разработка ПО отличается от материального производства, поэтому опыт управления, накопленный в этих отраслях, не всегда эффективен и применим в программных проектах. Управлять разработкой ПО надо иначе. Основные усилия руководителя, если он стремится получить наивысшую производительность проектной команды, должны быть нацелены на изучение и изменение объекта управления: людей и их взаимодействия.

Введение. Программирование новый вид человеческой деятельности

Как бы громко это не звучало, но это действительно так. В программировании есть ряд фундаментальных особенностей, которые коренным образом отличают разработку ПО от любой отрасли материального производства. Вот главные из них.

Программирование – это выражение коллективных идей на формальном языке. В отраслях материального производства путь от идеи до ее реализации выглядит следующим образом: научно-исследовательская работа, опытно-конструкторская работа, производство. В верхней части этой пирамиды находятся отраслевые научно-исследовательские институты, которые производят идеи и изобретают новые продукты. На втором этаже пирамиды работают конструкторы в конструкторских бюро, в задачу которых входит реализация нового продукта в чертежах деталей и технологиях изготовления и сборки. На нижнем уровне находятся производственные мощности - заводы, на которых инженеры и рабочие воплощают «в железе» чертежи и технологии. Если проводить аналогию, то программисты работают исключительно на вершине описанной пирамиды. Роль конструкторского бюро для программного проекта выполняют компилятор и сборщик программ. А аналогом завода в производстве ПО, который переводит конструкторскую документацию в продукт, доступный потребителю, служит вычислительный комплекс, на котором развертывается и выполняется созданная программа.

В программировании отсутствует теория. В программировании нет системы знаний о закономерностях создания программ. Наработки математиков в области логики, теории информации, численных методов, реляционной алгебры, теории графов и некоторых других дисциплинах на долю процента не покрывают многообразие программистских задач. Даже выдающиеся программисты не возьмут на себя смелость утверждать об архитектуре новой программной системы то, что она будет успешной. Материальное производство опираются на достаточно хорошо изученные законы математики, физики и химии.

В программировании отсутствуют средства визуализации. «Программный продукт невидим и невизуализуем. Геометрические абстракции являются мощным инструментом. План здания помогает архитектору и заказчику оценить пространство, возможности перемещения, виды. Становятся очевидными противоречия, можно заметить упущения. Масштабные чертежи механических деталей и объемные модели молекул, будучи абстракциями, служат той же цели. Геометрическая реальность схватывается в геометрической абстракции» [1]. У программного обеспечения нет геометрического представления. Поэтому большая часть усилий участников разработки ПО тратится на синхронизацию ментальных моделей программного продукта.

Все, что составляет программный проект, очень изменчиво, неопределенно и плохо предсказуемо. Неопределенность, изменчивость и непредсказуемость начинаются на стадии определения концепции программного продукта и сопровождают его на протяжении всего жизненного цикла. Никому не придет в голову вносить архитектурные изменения в уже построенное здание. В программировании это обыденная практика.

Программирование можно сравнить, разве что, с созданием поэмы с заданным качеством и в определенный срок коллективом из двадцати или двухсот авторов. При этом в поэме не должно быть ни одной лишней или пропущенной запятой.

Классические методы управления не работают

Можно провести аналогию между эволюцией методов, применяемых в системах автоматиче-

ского управления летательными аппаратами, и развитием подходов к управлению программными проектами.

«Как получится». Разомкнутая система управления. Например, провели kick-off meeting, поставили задачи разработчикам и пошли готовить банкет по поводу успешного завершения проекта. Аналогия: баллистический полет. Можно, но недалеко и неточно.

«Водопад». Жесткое управление с обратной связью. Расчет опорной траектории (план проекта), измерение отклонений, коррекция и возврат на опорную траекторию. Лучше, но не эффективно.

«Гибкое управление». Расчет опорной траектории, измерение отклонений, расчет новой попадающей траектории и коррекция для выхода на нее. «Планы - ничто, планирование - все» (Эйзенхауэр, Дуайт Дэвид)

«Метод частых поставок». Самонаведение. Расчет опорной траектории, измерение отклонений, уточнение цели, расчет новой попадающей траектории и коррекция для выхода на нее.

Все эти подходы перестают работать в случаях, когда структура и свойства управляемого объекта нам не известны и изменяются со временем. Эти подходы так же не помогут, если текущие свойства не позволяют объекту двигаться с требуемыми характеристиками. Например, летательный аппарат не может развить требуемое ускорение или разрушается при недопустимой перегрузке. Аналогично, если рабочая группа проекта не может обеспечить необходимую производительность и постоянно работает в режиме аврала, то это приводит к уходу профессионалов из проекта.

Адаптивное управление

Адаптивное управление, дополнительно к прямым управляющим воздействиям, направлено на изучение и изменение свойств управляемого объекта. Продолжая аналогию с управлением летательными аппаратами, - это расчет опорной траектории, измерение отклонений, уточнение цели, уточнение объекта управления, адаптация (необходимое изменение) управляемого объекта, расчет новой попадающей траектории и коррекция для выхода на нее.

Результатом управления проектом (***r***) являются: соответствие спецификациям, срок и бюджет. Множество управляющих воздействий (***u***) ограничено: составить план работ, расставить приоритеты, назначить на работы исполнителей. Как правило, таким и только таким управлением (администрированием) занимаются менеджеры - поклонники диаграмм Ганта и фанаты MS Project. Если у команды проекта низкая производительность, то единственный путь ее повысить при административном подходе - это постоянное давление, авралы, сверхурочные и субботники. Работать больше, это совсем не значит - работать продуктивнее. Скорее наоборот. Излишнее давление и суета приводят к непродуманным решениям, большому проблемному коду и многочисленным последующим переработкам. Для подобных руководителей любой программный проект будет безнадежным.

Для того чтобы понять структуру и свойства объекта и воздействовать на них с целью приведения к желаемому состоянию, в проекте должен быть дополнительный контур обратной связи - адаптер (Рисунок 1).

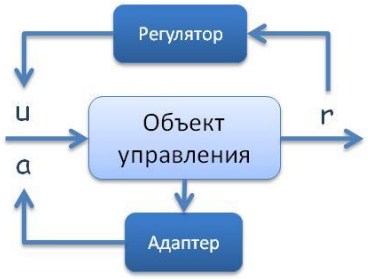

Рисунок 1. Дополнительный контур обратной связи в системе адаптивного управления

Вместе с тем, известно, что производительность разных программистов с одинаковым стажем может отличаться в десятки раз. Это типично для научных исследований, но в материальном производстве это просто немыслимо. Утверждаю, что производительность одного и того же программиста может так же отличаться в десятки раз. Заставьте лучшего в мире бегуна бегать в мешке, и он покажет в 10 раз худший результат. Заставьте лучшего программиста заниматься «сизифовым трудом»: плодить документацию (которую, как правило, никто не читает) в угоду «Методологии» (именно с большой буквы ‘М’), - и его производительность снизится в 10 раз.

Поэтому, основные усилия руководителя, если он стремится получить наивысшую производительность рабочей группы, должны быть направлены на изучение и изменение объекта управления: людей и их взаимодействия. Следовательно, задачу адаптивного управления мы можем разделить на две подзадачи:

1. Обеспечить эффективность каждого участника рабочей группы.
2. Обеспечить эффективные процессы взаимодействия.

Все люди разные и ситуаций, в которых они могут находиться в ходе проекта, бесчисленное множество. Бойтесь стереотипов. Если вы не учитываете индивидуальные особенности конкретной личности, то эффективность ваших взаимодействий сильно снижается.

Модель объекта управления нам неизвестна, следовательно, не может существовать исчерпывающий набор правил, тип «если…, то…», по которым смог бы действовать руководитель. Поэтому, сколько людей и ситуаций, столько и вариантов решений должен иметь эффективный руководитель в своем запасе. «Если у руководителя в руках только молоток, то все вокруг будут похожи на гвозди».

Принцип 1. *«Принцип достаточного разнообразия». Для «хорошего» управления количество возможных состояний управляющего устройства (разнообразие) должно быть не меньше, чем количество состояний объекта управления [2].*

Руководитель при поиске решения опирается на свой багаж знаний и умений. Он пытается понять каждого участника, классифицировать состояние, найти в своем опыте похожую ситуацию и адаптировать ранее использованное успешное решение применительно к данному конкретному случаю или изобрести новое решение. Таким образом, руководитель стремится помочь человеку (объекту управления) перейти в новое более эффективное с точки зрения целей проекта состояние.

Затем руководитель должен наблюдать за результатами своего воздействия – это и есть дополнительный контур обратной связи. Необходимо помнить, что понять человека можно, только слушая и *слыша*, что он говорит. Руководитель, который в течение недели не пообщался индивидуально с каждым из своих прямых подчиненных, зря получает зарплату. И совсем не обязательно разговор должен идти о статусе проектных работ. Порой, достаточно поговорить о погоде, кино или футболе.

После этого руководитель анализирует полученные результаты и аккумулирует новый опыт (положительный или отрицательный) в своей «базе знаний». Как это может происходить на практике, будет далее проиллюстрировано на примерах.

Чем опытней руководитель, тем точнее он может распознать и классифицировать сложившуюся ситуацию, тем больше в его «базе знаний» прецедентов, используя которые, он может синтезировать решение для данного конкретного случая. Именно поэтому в управлении программными проектами в первую очередь ценится опыт руководителя и только потом, возможно, его звания и знания.

Обеспечить эффективность каждого участника

Для того чтобы ваш сотрудник мог эффективно решить поставленную вами задачу, необходимо и достаточно выполнение четырех условий.

Принцип 2. *«Четыре условия эффективной работы».*

1. *Понимание целей работы.*
2. *Умение ее делать.*
3. *Возможность ее сделать.*
4. *Желание ее сделать.*

Для того чтобы обеспечить выполнение этих условий, руководитель должен уметь эффективно выполнять четыре функции.

Принцип 3. *«Четыре функции руководителя».*

1. *Направлять.* Если сотрудник не понимает что делать, задача руководителя - обеспечить общее видение целей и стратегии их достижения.
2. *Обучать.* Если сотрудник не умеет, задача руководителя – «обучать», быть наставником и образцом для подражания.
3. *Помогать.* Если у сотрудника не может выполнить работу, задача руководителя – «помогать», обеспечить исполнителя всем необходимым, убрать препятствия с его пути.
4. *Вдохновлять.* Если у сотрудника не достаточно желание выполнить работу, задача руководителя – «вдохновить», обеспечить адекватную мотивацию участника на протяжении всего проекта.

Известно, что ни одна задача не будет решена за любое, отведенное на это время, если человек не захочет ее сделать. Он всегда найдет для оправдания этого 100 «объективных» причин, вместо того, чтобы найти хотя бы одну возможность для решения задачи.

У каждого участника рабочей группы должна быть личная цель (внутренняя мотивация), которую он сможет достичь, продвигая проект к успеху. Начните с себя! Вам нужно четко понимать, в чем состоит ваш выигрыш в случае успешного завершения проекта. Добиться от участников приверженности проекту больше, чем имеете вы сами, вам не удастся.

Если у участника нет такой личной значимой цели, избавьтесь от него. Иначе вам придется потратить все свое время на «промывание его

мозгов» и попытки мотивировать его на эффективную работу.

Необходимо помнить, что по ходу проекта мотивы людей, как правило, изменяются.

Обеспечить эффективные процессы взаимодействия

В разработке ПО многие уже признали, что наиболее эффективные производственные процессы складываются в самоуправляемых и самоорганизующихся рабочих командах, для которых характерны ясность общих ценностей и целей, взаимное доверие, самоконтроль, взаимопомощь, взаимозаменяемость, коллективная ответственность за результаты труда, всемерное развитие и использование индивидуального и группового потенциалов.

Эффективные команды не образуются сами по себе, они кристаллизуются вокруг признанного лидера. Как не бывает лидеров без последователей, так и не бывает команд без лидеров. Поэтому первый шаг руководителя при создании эффективной команды - это стать лидером, вокруг которого сможет сплотиться рабочий коллектив.

Принцип 4. *«Принцип лидерства». Руководителю программного проекта недостаточно быть хорошим управленцем, он должен стать признанным лидером.*

Лидера нельзя назначить. Лидер должен быть признан коллективом. Чтобы руководитель получил признание в качестве лидера, необходимо выполнение следующих двух условий.

1. Признание коллективом профессиональной компетентности и превосходства руководителя.
2. Полное доверие коллектива к действиям и решениям руководителя, признание его исключительных человеческих качеств, убежденность в его честности, порядочности, вера в его искренность и добросовестность.

Для того чтобы получить признание, если, конечно, руководитель его объективно заслуживает, он должен использовать принцип номер 5.

Принцип 5. *«Четыре стратегии лидера». Не существует одной лучшей стратегии руководства. В зависимости от готовности участников рабочей группы выполнять задания руководителя, он должен использовать одну из 4-х стратегий [3]:*

1. *S1. «Директивное управление».* Руководитель говорит, указывает, направляет, устанавливает. Жесткое назначение работ, строгий контроль сроков и результатов.
2. *S2. «Объяснения».* Лидер "продает", объясняет, проясняет, убеждает. Сочетание директивного и коллективного управления. Объяснение своих решений.
3. *S3. «Участие».* Лидер участвует, поощряет, сотрудничает, проявляет преданность. Приоритетное коллективное принятие решений, обмен идеями, поддержка инициативы подчиненных.
4. *S4. «Делегирование».* Лидер делегирует, наблюдает, обслуживает. «Не мешать» - пассивное управление сформировавшегося лидера.

Мало стать лидером, надо еще суметь сплотить коллектив. Эксперты в области командного менеджмента [4] выделяют четыре обязательных последовательных стадии, через которые должна пройти рабочая группа прежде, чем она станет эффективной командой, это:

1. Формирование (Forming). Характеризуется избытком энтузиазма, связанного с новизной. Люди должны преодолеть внутренние противоречия, переболеть конфликтами прежде, чем сформируется действительно спаянный коллектив.
2. Разногласия и конфликты (Storming). Самый сложный и опасный период. Мотивация новизны уже исчезла, а сильные и глубокие стимулы у команды еще не появились. Неизбежные сложности или неудачи порождают конфликты и «поиск виновных». Участники команды методом проб и ошибок вырабатывают наиболее эффективные процессы взаимодействия.
3. Становление (Norming). В команде растет доверие, люди начинают замечать в коллегах не только проблемные, но и сильные стороны. Закрепляются и оттачиваются наиболее эффективные процессы взаимодействия. На смену битве амбиций приходит продуктивное сотрудничество. Четче становится разделение труда, исчезает дублирование функций.
4. Отдача (Performing). Команда работает эффективно, высок командный дух, люди хорошо знают друг друга и умеют использовать сильные стороны коллег. Все стремятся придерживаться выработанных общих процессов. Высок уровень доверия. Это лучший период для раскрытия индивидуальных талантов.

Часто случается, что рабочая группа вязнет на одной из стадий и никогда не достигает плато наивысшей производительности.

Принцип 6. *Принцип четырех «П». Эффективность разработки ПО есть функция четырех «П»: Продукта, Проекта, Персонала и Процесса.*

Не существует одного наилучшего процесса разработки ПО. Процесс должен постоянно адаптироваться в соответствии с технической сложностью продукта, текущим масштабом программного проекта, квалификацией и сплоченностью персонала.

Если руководитель не будет прилагать дополнительные усилия команда, рано или поздно, начнет «сползать» с плато наивысшей эффективности в состояние застоя и стагнации (Рисунок 2). Помните, что окружение и команда изменяются по ходу проекта. Прежняя мотивация ослабевает или перестает действовать.

Принцип 7. *«Принцип цикличности». Четыре стадии развития команды должны циклически повторяться, чтобы обеспечить непрерывный рост эффективности.*

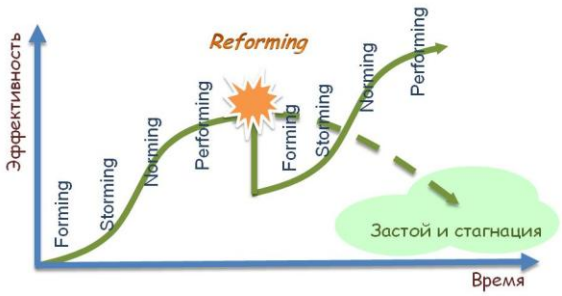

Рисунок 2. Перестройка (Reforming) и перевод команды проекта на новый уровень производительности

Изменяйте правила и процессы. Отказывайтесь от того, что перестало действовать или стало работать неэффективно. Перестраивайтесь и возвращайте команду в стадию Forming. Это позволит ей снова, пройдя через все этапы становления, выйти на новый более высокий уровень производительности.

Разумеется, делать это следует, после сдачи очередного релиза программного продукта, ну и, возможно, в случае глубокого кризиса проекта.

Заключение. Четырехкомпонентная модель программиста

Ранее мы говорили, что результатом управления проектом являются: соответствие спецификациям, срок и бюджет. Утверждаю, что у успешного программного проекта должен быть еще один, четвертый результат: *каждый участник команды уходил с работы в 18:00 с чувством победы.* Чувство победы – это чувство удовольствия человека, который приближается к своей цели. А все, что человек делает с удовольствием, он делает максимально эффективно.

Программист состоит из четырех компонентов: тело, сердце, разум и душа.

1. Телу необходимы деньги и безопасность.
2. Сердцу - любовь и признание.
3. Разуму – развитие и самосовершенствование.
4. Душе – самореализация.

Предоставьте все это вашим сотрудникам, и эффективность их труда возрастет многократно. В противном случае профессионалы, найдут все это в другой команде, а в вашей останутся только неудачники.

Ссылки

[1] Брукс Фредерик, "Мифический человеко-месяц, или Как создаются программные комплексы", Пер. с англ., СПб., Символ-Плюс, 1999.

[2] У.Р.Эшби “Введение в кибернетику” М, ИЛ, 1959

[3] Hersey P., Blanchard K.H. “Management of Organizational Behavior”, 6th ed., Englewood Cliffs: Prentice-Hall, 1993.

[4] Стивен У. Фланнес, Джинджер Левин, «Навыки работы с людьми для менеджеров проектов», М., Технологии управления Спайдер, 2004 г.