# Опыт применения Agile методологий в разработке электроники.

*Алексей Омельянчук. Нач. КБ Рубикон ООО «СИГМА-ИС».*

Впервые я попробовал применить элементы Agile методологии, когда основным продуктом, разрабатываемым нашей компанией, была большая многопользовательская программная система для управления в реальном времени сложной аппаратурой.
Заказчиками являлись, в частности, МинАтом и МинОбороны, это делало необходимыми большое количество документации согласно ГОСТ-19, и в целом компания была склонна к излишнему документированию. Путь был непростой, начали с RUP, активно использовали ClearQuest, RequisitePro, но через год стало понятно, что 70 человек в департаменте явно недостаточно для строгого выполнения всех процедур. Настала эра MSF, но лично у меня не сложилось целостного понимания MSF невзирая на неоднократно прочитанные книжки духовных отцов MSF Cockburn'a, McConnell'a и конечно MS учебник по MSF. Общая идеология нацеленности на результат, на потребителя, итеративности, быстрого реагирования на изменения и тесного общения в моем понимании плохо стыковалась с множеством обязательных весьма формальных документов, особенно в части управления требованиями. Единственное, что мне сильно запало в душу пока изучал MSF и иногда помогало и тогда и в дальнейшем, это побочные технические приемы – Personas и Usage Scenarios (кто не в курсе, что это такое - см. ссылки [1]).
Конечно, еще лет 5 назад у нас появились книжки про XP, но никто не воспринимал всерьез столь экстремальный подход. И вот, с появлением SCRUM и многих популярных книг, объясняющих основы Agile методологий, я начал пробовать плавно сдвигать работу в эту сторону. Однако исторически структурированный не по проектам, а по специализации департамент разработки ПО довольно трудно принимал новшества.
Значительно восприимчивей оказался небольшой (12 человек разработчиков) департамент электроники. Для такого количества разработчиков, понятно, применение тяжеловесных методик типа RUP было невозможно, поэтому работа шла «как получится», в основном ориентируясь на привычные для конструкторов ГОСТ 2 и 15 серии, то есть итерации были запланированы изначально под названием литера «П» (техпредложение), литера «Э» (эскиз-проект), литера «О» (образец), литера «О1», литера «А». Как нетрудно догадаться, 99% времени разработка находилась где-то между «П» и «О», то есть вроде бы уже есть макеты, но представить их как образцы на испытания нельзя. Именно в этом департаменте как раз начиналась разработка новой серии устройств - микропроцессорный пульт для управления системой охраны и контроля доступа. Миниатюрное устройство с довольно большим объемом программирования. Инженеры-схемотехники и программисты с интересом восприняли идею раннего представления Заказчику образцов с частичной функциональностью – именно этот пункт был мной принят за первый и важнейший – фиксированные итерации с наличием в конце каждой продукта в состоянии, пригодном для показа.
Сам по себе факт нетривиальный. Даже программное обеспечение, как известно, очень трудно собрать воедино в продукт. А что говорить про электронику – постоянно оказывается, что прошивка микропроцессора существует для версии 5 (на тех платах работают программисты), аналоговые части отлажены на плате версии 7 (на которой исправлены некоторые ошибки схемотехники), а корпус рассчитан под плату версии 4 с некоторыми изменениями, которые предполагались в версии 5, но на самом деле реализованы были иначе. В-общем, первые разы так и показывали: на этом столе – как работает функционал, на этом столе (другая голая плата) – как она успешно выдерживает всякие перегрузки, а на третьем столе – как это удобно держать в руках, если кое-что подклеить скотчем, а куда-то воткнуть гвоздик чтобы открылась крышка.
Следующий необходимый момент SCRUM – наличие backlog – появился автоматически, когда впервые возник вопрос «ну ладно, сейчас что-то показали, а что будем показывать в следующий раз?» по сути, это был первый "planning meeting", неиспользованные идеи которого остались в качестве backlog на следующие итерации.
Результат внедрения первых элементов SCRUM оказался удивительным, хотя и не совсем тем, который хотелось. Руководство компании, наконец, увидело, что именно предположительно создается, и осознало, что идея разработки была совершенно провальной (на том сегменте рынка, на котором работала эта компания), весь проект был закрыт. Итог – ядро группы разработки вместе со мной ушло в другую компанию, которой нужен был именно такой продукт. Нам жалко было уже потраченных сил.
В новой компании разработки были еще более ориентированы на схемотехнику с минимальным количеством программирования. Например, пожарный датчик. Объем программы меньше килобайта, при этом схемотехника месяцами вылизывается до последней детали – каждый цент экономии при миллионных тиражах имеет значение.
Как же начать применять Agile управление разработками в таком случае?
Для начала я обнаружил естественный ритм, задаваемый производством и привычный многим разработчикам еще с советских времен – ежемесячные и ежеквартальные отчеты. Что ж, месячная итерация

ничем не хуже любой другой, если она удачно вписывается в менталитет всей организации. Главное – организовать ежемесячные «показы» (вскоре их стали называть «выставками достижений отдельского хозяйства») с максимальным размахом. Первые месяцы на них удавалось приводить менеджеров уровня генерального директора, и от самой организации и от Заказчиков и от смежников. Впоследствии ощущение новизны притупилось, и на месячных показах присутствовали в основном технические сотрудники, но ежеквартально обязательно показ проводился на самом высоком уровне.
Почему возникли особые «квартальные» показы. Дело в том, что при разработке электроники технически трудно за месяц изготовить новый вариант схемы. Закупка комплектации, изготовление платы и ее монтаж занимают около 2-х недель, что слишком много при месячной итерации. Получается, что разработчик 2 недели думает как улучшить предыдущую схему, а потом 2 недели идет изготовление новой, а разработчик тем временем продолжает работать на старой схеме, ему в голову приходят новые идеи, а потом приходит экземпляр очередной версии схемы, и все, что было сделано за последние две недели можно почти выбросить – теперь заново исследуются особенности новой схемы. Поэтому предполагается, что не каждый месяц будут делаться новые макеты с измененной схемотехникой, да и часто вполне возможно долго дорабатывать схему вручную, пока количество перемычек и подклеенных на весу деталей не превысит пределы разумного. В результате всего этого месячные итерации не всегда имеют в итоге наглядный, демонстрируемый прогресс. Внешне частенько даже выглядит, что вместо красивой платы в новом корпусе (как было месяц назад) демонстрируется прибор, склеенный из двух кусков разных плат, который в корпус не лезет в принципе, и работает только в присутствии автора, поддерживающего плохо припаянные детали.
Вторая проблема – многие свойства невозможно продемонстрировать при «показе». Так, испытания на климатическую устойчивость занимают несколько дней (даже в основной своей части). Испытания на максимальную дальность работы радиоканала – проводятся на полигоне далеко от Москвы, в безлюдной местности при минимальном уровне помех. Таким образом, показываются просто «протоколы испытаний», что, конечно, не слишком наглядно.
В результате всех этих факторов и сложилась двухуровневая система: месячные итерации, в результате которых нередко прогресс понятен только техническим специалистам, и ежеквартальные «релизы», к которым обязательно подчищаются все хвосты, изготавливаются новые версии электроники, как правило, изготавливаются новые прототипы корпусов – вобщем, на момент квартального «релиза» изделие имеет вполне товарный вид. Помимо самого изделия, которое можно подержать в руках, включить, попробовать его в работе (обычно один экземпляр собран «чтобы взять в руки», а второй заранее смонтирован на стенде со всем необходимым окружением «чтобы его попробовать»), на квартальных показах предъявляется список неустраненных замечаний, текущий список обнаруженных ошибок, оставшийся backlog, что в целом позволяет оценить текущее состояние и принять решение – когда хватит улучшать и пора ставить на конвейер. Сама постановка на серию займет еще 3-6 месяцев, за это время будут устранены почти все имеющиеся замечания и даже дополнительно реализовано что-то из бэклога, это тоже всем понятно и принимается во внимание, так что тот факт, что основные релизы жестко привязаны к календарному кварталу вовсе не означает, что разработка всегда занимает кратно кварталу. После принятия решения на квартальном показе изделие переходит уже в режим проектного управления – дальнейшие работы достаточно предсказуемы и весьма дорогостоящи, поэтому они планируются методами сетевых графиков, PERT/Gantt-диаграмм, вобщем дальше используется MS-Project. Да, на этом этапе еще могут быть работы типа «реализовать оставшиеся 2-3 функции», но эти работы невелики, и, как правило, не будет беды если они не попадут в изделия опытной партии, появятся чуть позже – не страшно. Разумеется, это возможно только если недоразработанные функции заведомо не приведут к серьезным изменениям конструкции и технологии производства.
Поскольку одним из главных параметров изделия была цена, как минимум к квартальным показам приходилось обновлять оценки себестоимости каждого изделия в серии. Так что всем было интересно. Большие директора разглядывали расчеты себестоимости и крутили в руках неработающие массогабаритные образцы, а технические специалисты от заказчика щелкали тумблерами на стенде и пытались придумать еще способы заставить устройство выйти из строя.

Пока мое изложение выглядело в розовом свете. Дальше начнутся сложности.
Главная сложность – привычная для электронщиков изолированность. По сравнению с ними можно сказать, что программисты – врожденные командные игроки, общительные, всегда работают группой. При разработке электроники узкая специализация куда серьезнее чем в программировании. У нас в отделе одновременно в работе были устройства на 5-ти разных семействах процессоров, некоторые с Гарвардской архитектурой, а некоторые с Неймановской, некоторые программируются на “C”, для некоторых уместен “C++”, а некоторые требуют ассемблер, и все существенно отличаются по особенностям встроенной периферии, да и средства разработки нередко у каждого процессора свои уникальные, «фирменные». История из практики: человек работает с Микрочип-процессорами, привык к их компилятору и отладчику ICD2, а тут ему вдруг предлагают подключиться к коллеге и помочь ему срочно доработать кусок кода на процессоре Renesas. Даже если найдется запасной адаптер для внутрисхемной отладки Renesas, его еще надо установить, проинсталлировать соответствующие программы (собственные и уникальные), освоить их…. Даже при использовании средств отладки от ведущих производителей – Keil, IAR (или при использовании

Eclipse+GNUCC), когда вроде бы затраты времени на освоение новой среды не так уж велики, очень много времени уходит на изучение самого процессора, его специфики. К слову, система прерываний PIC16 и Renesas M16 имеют только одно общее – и в том и в другом случае можно по некоторым событиям прервать исполнение основной программы. Человек, ранее незнакомый с новым процессором наверняка будет склонен использовать «универсальные» решения, например, реализовать обработку всех событий реального времени в одном прерывании от таймера (и не использовать никакие другие прерывания).

Еще хуже, что помимо специализации по номенклатуре электроники, разработчики специализируются по предметной области. Физика пожарного дымового датчика совершенно непохожа на физику работы пассивного инфракрасного датчика движения, хотя оба они инфракрасные. Особенности разработки счетчиков электроэнергии изложены в сотнях нормативных документов, одно ознакомление с которыми занимает месяцы.

Наконец, чисто механически разные рабочие места специализируются на разных работах. У одного инженера стоит на столе набор радиочастотных измерителей и генераторов, и чтобы передать их соседу надо на несколько дней остановить работу, перенести их все, помочь новому человеку освоить новую технику. У другого инженера в тумбочке стола смонтирован комплект нагрузок, так что передать их другому просто невозможно, проще на время поменяться рабочими местами. И это речь только о «поменяться», а если нужно подключить троих к одной проблеме, руководство крайне негативно рассматривает предложение закупить еще 2 комплекта необходимых узкоспециализированных приборов.

Наконец, например, персональные компьютеры с WindowsXP достаточно одинаково себя ведут. Есть множество организаций, контролирующих их совместимость. Хотя, как известно, и там не все так просто. А что уж говорить про экспериментальные устройства, собранные в единичных экземплярах? Их просто не может быть двух одинаковых. Когда два разработчика работают над одним и тем же проектом, они раздельно экспериментируют над своими экземплярами, и очень быстро они становятся несколько отличными. Программный код достаточно легко (по крайней мере, есть много книжек на тему как это сделать) объединить в единый TRANK в системе управления версиями, а сделать то же самое с железом невероятно трудно. Нет никаких способов автоматизировать отслеживание внесения изменений в железо. Только самодисциплина, только контроль со стороны руководства и коллег за наличием исправлений в документации.

Традиционное построение проектной группы для небольших разработок было таково: один конструктор-механик (разрабатывает пластик), один схемотехник (основная разработка электроники), иногда один программист (подключается по мере необходимости, если требуется высокоуровневое программирование). Понятно, что внутри таких проектных групп невозможно общение специалистов, каждый занимается своим делом и ему просто не с кем посоветоваться даже когда хочется. Основной механизм передачи опыта – курилка. А что делать некурящим?

Предложенное решение было таково: группу сходных проектов (например, все датчики, даже построенные на разных процессорах и измеряющие разные физические величины) принудительно объединили в группу, и далее принимали различные меры по улучшению общения между коллегами. Как крайняя мера – просьба временно переключиться и помочь коллеге в работе над другим проектом. Но это действительно крайняя мера, когда всем очевидно, что по каким-то причинам проект запаздывает.

Основной путь организовать общение – нечто среднее между «парным программированием» и каноническим «code review». Еженедельно поочередно устраиваются обсуждения проблем проектов в рамках группы. Как правило, совещания по специализации – или посвященные конструкции и дизайну, на них присутствуют конструкторы-механики, но обычно отсутствуют программисты. Или совещания, посвященные схемно-алгоритмическим проблемам, на них обычно отсутствуют механики. Подчеркну, совещания именно регулярные, предназначенные для взаимного информирования и оперативной помощи, до того как проблема назрела. Ведь это так приятно – надавать соседу советов, особенно если он об этом не просил. Именно вследствие таких обсуждений иногда и возникали просьбы к тому или иному сотруднику – отложить свою работу и временно помочь соседу, например, провести испытания в климатической камере, или собрать стенд для проверки. Как правило, целевая задача при подключении «соседа» ставилась конкретная и достаточно изолированная, чтобы не было проблем специализации, описанных выше. Для проведения разовой побочной работы никто не подготовлен, поэтому сосед может сделать ее столь же успешно, как и основной разработчик. Однако результат таких «подключений» намного шире. «Помощнику» неизбежно приходится глубже разобраться в чужих схемных решениях, нежели он делал на совещании, и его советы становятся куда более конкретными и полезными. Такая задача не на всю итерацию, обычно всего на несколько дней, максимум на пару недель. Таким образом, возник как бы суперпроект «все датчики», по которому на итерацию планировалось чуть улучшить один, переделать другой, проверить некоторые идеи на третьем. Заодно такое формирование суперпроектной группы психологически облегчало проведение месячных показов. Если по одному датчику демонстрировался лишь протокол испытаний, то по другому – новый прототип корпуса, и вместе было и полезно и интересно и никому не стыдно.

Помимо еженедельных совещаний «без повода», проводились и обсуждения «по поводу», совсем похожие на публичные “code review”. Например, когда разработчик заявляет, что в принципе он все сделал. Так, осталось только мелочи доработать. Или наоборот, когда он говорит, что все, что делали, никуда не годится,

надо начинать заново, и он уже знает как. Тут тоже помогло влияние производства, параллельно осваивавшего сертификацию на автомобильные стандарты управления качеством. Кстати, по сравнению с этим стандартами (обязательными для локальных поставщиков Renault, Ford и других) обычная сертификация ISO9000 – детский лепет. Так вот, одно из популярных, произошедших из системы 6-сигма, мероприятий на фабрике называлось FMEA (Fault Mode and Effect Analysis). Мы в подразделении разработки подхватили термин, и проводили такие совещания по каждому достойному поводу. В отличие от еженедельных, тут собирается не проектная группа, а все умудренные опытом аксакалы, из всех групп, и помогают предсказать – какие проблемы следует ожидать от предложенного решения, как можно будет проверить, что проблема миновала стороной, и что примерно делать, если не миновала. Например, специалисты из группы электросчетчиков всегда напоминали – «а что будет, если включат не в 220 а в 380?» Правда результатом обычно были не меры по дополнительной защите входа питания, а внесение в паспорт явного указания, что производитель не несет ответственности за повреждения в результате превышения напряжения питания. А кроме анекдотов, действительно, специалисты, съевшие несколько собак на слаботочных датчиках сразу указывают на проблемы, которые могут случиться при медленном снижении питания. Специалисты, разрабатывающие контрольные приборы сразу указывают на огрехи, за которые они бы растерзали всех разработчиков датчиков и уж от своих-то они меньше всего такой подлости ожидали (нелинейная характеристика порой непредсказуемо сказывается на работе ППК в широком диапазоне напряжений питания и при максимальной длине подключенных проводов).
Большой проблемой явилось то, что ежедневный SCRUM-meeting практически не имеет смысла в группе, объединяющей несколько проектов. Люди не настолько подробно знакомы с работой соседей, чтобы это было хоть сколько-то интересно. Кроме того, скорость изменений состояния при разработке электроники объективно ниже, чем при программировании – типичный ответ «перепаиваю на новую схему» может звучать несколько дней подряд и это не является признаком плохой работы. Отказ от ежедневных митингов в многопроектных группах привел и к противодействию им в группах, которые работали по одному проекту. Тем более, что и в формально одном проекте на самом деле было много весьма разных субпроектов. Из-за довольно узкой специализации работа одного схемотехника по периферийному модулю на PICe неинтересна тем, кто программирует ARM7 на центральном контроллере, и наоборот. Я до сих пор в сильном сомнении – надо ли было их проводить, насколько часто, и всем вместе или разбив проектную группу на субпроекты (и отдельно схемотехники, отдельно программисты). В результате проектные митинги собирались в многопроектных группах раз в неделю и частично выполняли роль SCRUM-meeting'ов. В однопроектной группе митинги собирались один-два раза в неделю, то есть не очень регулярно.
Чтобы компенсировать отсутствие ежедневных докладов перед лицом товарищей, приходилось мне лично осуществлять ежедневные обходы и разговоры, хотя доклад начальнику куда как уступает по эффективности докладу товарищам. Надеюсь, в будущем смогу найти форму организации ежедневного общения коллег, достаточно интересную для них самих, чтобы такие митинги происходили не из-под палки.

Что нам не удалось толком реализовать, так это планирование в настоящем стиле SCRUM и количественное измерение скорости. Быть может, все еще впереди. Первоначально мы выбрали слишком крупные задачи, поэтому они часто не умещались в итерацию, и никакая "burndown chart" была невозможна. Мы использовали вместо нее плакат, на котором просто вычеркивали пункты по мере их реализации. Потом постепенно сделали задачи мельче. Планирование проводили силами команд, но без количественных оценок отдельных функций и задач, по следующей известной методике: берем упорядоченный по важности backlog, тыкаем куда-то пальцем и спрашиваем у группы: «вот по сюда успеем за следующий месяц?». Ответ почти наверняка «да не, ну что вы, куда там…». Тогда двигаем палец вверх и продолжаем «ну а вот до сих, успеем ?». За несколько итераций, руководствуясь тональностью общего ответа, палец устанавливается на какой-то строке. Теперь быстро ксерим этот список и даем каждому, чтобы внимательно посмотреть с карандашом. Через пять минут всех спрашиваем мнение – какие могут быть проблемы и с какими пунктами плана. Еще раз возвращаемся и еще немного корректируем список функций на следующую итерацию. Все. Да, это не "planning poker", и нет никаких впрок расписанных оценок для пунктов в backlog'е, но ведь не все же сразу? Многие инженеры пришли с фабрики, они там привыкли работать от свистка до звонка. Кроме того, чтобы применять обезличенные «скорость» и «трудность» надо, чтобы хоть в какой-то мере в практику вошла взаимозаменяемость, чтобы люди в группе не зацикливались на своей и только своей привычной специализации.
По известному Нокиа-тест ([2]) значительным недостатком нашей организации был и тот факт, что нам не удалось обеспечить отсутствие отвлечений во время итерации. Как-никак, а старые продукты стоят на серии, выпускаются тысячами ежедневно, и когда на конвейере возникает проблема, приходится все бросать и решать ее немедленно. Мы сочли, что это не так уж и страшно, и честно при планировании напоминали членам группы: «а вы учли, что в Мордовии наверняка опять будут проблемы с изделием X, а в Томилино – с изделием Y, которое после двухмесячного перерыва снова поставят на серию?». В конце-то концов, считается же нормальным планировать потери времени на болезни и общеконторские совещания, так почему бы так же и не планировать отвлечения на другие внеплановые работы, хотя бы «в среднем, по опыту». Вот чего мы точно никогда не допускали, так это того, что задачи по проекту вдруг меняются

посреди итерации. За год я не припомню ни одного случая, когда внезапные изменения приоритетов отразились бы в текущей итерации. Правда, бывали случаи, когда задача вроде бы стоит, а потом оказывается, что она просто плохо сформулирована, поэтому разработчик вместо нее сделал две другие.

Помимо попыток внедрения чего-то похожего на SCRUM, мы провели довольно интересные эксперименты в способе задания требований. Поскольку речь шла не о разработке принципиально новых, не имеющих аналогов устройств, то требования ВСЕГДА задавались «относительно прообраза». Даже если у нового устройства имелся прямой прообраз среди своей продукции, всегда указывался маркетинговый конкурент и основные маркетинговые идеи – по каким параметрам новый продукт может уступать (и насколько), а по каким он должен превосходить конкурента. Главный параметр, повторюсь, всегда была цена (себестоимость), его обязательно задавали явно, с высокой точностью. Остальные параметры нередко указывались также ссылкой на конкурентов – дескать «по остальным параметрам должен быть не хуже чем средним в следующей группе изделий конкурентов… Это очень корректная формулировка. Новое выводимое на рынок устройство, как правило, должно не быть явно хуже основных конкурентов, и хотя бы одного конкурента явно бить по какому-то параметру, который маркетинговые специалисты сочли сегодня самым важным.
Должен отметить, что такое задание цели (vision проекта) весьма компактно и уже отвечает почти на все вопросы, даже без написания подробного техзадания. Разработчики могут сами по мере необходимости придумывать способы сравнения всех параметров с конкурентами и проводить их сравнительные испытания. При этом, кстати, нередко выясняется, что разрекламированный продукт фактически уступает многим середнячкам, и ориентироваться надо вовсе не на него, как сначала казалось.
Единственное, что очень трудно сравнивать разработчикам, и обязательно должно делаться заказчиком (пользователем) – это сравнение “usability” разных устройств, и особенно тонкий баланс “usability” и цены – как правило, выскребание последних центов в себестоимости делается лишь за счет существенных уступок в части удобства пользователя. Когда остановиться, чем можно пожертвовать в борьбе за цену, а чем не стоит – вопрос, на который может ответить только заказчик, и только непосредственно попробовав руками разные варианты. Никакие предварительные обсуждения не стоят и одной минуты показа в конце первой же итерации.
В этом отношении нам повезло, был налажен тесный контакт с «продажниками», организаторами дистрибьюторской сети, хорошо знающими не только «что люди говорят», но и «на основе чего» они действительно делают выбор при покупке.
Соответственно, “user stories” реально надо было писать только для новых функций, или функций, в которых мы должны были получить преимущество над конкурентами. В остальном же фактически вместо “user stories” использовались ссылки на разделы руководства по эксплуатации какого-нибудь конкурента.

Перечитав получившуюся статью, должен признать, что ситуация в управлении разработками получилась несколько приукрашена. Не все, что описано равно эффективно работало во всех проектах. Главное, большая часть сказанного относится к концу прошедшего года, а не к его началу, когда не то что backlog’а или vision’а в проектах не было, не было даже общепризнанного списка проектов, которым отдел занимался. Тем не менее, надеюсь, кому-то будет полезен этот опыт расширения SCRUM на смежную область. Я встречал утверждения, что Embedded Agile – это абсолютно то же самое, что и обычный Agile процесс на обычных компьютерах. Может быть это и так в приложении к коммуникатор и смартфонам, которые сейчас превосходят по всем параметрам рядовые компьютера десятилетней давности. Но в приложении к разработке микроконтроллерных устройств действительно микро-размера, я точно знаю, есть много особенностей. Частично мне, похоже, удалось их преодолеть, или даже обратить на пользу, ну а частично еще все впереди. Если у вас есть вопросы, мнения или замечания, не стесняйтесь писать на мэйл автора.

Литература:
[1]
http://en.wikipedia.org/wiki/Personas
http://www.infodesign.com.au/usabilityresources/design/scenarios.asp
http://www.foruse.com
можно нагуглить еще много-много интересных статей…
[2]
http://jeffsutherland.com/scrum/scrumbutttest.pdf