case

27 июля 2010 в 21:20

Agile команда и контракты с фиксированной ценой

перевод

Управление проектами*

Контракты с фиксированной ценой — это зло, вот что можно услышать от адептов agile. С другой стороны, такие контракты — это реальность, с которой сталкиваются многие agile команды. Но что, если мы попытаемся укротить это зло, а не бороться с ним?

Как компания может выполнить такой контракт с использованием гибкой методологии для достижения лучших результатов с меньшими рисками? В этой статье мы постараемся ответить на эти вопросы.

Так давайте же начнем с самого контракта.

Фиксированная цена, время и объем обязательств

Такие контракты фиксируют сразу три магических фактора — деньги, время и объем обязательств. Являются ли цена и сроки проблемой для agile команд? Ну, не должны быть. На самом деле, таймбоксинг (timeboxing) — это обычная практика. Ограничение бюджета только помогает таймбоксингу лучше работать.

Настоящей проблемой контрактов с фиксированной ценой является объем обязательств, ведь обычно прописано, что именно должно быть сделано, вместо того, сколько именно нам следует работать.

Почему клиенты так одержимы фиксированным объемом обязательств? Мы понимаем, что они хотят знать, сколько им придется платить (а кто не хочет?) и когда они получат результат. Единственное, чего они не знают, это что именно они хотят получить (даже если говорят, что знают).

Корни проблемы с фиксированным объемом обязательств в следующем:

- Недостаток доверия между заказчиком и исполнителем
- Недостаток понимания процесса разработки программного обеспечения
- Непонимание того, что значит «объем обязательств»

Каждый контракт с фиксированной ценой имеет сопроводительный документ, «Техническое задание» или что-то в этом роде. Этот документ призван снизить риски упустить что-то важное и создать общее понимание того, что должно быть сделано, получается некая иллюзия предсказуемости.

Основные **ошибочные предположения при фиксировании объема обязательств**:

- Чем больше деталей мы включим в определение объема обязательств, тем лучше мы поймем друг друга
- Четко определенный объем обязательств предотвратит изменения
- Фиксированный объем обязательств необходим, чтобы лучше подсчитать цену и время

От фиксированного объема обязательств к фиксированному бюджету

Теперь, когда мы понимаем, что основной конфликт между применением гибкого мышления и контракта с фиксированной ценой заключается в фиксированном объеме обязательств, мы можем сосредоточиться на преобразовании фиксированного объема обязательств в фиксированный бюджет.

Это означает, что вместо предоставления детальной спецификации требований нам надо сосредоточиться на отслеживании как можно большего числа требований в виде пользовательских историй. Нам необходимо построить первоначальный список задач по продукту, который мы попытаемся оценить с помощью техники баллов сложности (story points) и общекомандных техник оценки, таких, как покер планирования (Planning poker).

Во-первых, мы должны понимать, что более высокий уровень детализации программного обеспечения означает совершенно разные вещи для обеих сторон договора. Компании-разработчики программного обеспечения обычно сосредотачиваются на технических деталях, тогда как другая сторона больше ориентирована на финансовую сторону вопроса. Спецификации требований обычно создаются компаниями-разработчиками программного обеспечения и являются предохранительными. Вот где вступают в игру три ключевых игрока.

Пользовательские истории как способ выражения требований понятны и для заказчиков, и для исполнителей. Понимание создает доверие и ощущение общего видения проекта. Истории пользователей легко записать и стереть, особенно если писать их на карточках. Еще они ориентированы на функционал, поэтому обеспечивают четкое видение реального объема работ по проекту, и мы можем сравнить их между собой по срокам и усилиям.

Баллы сложности (story points) как способ оценки историй вначале сложно объяснить клиенту, поскольку это не обычный способ выражения усилий. Но зато они снижают опасность недооценки объема обязательств. Как это происходит? Баллы сложности по своей природе относительны, фокус здесь либо на общий объем, либо на группу историй, тогда как традиционная оценка (как правило, делается в человеко-часах) пытается анализировать каждую функцию в отдельности.

Критерий готовности (definition of done) — это другой метод построения доверия и общего понимания процессов и планирования в проекте. Обычно, когда клиент впервые сталкивается с пользовательскими историями, даже если ему нравится, как они написаны, для него неочевидно, что такое реализация истории. Команды, которые обсуждают критерий готовности с клиентом, лучше понимают, чего он от них ожидает. Они также дают более точную оценку. К тому же на стороне клиента критерий готовности создает условия для более высокого уровня принятия пользовательской истории.

Например, критерий готовности для историй в свежем веб-приложении может включать следующие пункты:

- Протестировано под IE 7/8, Firefox 3.x и Chrome
- Созданы соответствующие разделы в «Руководстве пользователя»

Позже команда может создавать другие, более глубокие внутренние элементы критерия готовности, но на стадии подготовки контракта эти общие указания дают клиенту и команде понимание того, что мы хотим создать в рамках объема работ по проекту.

Надеемся, что учитывая наш критерий готовности, мы придем к некому значению, которое разложит **бюджет проекта по баллам сложности**. И это значение, а не пользовательские истории, которые за ним стоят, это первое, что должно быть зафиксировано в контракте. И это уже открывает двери изменениям.

Хотя у нас фиксированный объем обязательств, мы все равно хотим гибко принимать изменения. У нас есть инструменты (истории и баллы сложности), которые можно использовать, чтобы сравнивать требования. Это позволяет нам обмениваться требованиями, оставаясь в рамках фиксированного объема обязательств. И если мы можем оставаться в этих рамках, то сможем и вписываться в фиксированные сроки и цену.

Первоначальная оценка

Самая сложная часть подготовки контракта с фиксированной ценой — это определение цены и графика, который будет зафиксирован на основе четко определенного объема обязательств. Давайте посмотрим, что может привнести команда в такой контракт при помощи гибкой первоначальной оценки проекта.

Во-первых, **обучение**. Встречайтесь с клиентом и рассказывайте, как вы будете работать. Следует рассказать ему, что такое пользовательские истории, как мы собираемся их оценивать и что такое критерий готовности. Зачастую, это необходимо делать еще до подготовки предложения на клиентский запрос на предложение (RFP).

Следующий шаг — это **сбор пользовательских историй**. Его можно организовать как несколько быстрых сессий в течение 1-2 дней. Этого времени достаточно, чтобы найти большинство историй, которые сформируют видение продукта, не забираясь глубоко в дебри функционала. С этой точки зрения очень важно обсудить с клиентом критерий готовности для историй, итераций и релизов.

Нам следует знать:

- Окружение, в котором нужно тестировать истории (количество браузеров или мобильных платформ, или операционных систем)
- Необходимая документация
- Где нужно разворачивать завершенные истории, чтобы клиент мог на них посмотреть
- Что следует делать клиенту (например, принимать участие в демонстрационной сессии)
- Как часто нужно встречаться и кто участники

Эти и, наверное, еще многие факторы повлияют на оценку и дадут общее понимание ожиданий и качества с обеих сторон. Они также помогут сделать оценку менее оптимистичной, каковой она получается, если команда обсуждает только технические аспекты реализации пользовательских историй.

Обсудив с заказчиком набор историй и критерий готовности, мы можем начинать оценку. Это хорошо известная

часть процесса. Здесь самое важное действие — это привлечение как можно большего числа будущих членов команды, чтобы оценка проводилась коллективно.

Такие техники, как planning poker, снижают риск недооценки проекта из-за мнения отдельного члена команды, особенно если этот член команды еще и самый опытный, а именно так и случается, когда оценкой занимается один человек. Также важно, чтобы истории оценивались людьми, которые на самом деле будут реализовывать систему.

> **Шкала подобная ряду Фибоначчи** (1, 2, 3, 5, 8, 13, 20, 40, 100) весьма удобна для оценки историй в баллах сложности. Относительная оценка начинается с отбора самых простых или маленьких историй. Они получают 1-2 балла сложности — это базовый уровень для дальнейшей оценки.

На самом деле, во время первоначальной оценки часто сложно оценить истории, используя самые низкие значения, такие как 1 и 2. Дело в том, что чем выше оценка, тем меньше мы знаем об истории. Поэтому оценка в баллах проще на этом начальном уровне, ведь намного проще сказать, что история А в 2 раза сложнее, чем история В, чем сказать, что история А займет 25 человеко-часов (помните о критерии готовности?), а история В займет 54 часа.

Это отлично работает, даже если мы возьмем 3-х или 5-ти бальные истории за базовый уровень, в таком случае нам проще будет разбить их на мелкие истории уже на стадии разработки. Опасайтесь историй с 20, 40 или 100 баллами. Такой способ оценки предполагает, что мы ничего не знаем о том, что будет реализовываться, поэтому нужно сразу же более детально обсудить все с заказчиком, а не радостно вписывать их в контракт.

Результатом оценки является общее количество баллов сложности, которые описывают первоначальный объем работ по созданию продукта. Именно это число должно быть зафиксировано как объем обязательств по контракту, а не отдельные пользовательские истории.

Фиксирование цены и времени

Общее количество баллов сложности, вычисленных на основе набора историй, напрямую не дает нам цену и время. Как же их заполучить? Нам нужны магические числа.

Не стоит удивляться, ведь в контрактах с фиксированной ценой всегда существуют магические числа. Все оценочные значения всегда были такими числами. Сейчас мы изменяем способ оценки, но в конце нам все равно нужно прогнозировать будущее на основе прошлого опыта. Так что же нам следует знать? **Прогнозируемая производительность команды.**

Предположим, мы оценили наши истории общим количеством в 300 опорных точек. Нам нужно определить производительность команды, основываясь на следующем:

- Производительность команды на прошлых проектах
- Количество действий, выполняемых на каждой итерации (см. Критерий готовности)
- Легкость общения с клиентом (конечно, идеальный и труднодостижимый вариант, это когда клиент находится на месте как часть команды, но обычно это не так. У разных заказчиков есть разное количество времени, которое они готовы потратить на общение с командой) — чем меньше времени есть у клиента на нас, тем ниже наша производительность
- Просто ощущения, которые у нас вызывает объем работ по проекту (да, выдаем желаемое за действительное)

Допустим, наша команда состоит из 5 человек — 3 программиста, 1 тестировщик и 1 руководитель группы — которые будут общаться с заказчиками и заинтересованными лицами, так же, как с другими людьми вне команды (например, со Scrum Master в методологии Scrum).

Допустим, что наша команда постарается достичь скорости работы в 20 опорных точек за 2-х недельную итерацию. Это и есть наше главное магическое число в контракте. Многие факторы могут повлиять на производительность в течение всего времени работы над проектом, тем не менее, если команда, с которой мы работаем, не нова, и сам проект не является чем-то совершенно неизвестным, то эти числа могут быть взяты из прошлого опыта.

Теперь мы можем столкнуться с одним из двух ограничений, навязываемых нам клиентом в контракте:

- Клиент хочет получить продукт так быстро, как только мы сможем его сделать (и, желательно, еще быстрее)
- Клиент хочет, чтобы мы сделали максимум к дате X (которая является нашим конечным сроком)

Сценарий 1

В первом сценарии мы можем определить расчетное время завершения при помощи следующей формулы:

*время = (<баллы сложности> / <производительность>) * <длина итерации>*

В нашем сценарии будет:

*время = (300 точек / 20 точек) * 2 недели = 30 недель (или 15 итераций)*

Если расчетное время неприемлемо, то единственное, что мы можем изменить — это производительность команды. Как бы там ни было, зависимость не линейна, и если мы увеличим размер команды в два раза, производительность в два раза не увеличится.

Сценарий 2

Мы знаем, что срок окончания проекта наступит через 20 недель, а это всего 10 из наших итераций, соответственно мы сможем выполнить только 20 баллов из нашего объема работ. Теперь зависит от клиента, что можно удалить из объема работ без потери возможности сделать что-то, имеющее необходимую ценность.

Здесь мы также можем (до некоторой степени) изменить размер необходимой команды, чтобы повысить производительность.

Зная необходимый размер команды и конечную дату, мы, наконец, можем рассчитать основную цену контракта — стоимость рабочего времени:

*деньги = <кол-во часов на итерацию> * <кол-во итераций> * <люди> * <цена часа работы>*

В нашем примере получаем:

*деньги = 80 часов * 15 итераций * 4 человека * $100= $480,000*

Теперь у нас есть значения контракта с фиксированной ценой:

- цена — $480,000
- время — 15 итераций (30 недель)
- объем — 300 опорных точек

Эти простые расчеты, конечно, являются лишь частью стоимости, которая появится в контракте в конечном итоге, но это также и самая трудная для определения часть. Мы хотели показать, что то, как мы работаем на гибких проектах можно перенести и в процесс переговоров по контракту.

Есть еще один вопрос. Клиент спрашивает: «Почему так дорого?» Вот где на самом деле начинаются переговоры, и единственный фактор, который в силах изменить компания-производитель, это стоимость человеко-часа. Это те $100, о которых мы договариваемся. Не длина итерации, даже не количество итераций. У нашей команды нет супер-сил, и мы не станем работать в два раза быстрее, если просто об этом договоримся.

> **Это те деньги, которые мы согласны потерять.** Если мы говорим, что можем снизить цену, то только потому, что мы заработаем меньше, а не потому, что станем работать быстрее.

Отслеживание прогресса и бюджета

Теперь наш контракт подписан (это ведь было просто, не так ли?), пора создавать программное обеспечение в согласованных рамках. Вы, наверное, уже слышали, что гибкая методология не решит ваших проблем, но сделает их видимыми. Поэтому мы хотим пораньше узнавать, как идут дела, чтобы предпринять необходимые действия. Ведь есть вероятность, что наше магическое число в контракте (наша производительность) было ошибочным.

Burndown диаграммы — это весьма распространенный способ отслеживания прогресса во многих agile проектах.

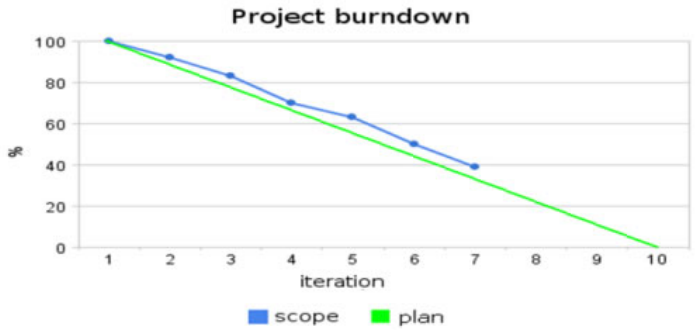

Рис. 1 — Запланированный прогресс и реальный прогресс.

Они отлично подходят для визуализации запланированного прогресса и реальности. Например, burndown диаграмма на Рис. 1 выглядит неплохо:

Мы немного не вписываемся в план, но это не значит, что мы сделали большую ошибку в определении производительности в процессе обсуждения контракта. Вероятно, многие команды захотели бы, чтобы их графики выглядели именно так. Но проблема в том, что этот график отображает только два из трех факторов — объем обязательств и время. А как же деньги?

Рис. 2 — Прогресс выполнения работ и расходования бюджета.

Диаграмма на Рис. 2 показывает две шкалы — объем работ и бюджет. Эти два фактора выражены здесь в процентах неспроста. Ведь нет другого способа сравнить эти два показателя (один считается в баллах сложности, а второй — в человеко-часах или потраченных деньгах).

Вторая диаграмма показывает не очень радужные перспективы. Мы тратим больше денег, чем ожидали. Возможно, из-за того, что мы привлекли дополнительные ресурсы, чтобы достичь ожидаемой продуктивности. Когда все три фактора показаны на одной диаграмме, мы сразу видим проблемы, а итерация 4 на этом примере — это момент, когда нам стоит поговорить с клиентом, пока не стало слишком поздно.

Чтобы таким образом отслеживать объем работ и бюджет, нам необходимо:

- Отслеживать выполненные баллы сложности на каждой итерации
- Отслеживать реально потраченное время на каждой итерации (в человеко-часах)
- Пересчитывать баллы в проценты от общего объема работ и рисовать шкалу на основе процентного показателя от общего объема
- Пересчитывать бюджет, зафиксированный в контракте (или его часть), в общее количество человеко-часов — это наши 100% бюджета — и рисовать шкалу расходования бюджета на основе процентного показателя от общего бюджета

В предыдущем примере у нас было:

- Объем работ — 300 баллов сложности
- Бюджет на 30 недель * 4 человека * 40 часов в неделю = 4,800 человеко-часов

Это наши общие показатели. Расход 120 часов на 6 баллов сложности означает, что мы расходовали 2,5% бюджета на выполнение 2% объема работ.

Принятие изменений

Нам нужно еще раз вернуться к тому моменту, когда мы изменили фиксированный объем обязательств на фиксированный бюджет. 300 баллов сложности из предыдущих примеров помогут нам изменить содержание изначального списка пользовательских историй. Это один из наиболее важных моментов, к которому мы хотим придти, применяя гибкую методологию к контракту с фиксированной ценой.

Agile легко принимает изменения, нам лишь нужно рационализировать управление изменениями в контракте с фиксированной ценой. Это всегда было и остается сложной частью работы, но с небольшой помощью, меняя фокус с анализа требований на ограничения и лимиты на ранних этапах процесса, мы приветствуем изменения на любом этапе проекта.

Сложность состоит в том, чтобы убедить клиента в том, что истории можно заменять, поскольку они должны быть сравнимы в плане количества необходимых усилий, которые нужно приложить для их выполнения. Поэтому, если по какой-либо причине у клиента возникает новая восхитительная идея, что нам нужно заняться новым набором историй (например, оцененным в 20 баллов сложности), то только от клиента зависит то, что нам придется убрать историю на 20 баллов из конца списка задач, чтобы освободить место для новой.

Или, возможно, клиент хочет добавить еще одну итерацию (помните производительность в 20 баллов за итерацию?). Рассчитать цену этих новых историй очень просто:

*деньги = 1 итерация * 80 человеко-часов * 4 человека * $100 = $24,000*

И самое сложное в таком виде контрактов, это когда мы в процессе работы над проектом узнаем, что некоторые истории займут больше времени, чем мы ожидали, поскольку их оценивали «на глаз», а теперь мы знаем о них намного больше, чем вначале. Но даже такая ситуация не всегда ведет к большим проблемам, ведь в это же время некоторые истории займут меньше времени, чем запланировано. Поэтому, опять повторю, отслеживание в течение выполнения работ по контракту предоставит вам очень ценную информацию. Ранние разговоры о проблемах помогут в переговорах, ведь мы будем обсуждать действия, которые нужно предпринять, чтобы избежать таких проблем, вместо того, чтобы говорить о спасательных операциях после грандиозной катастрофы.

Завоевание доверия

Чтобы использовать все эти техники в контрактах с фиксированной ценой, необходима одна вещь — **доверие**. Но, как мы знаем, невозможно завоевать доверие просто описывая то, какую замечательную работу выполнит наша команда для заказчика.

Принципы гибкой методологии предоставляют отличный путь для решения этой проблемы. С каждой итерацией мы создаем некую ценность для клиента. Но самое важное, что мы сначала фокусируемся на самых ценных функциях. Поэтому, лучший способ завоевать доверие клиента — это разбить контракт на несколько частей.

Начните с малого, с пилотной части в 2-3 итерации (которые также будут с фиксированной ценой, но короче). Программное обеспечение, полученное в итоге, должно оправдать ожидания клиента по ценности. Фактически, это должны быть некоторые рабочие части ключевых функций. Работающий продукт докажет, что вы сможете сделать и остальное. Также это даст вам возможность подтвердить первоначальные предположения о производительности и обсудить следующую часть.

Время, потраченное на пилотную часть, также должно быть небольшим по сравнению с объемом работ, которые еще предстоит сделать. В таком случае, если клиент будет неудовлетворен результатами, он сможет отказаться от ваших услуг раньше, чем будет слишком поздно, ему не придется продлевать контракт и в конце концов провалить проект.

Резюме

Контракты с фиксированной ценой часто воспринимают как очень и очень вредные, и многие адепты agile говорят, что их просто стоит избегать. Но в большинстве случаев избегать их не получается, поэтому необходимо искать способы заставить их работать на нашу цель — создание качественного программного обеспечения.

На самом деле, некоторые аспекты фиксированных факторов даже лучше подходят для agile команд, ведь мы привыкли работать во временных блоках (time boxes), а фиксированная дата в контракте (а также фиксированная цена) и является этими временными блоками и границами. Единственное, что действительно нас беспокоит, это объем обязательств, и в этой статье мы постарались собрать идеи о том, как справляться с этим ограничением.

Замысел этой статьи был не в том, чтобы сказать, что agile — это панацея от всех проблем контрактов с фиксированной ценой, а в том, чтобы показать другие пути работы в этой области с использованием гибкой методологии.

Об авторе

Марсин Небудек (Marcin Niebudek) — бизнесмен и совладелец компании Agilers, а также создатель **tinyPM**,

небольшого инструмента для управления agile проектами. Марсин безумно идеалистичен в отношении разработки программного обеспечения и твердо верит в этот путь. Ему нравится экспериментировать со всеми направлениями технологий Agile и Lean Development. Еще он любит писать программное обеспечение, читать о нем, думать о нем и вести блог о нем. Он — автор блога «Battle for Agility», который описывает движение agile в польской IT индустрии. Марсин — Java-разработчик с опытом работы более 8 лет в области создания корпоративных систем с использованием всех видов технологий из арсенала Java EE.

Agile, SCRUM, Управление проектами

+31 3028 83

Перевод: **Marcin Niebudek**

Панкратов Слава @case карма 56,9 рейтинг 0,0

Комментарии (26)

отслеживать новые: в почте в трекере

yshilyaev 27 июля 2010 в 23:19 # +7

Букв много, но про проблему работы по скраму с фиксированной ценой — практически ничего.
Классически клиент по договору Fix price желает зафиксировать все 3 параметра: стоимость, объем работ, сроки. Т.е. там тоже есть люди, которые читали PMBoK. Рассказами про стори-поинты и числа Фибонначи людей от бизнеса можно вгнать в ступор, но убедить так работать — невозможно. По крайней мере у нас.
Челов от бизнеса надо убеждать по-другому. :)

steel_ne 28 июля 2010 в 09:53 # 0

Челам от бизнеса говорится — это стоит 480 косарей. А вот откуда это число…

yshilyaev 28 июля 2010 в 10:23 # +2

Сказал ты челам, что это стоит полмульта. И дальше? На релизах они тебе выкатили еще 10ть фич. Искусство в работе по фикс-прайсу заключается не в оценках как таковых, а в составлении контрактов. Как составить контакт так, чтобы либо scope не был зафиксирован, либо цена. Еще лучше, когда зафискирована ставка, а scope и бюджет нет. Тогда это уже не фикс-прайс, а time&material. :) Каковыми и являются большинство agile контактов.

steel_ne 28 июля 2010 в 11:01 # +1

А без договора и любой agile превращается в рабство. «Вы копайте, считайте часы, а если мне не понравится, не заплачу. И еще должны будете».

Если все зафиксировать (и хотелки тоже) и правильно все оценить, то вполне можно работать по любой технологии. Какая разница заказчику, в какую песочницу играются программеры?

Jay_Di_Human 28 июля 2010 в 14:32 # 0

Стратегически мыслящему заказчику **важно**, в какую песочницу играются программеры. Особенно, если он настроен на долгосрочное сотрудничество.
И да, такие заказчики бывают. К сожалению, в основном зарубежные.

yshilyaev 28 июля 2010 в 15:34 # 0

С одной оговоркой — Если это профессиональный заказчик.
Наши (да и не только наши) заказчики от бизнеса ничего в программировании не понимают, поэтому красоты процесса оценить не могут. ;(

Jay_Di_Human 28 июля 2010 в 15:40 # 0

Мне повезло однажды работать с таким заказчиком. Но лишь однажды. Заказчик из США, Бостон. Возможно, это потому, что они сами технари и сами представляли бизнес.

yshilyaev 28 июля 2010 в 15:33 # +1

Ну таки я всегда так и объясняю, что заказчику до одного места по какой схеме работают разработчики. Главное, чтобы эта схема была близка к классической схеме: Товар — Деньги — Товар.

Если же процесс основан на схеме: «Деньги — Может быть товар — Ой мы тут не все сделали — Как это вы не это хотели? — Дайте денег, суки!» То, пардон, хоть какие названия говори.
Но по скраму процесс товар-деньги-товар… мне получалось организовывать. Заказчик не знал, что это Скрам. ;)

steel_ne 28 июля 2010 в 09:51 # 0

Та все правильно написано. При фиксировании цены главное не провтыкать с оценками. Это боль любой проектной технологии. Вот и разжевывается метод оценки с ключевой, на мой взгляд, фразой:

Это те деньги, которые мы согласны потерять.

acerv 28 июля 2010 в 09:59 # 0

про владельца продукта — ничего. а казалось бы, у него самая важная роль в общении с заказчиком и выяснении всех требований.

yshilyaev 28 июля 2010 в 10:21 # 0

Владелец продукта (Product Owner) — это роль заказчика, вообще-то. ;)
Если мы PO подменяем своим, то это скорее Proxy PO.

cyber_art 28 июля 2010 в 11:08 # +1

Product Owner — это человек ответственный за продукт. В идеальном случае — это Заказчик, но обычно он не готов им быть: занят другими делами, не хочет вникать в процесс, и т.п. Поэтому обычно PO становиться кто-либо из команды (аналитик, ПМ).

yshilyaev 28 июля 2010 в 13:24 # +1

Я об этом и говорил. :)
Такого называют Proxy Product Owner.

НЛО прилетело и опубликовало эту надпись здесь

НЛО прилетело и опубликовало эту надпись здесь

case 28 июля 2010 в 14:31 # 0

Спасибо, поправим.

JuliaTem 28 июля 2010 в 11:27 # +1

Вот объясните мне, человеку работавшему и по аджайл и без.
Если одно из основных достоинств аджайл — это то, что заказчику не надо заранее составлять чёткое тз, и он может изменять требования в процессе разработки. То о каком фиксированном наборе обязательств может идти речь?
Как только мы зафиксируем обём работ, это перестанет быть аджайлом :)

exaide 28 июля 2010 в 11:41 # +4

Красной нитью через статью проходит мысль:
«Аджайл с фиксированными обязательствами подобен аджайлу без фиксированных обязательств, только с фиксированными обязательствами» :)

case 28 июля 2010 в 14:34 # 0

Этот вопрос часто задается на конференциях и в кулуарах. Потому и выбрал именно эту статью.

yshilyaev 28 июля 2010 в 15:38 # +2

Ага. На одной конференции вышел один мой знакомый и рассказывал как они работают по фикс-прайсу по аджайлу. Очень убедительно рассказаывал. А потом когда началась сессия вопросов в него полетели: «А как вы боретесь с ростом требований?» «А как вам их оплачивают?»… На что докладчик только и отвечал: «У нас с ними выстроены доверительные отношения».
Через 5ть минут подвох вскрылся: у компании Заказчика и компании Разработчика одни и те же владельцы. Это две аффилированные компании. В такой ситуации можно доверять. ;)

case 28 июля 2010 в 16:12 # 0

Шикарно! :)

Popik 28 июля 2010 в 12:08 # 0

Я завидую вашему окладу в час

Hellbot 28 июля 2010 в 12:32 # 0

Не оклад, а стоимость. Слегка разные цифири.

dammer 28 июля 2010 в 22:02 # 0

Всё равно завидую)

karevn 29 июля 2010 в 02:10 # 0

Сдаётся мне, серьезное преимущество scrum — это чёткая сформулированность user story на момент когда она попадает в цепкие лапки разработчика. Потому что эту story уже на десять раз посмотрели во время покеров и плэннигов и еще раз пересмотрели непосредственно перед запуском итерации — это повышает вероятность вылавливания проблем постановки задач и резко повышает продуктивность за счет того, что программист садится и просто реализует, а не бегает к PO/PM каждые десять минут. А ТЗ неизбежно устаревает, в голове же всё не удержишь.

maxim_ge 17 августа 2010 в 13:57 # 0

На мой взгляд, самая большая проблема тут — **Прогнозируемая производительность команды**

Чтобы ее получить, необходимо выполнить несколько итераций. Хенрик Книберг в Scrum и XP: заметки с передовой рекомендует в качестве продолжительности спринта 3 недели. Т.е. пилотный проект, получается, будет длиться несколько месяцев. Хорошо, когда это лишь малая часть от общего бюджета, т.е. проект — весьма крупный.

С трудом представляю, как можно оценить производительность за одну итерацию — народ только-только начнет «въезжать» в проект.

Ну т.е. самая крупная проблема проектов с фиксированой ценой по сути осталась за рамками статьи. Ну что ж, зато еще раз прочитали изложение методологии SCRUM :)

4 Kotlin ♥ FP (17)

5 NGINX — Ускорение или Детектив для программиста «Оптимизация под Windows» (19)

Все публикации **Популярные хабы** **Компании**

Лучшее на Geektimes

1 Что не так с перезапуском «Кинопоиска» (126)

2 Астронавт Марк Уотни и РИТЭГ (4)

3 Парадокс Ферми (37)

4 Практический обзор Google AMP (Accelerated Mobile Pages) (8)

5 Astroneer — игра, которая позволит почувствовать себя «Марсианином» (24)

Все публикации **Популярные хабы** **Компании**

Вакансии на «Моём круге»

Senior Front-End developer
Казань • Полный рабочий день

Frontend разработчик
Полный рабочий день

Разработчик 1С-Битрикс (Bitrix)
Москва • Полный рабочий день

Аналитик-тестировщик на Фактор
Москва • Полный рабочий день

Backend-разработчик Java
Санкт-Петербург • Полный рабочий день

Frontend Developer
Берлин • Полный рабочий день

PHP-разработчик
Минск • Полный рабочий день

Node.JS developer
Минск • Полный рабочий день

Программист c++
Калининград • Полный рабочий день

HR manager
Калининград • Полный рабочий день

разместить вакансию все вакансии

Заказы на «Фрилансим»

Настройка VDS сервера Linux для форума Vbulletin
09.10.2015 • 2 отклика

MVC-скелет для сайта на базе Lua/Openresty
09.10.2015 • 0 откликов

Написать движок и сайт для подбора а\м в наличии и связи с дилерами
09.10.2015 • 6 откликов

Система автоматизации управления конфигурациями

09.10.2015 • 2 отклика

Нарезать готовый дизайн приложения (macos)
09.10.2015 • 2 отклика

Интеграция темы Wordpress
09.10.2015 • 3 отклика

Сетевое взаимодействие для дейтинг-приложения на Android
09.10.2015 • 2 отклика

Разработать коннектор на Node.js (Socket IO)
09.10.2015 • 2 отклика

Разработка ИС на PHP – Yii – Bootstrap
09.10.2015 • 3 отклика

Настройка кампании в AdWords (English)
08.10.2015 • 4 отклика

разместить заказ все заказы