Ag;)eChecklist

Очень краткое описание практик гибкой разработки

Асхат Уразбаев, Никита Филиппов

Версия 2.0. 17.11.2011

О ЧЕКЛИСТЕ

- ✓ Чеклист представляет собой оч.кратк. описание практик гибкой разработки.
- ✓ Чеклист носит рекомендательный характер.
- ✓ Вы используете его на свой страх и риск.
- ✓ Ни одна из практик не является обязательной.
- ✓ Вы можете изменять практики в соответствии с проектной необходимостью и собственным представлениям.
- ✓ При применении подключайте голову.

Кодифицированного или даже просто общепринятого описания Agile не существует. Данный текст – компиляция различных литературных источников, замешанная на опыте и знаниях специалистов ScrumTrek.

ОГЛАВЛЕНИЕ

КОМАНДА

Команда (team)

В методологии Scrum команда является самоорганизующейся и самоуправляемой. Команда берет на себя обязательства по выполнению объема работ на спринт перед Владельцем продукта. Работа команды оценивается как работа единой группы.

Команда в Scrum кроссфункциональна. В нее входят люди с дополняющими навыками – разработчики, аналитики, тестировщики, дизайнеры. Нет заранее определенных и поделенных ролей и специализаций в команде, ограничивающих область действий членов команды. Команда состоит из инженеров, которые вносят свой вклад в общий успех проекта в соответствии со своими способностями и проектной необходимостью. Команда самоорганизуется для выполнения конкретных задач в проекте, что позволяет ей гибко реагировать на любые возможные изменения.

Размер команды ограничивается размером группы людей, способных эффективно взаимодействовать «лицом к лицу». Типичный размер команды - 7 плюс минус 2. По нашему опыту, чем ближе к нижней границе, тем лучше.

ЧЕКЛИСТ

- ✓ Команда сидит в одном месте (colocaltion)
- ✓ Члены команды чувствуют ответственность по отношению друг к другу
- ✓ Команда кроссфункциональна – нет закрепления отдельных ролей за членами команды
- ✓ Члены команды работают совместно над завершением фич более высокого приоритета в первую очередь
- ✓ Члены команды признают наличие проблем и просят друг друга о помощи
- ✓ Члены команды помогают друг другу

СКРАМ-МАСТЕР

Скрам-мастер (ScrumMaster)

Скрам-мастер - член команды. Основная его задача — помочь команде стать самоуправляемой и самоорганизующейся. Он следит за тем, чтобы команда выполняла принятые ею решения, за соблюдением практик и отвечает за решение проблем, обнаруженных командой и находящихся вне ее компетенции.

Скрам-мастер не раздает задачи членам команды. Скрам-мастер - один из членов команды. Он может быть разработчиком, тестировщиком, аналитиком и т.д.

Скрам-мастер проводит командные митинги (стендап, планирование, демонстрацию и ретроспективу).

Скрам-мастер также отвечает за удаление всех внешних препятствий, мешающих команде. Например, если команде нужен новый сервер, то именно скрам-мастер вступит в бой с бюрократическими силами компании. На практике часто бывает, что скрам-мастер просто обозначает соответствующую проблему менеджеру и уже менеджер занимается ее решением.

Скрам-мастер следит за климатом внутри команды и старается создать атмосферу доверия. Это ни в коем случае не означает, что скрам-мастер «гасит» все конфликты между членами команды. Напротив, конфликт должен быть вытащен на обсуждение как можно быстрее и конструктивно решен.

По мере того, как команда становится самоорганизующейся, роль скрам-мастера уменьшается.

ЧЕКЛИСТ

- ✓ Скрам-мастер есть
- ✓ Скрам-мастер следит за выполнением командой практик Scrum
- ✓ Скрам-мастер член команды (а не менеджер и не владелец продукта)
- ✓ Скрам-мастер помогает устранять внешние препятствия
- ✓ Скрам-мастер следит за выполнением командой принятых ею решений

ВЛАДЕЛЕЦ ПРОДУКТА (PO)

Владелец продукта (Product Owner), менеджер продукта (ProductManager)

Владелец продукта (Product Owner) – это человек, отвечающий за разработку продукта. Это может быть менеджер продукта для продуктовой разработки, менеджер проекта для внутренней разработки и представитель заказчика для заказной разработки. Владелец продукта – это единая точка принятия окончательных решений для команды в проекте, именно поэтому это всегда один человек, а не группа или комитет.

Это позволяет избежать проблемы множественности заказчиков. Владелец продукта формулирует эффективный путь достижения целей, и доносит его до каждого, кто в том нуждается. Владелец продукта должен четко понимать, какие фичи должны быть сделаны, и каковы их приоритеты.

Владелец продукта ставит задачи всей команде, но он не вправе ставить задачи конкретному члену проектной команды.

ЧЕКЛИСТ

- ✓ У команды только один владелец продукта
- ✓ Владелец продукта устанавливает приоритеты элементов баклога для своей команды
- ✓ Владелец продукта достаточно хорошо разбирается в продукте и предметной области для того, чтобы правильно расставить приоритеты
- ✓ Владелец продукта отвечает за формирование концепции продукта (ProductVision)
- ✓ Владелец продукта управляет ожиданиями заказчиков и всех заинтересованных лиц
- ✓ Владелец продукта отвечает за предоставление требований команде
- ✓ Владелец продукта отвечает за приемку результатов в конце каждой итерации

ИТЕРАЦИЯ

Итерация (Iteration), Спринт (Sprint)

В течение одной итерации проектная команда общается с заказчиками, анализирует, пробует, разрабатывает и тестирует код. В конце каждой итерации демонстрируется полностью доделанная за итерацию функциональность. Заказчики смотрят на результаты работы. Все предложения по улучшению планируются на последующие итерации. Внутри итерации заказчики стараются воздерживаться от изменения требований.

Такими короткими шагами раз за разом мы приближаемся к желаемому результату, корректируя по ходу направление нашего движения.

Длина итерации — от 1 до 4 недель. Типичная длительность итерации – 2 недели.

ЧЕКЛИСТ

- ✓ В конце итерации есть готовый инкремент продукта
- ✓ Фичи, которые начинаются в эту итерацию, в ней же и доделываются
- ✓ Разработка фичи включает тестирование и исправление найденных багов
- ✓ Команда соблюдает приоритеты фич, установленные Владельцем продукта
- ✓ В большинстве случаев команда делает то, что было запланировано. Иногда чуть больше, иногда чуть меньше
- ✓ Команда сообщает владельцу продукта, когда отстает от плана итерации
- ✓ В случае отставания от плана команда предпринимает корректирующие действия
- ✓ Для каждой фичи команда знает, каким образом и от кого получить дополнительную информацию в случае необходимости
- ✓ Проблемы обнаруживаются быстро и обсуждаются командой сразу же
- ✓ Длина итерации не меняется после каждой итерации
- ✓ Вся незапланированная работа учитывается
- ✓ Не более одного дня задержки между итерациями
- ✓ Заинтересованные лица знают о том, что работа ведется итерациями

БАКЛОГ ПРОДУКТА

Баклог продукта (Product Backlog)

Баклог продукта – это приоритезированный список имеющихся на данный момент бизнес-требований и технических требований к системе.

Баклог продукта может включать фичи (пользовательские истории, запросы на изменения, сценарии использования и т.д.), технические истории, баги и технический долг, открытые вопросы. Баклог продукта также включает задачи, важные для команды, например «провести тренинг», «добавить памяти на машины».

Баклог продукта постоянно пересматривается и дополняется – в него включаются новые требования, удаляются ненужные, пересматриваются приоритеты. За баклог продукта отвечает Владелец продукта. Он работает совместно с командой для того, чтобы получить приближенную оценку на выполнение фич из баклога продукта.

ЧЕКЛИСТ

- ✓ Владелец продукта управляет и координирует баклог продукта
- ✓ Баклог продукта содержит фичи (а не технические задачи)
- ✓ Баклог продукта виден каждому
- ✓ Баклог продукта обновляется перед планированием итерации
- ✓ Владелец продукта понимает все фичи из баклога

ПЛАНИРОВАНИЕ ИТЕРАЦИИ

Планирование итерации (iteration planning), планирование спринта (sprint planning)

Встреча, на которой команда и владелец продукта планируют итерацию. Владелец продукта ставит цели итерации и представляет фичи, команда декомпозирует фичи на технические задачи и совместно оценивает их. Итоговый план таймбоксится, то есть в него включаются только те фичи, которые команда планирует успеть сделать в итерации.

Для таймбоксинга итерации используется Скорость (Velocity) команды.

Декомпозиция задачи должна быть достаточно детальной. Для двухнедельной итерации рекомендованная длительность технической задачи – порядка дня (не более двух дней).

ЧЕКЛИСТ

- ✓ Владелец продукта и члены команды участвуют лично
- ✓ Результат - план итерации
- ✓ Все члены команды согласны с планом
- ✓ Все члены команды согласны с тем, что план может быть выполнен за итерацию
- ✓ Каждая фича имеет приоритет внутри итерации
- ✓ Все технические задачи имеют оценки
- ✓ Все фичи имеют оценки
- ✓ Планирование начинается и заканчивается вовремя
- ✓ На планировании итерации технические задачи не назначаются на людей

СКОРОСТЬ КОМАНДЫ

Скорость команды (Velocity)

Скорость команды рассчитывается, как сумма оценок фич, которые были сделаны командой в итерацию.

Как правило, это значение со временем становится более или менее стабильным и от итерации к итерации меняется незначительно. Знание этой величины и оценок фич позволяет таймбоксить итерацию и осуществлять долгосрочное планирование.

ЧЕКЛИСТ

- ✓ Скорость рассчитывается после каждой итерации
- ✓ В расчете скорости учитываются только те фичи, которые удовлетворяют критерию готовности
- ✓ Скорость используется для долгосрочного планирования

ДОЛГОСРОЧНОЕ ПЛАНИРОВАНИЕ

Долгосрочное планирование (long-term planning), Планирование релиза (release planning)

Команда совместно с владельцем продукта осуществляет долгосрочное планирование. Команда оценивает фичи из баклога.

Фичи оцениваются в условных единицах (story points) либо идеальных человеко-днях (ideal man-days). Для оценки часто используется практика «Покер планирования» (Planning poker).

Владелец продукта формирует долгосрочный план, часто называемый «планом релиза» (release plan) на основании оценок команды и знания скорости команды. В долгосрочном плане указано, какие фичи в какой итерации планируется сделать.

ЧЕКЛИСТ

- ✓ Владелец продукта отвечает на вопросы команды во время оценки
- ✓ Только команда может оценивать
- ✓ Все члены команды участвуют в оценке
- ✓ Фичи высокого приоритета небольшие, так что несколько фич может быть сделано за одну итерацию
- ✓ Долгосрочный план корректируется каждую итерацию

ПЛАН ИТЕРАЦИИ

План итерации (iteration plan), Баклог спринта (sprint backlog)

План итерации – набор фич и задач, на которые фичи декомпозируются, которые команда собирается выполнить в итерацию.

План итерации поддерживается командой в актуальном состоянии в течение итерации. Для ведения плана итерации удобно использовать Доску Задач (Task Board)

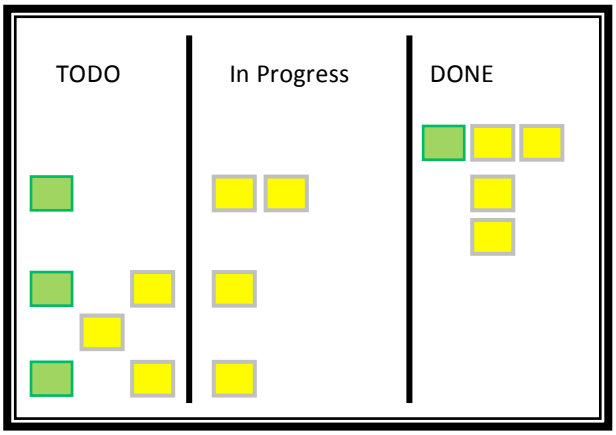
ЧЕКЛИСТ

- ✓ Все видят план итерации
- ✓ Для каждой фичи из баклога указаны приемочные тесты
- ✓ Для каждой фичи из баклога указан сценарий демонстрации
- ✓ Каждая фича на плане декомпозируется в технические задачи
- ✓ Оценки задач при необходимости корректируются членами команды в течение итерации
- ✓ Члены команды регулярно актуализируют план
- ✓ Все члены команды могут самостоятельно поменять план итерации: добавлять и удалять задачи, изменять статусы и оценки

ДОСКА ЗАДАЧ

Доска задач (TaskBoard)

Доска задач – физическая доска, висящая в комнате, где находится команда. Пространство доски поделено на вертикальные полосы.



На доске левая полоса (To Do) предназначена для задач, за которые еще никто не брался. Прикрепляем карточки на эту полосу так, чтобы карточки с задачами находились рядом с соответствующими карточками фич. Вторая полоса (In Progress) предназначена для задач, которые находятся в работе. Разработчик берет задачу из TODO, перевешивает в In progress и подписывает в тот момент, когда берет задачу в разработку.

Как только задача сделана, разработчик перемещает карточку дальше, в третью полосу (Done). Теперь он может взяться за следующую задачу. Он берет карточку из первой полосы, подписывает и перемещает на вторую.

Таким образом, задачи постепенно передвигаются из первой в последнюю полосу. К концу итерации туда должны переместится все задачи. Количество полос может быть больше, они, фактически, отражают жизненный цикл задачи или фичи.

ЧЕКЛИСТ

- ✓ Доска задач висит в комнате команды
- ✓ Стендапы проводятся возле доски задач
- ✓ По итогам стендапа доска задач актуализируется
- ✓ Каждая задача в InProgress имеет ответственного

СТЕНДАП

Стендап (Standup meeting), Скрам (Daily Scrum Meeting)

Короткая ежедневная встреча, предназначенная для синхронизации работы команды.

Проводит митинг скрам-мастер. Он спрашивает по кругу всех членов команды, задавая три вопроса

1. Что сделано вчера?
2. Что будет сделано сегодня?
3. С какими проблемами столкнулся?

Каждый член команды отвечает на эти вопросы.

Достаточно часто завязываются обсуждения по тем или иным вопросам. Пока два человека спорят друг с другом, все остальные скучают и начинают отвлекаться. Поэтому задача скрам-мастера состоит в том, чтобы остановить обсуждение и вынести его за пределы стендапа. Даже если возникшая дискуссия затрагивает всех участников, лучше ее остановить, закончить стендап и затем снова продолжить обсуждение.

ЧЕКЛИСТ

- ✓ Проводится в одно и то же время и в одном и том же месте
- ✓ Длительность не более 15 минут
- ✓ Начинается и заканчивается вовремя
- ✓ Все члены команды участвуют
- ✓ Все члены команды отвечают на три вопроса
- ✓ Стендап не прерывается
- ✓ Члены команды сами выбирают задачи на стендапе
- ✓ Скрам-мастер не раздает задачи
- ✓ Члены команды обращаются друг к другу, а не отчитываются перед скрам-мастером или менеджером

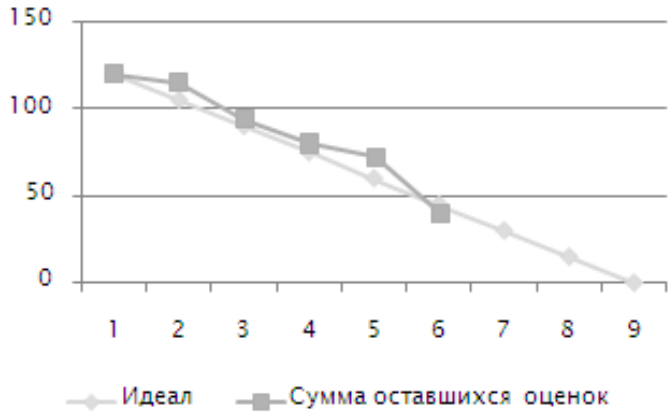
ДИАГРАММА СГОРАНИЯ

Диаграмма сгорания (burndownchart)

Для отслеживания прогресса в спринте используется диаграмма сгорания (burndownchart). По оси ***x*** откладываются дни итерации, по оси ***y*** –сумма оценок **несделанных** задач.

В начале итерации эта сумма равна сумме всех оценок всех задач. Каждый день в одно и то же время (например, перед стендапом) скрам-мастер рассчитывает новую точку и ставит ее на диаграмме.

На диаграмму также наносят прямую линию, которая демонстрирует «идеальное» сгорание работы. Если график сгорания выше, чем идеальная кривая, значит команда отстает, если ниже – опережает график.



ЧЕКЛИСТ

- ✓ Диаграмма видна всем
- ✓ Диаграмма предназначена для команды, а не менеджеров
- ✓ Диаграмма актуализируется каждый день
- ✓ Команда предпринимает действия, когда значения слишком велики или малы

ДЕМО

Демо (Demo), Ревью спринта (Sprint Review), показ

Главная цель демонстрации – получить обратную связь и наладить общение со всеми заинтересованными лицами. Практика позволяет задать ритм такому общению.

Демонстрация проводится в конце каждой итерации. Команда показывает результаты своей работы, например, последовательно показывая сделанные фичи.

Не имеет смысла показывать и обсуждать на демо вопросы, не касающиеся приглашенных заинтересованных лиц.

ЧЕКЛИСТ

- ✓ Проводится после каждой итерации
- ✓ Демонстрируется работающая система (не презентации и не написанные классы)
- ✓ Только сделанные фичи демонстрируются
- ✓ Все заинтересованные лица приглашаются на демо
- ✓ Команда получает обратную связь от заинтересованных лиц
- ✓ Заинтересованные лица получают обратную связь
- ✓ Владелец Продукта корректирует баклог продукта в соответствии с пожеланиями заинтересованных лиц

РЕТРОСПЕКТИВА[1]

Ретроспектива (retrospective)

Ретроспектива предназначена для улучшения процесса разработки. Команда собирается вместе для того, чтобы проанализировать прошедший спринт и решить, что нужно улучшить и как сделать работу еще более эффективной. Процесс становится по-настоящему адаптивным и постоянно улучшается самой командой.

ЧЕКЛИСТ

- ✓ Ретроспектива проводится в конце итерации
- ✓ Все члены команды и владелец продукта участвуют
- ✓ Все члены команды и владелец продукта имеют равное право голоса
- ✓ Ретроспектива не является встречей по поиску виноватых
- ✓ Результатами ретроспективы является план улучшения процесса
- ✓ План улучшения процесса претворяется в жизнь
- ✓ Все участники высказываются
- ✓ Нет «лишних» гостей

[1] Перевод

КРИТЕРИИ ГОТОВНОСТИ (DOD)

Критерии готовности (Definition of Done, DoD)

Критерии готовности – согласованное описание определения готовности фичи в виде чеклиста.

Можно ли считать фичу сделанной? Такой вопрос часто возникает в конце итерации и иногда является источником споров команды, владельца продукта и прочих заинтересованных лиц.

Команда и владелец продукта договариваются о критериях готовности фичи. Они формулируются сразу для всех фич и могут доопределяться для каждой конкретной фичи.

Пример. Критерии готовности могут быть сформулированы так:
- ✓ Фича создана и работает на stage.
- ✓ Проведено тестирование,
- ✓ Отсутствуют критические и мажорные дефекты,
- ✓ Написаны автоматизированные приемочные тесты

И так далее.

Иногда критерий готовности также определяется для отдельной технической задачи или требования.

ЧЕКЛИСТ

- ✓ Критерии готовности сформулированы командой и владельцем продукта
- ✓ Команда соблюдает критерии готовности
- ✓ Все члены команды и владелец продукта знают критерии готовности
- ✓ Критерии готовности включает в себя тестирование
- ✓ Команда не зависит от других людей для того, чтобы обеспечить выполнение критериев готовности
- ✓ Критерии готовности регулярно корректируются

РЕКОМЕНДОВАННОЕ ЧТЕНИЕ

- ✓ *Scrum и XP: заметки с передовой*, Хенрик Книберг (http://scrum.org.ua/scrum-i-xp-zametki-s-peredovoj/) (на русском)
- ✓ *Статьи AgileRussia* http://www.agilerussia.ru/
- ✓ *Блог ScrumTrek* http://blog.scrumtrek.ru/
- ✓ *Сообщество AgileRussia в facebook* **http://tinyurl.com/agilerussiafb**
- ✓ *Scrum. Гибкая разработка ПО*, Майк Кон
- ✓ *Agile Estimating and Planning* by Mike Cohn
- ✓ *User Stories Applied* by Mike Cohn
- ✓ *Agile Software Development* by Alister Cockburn
- ✓ *Agile Retrospectives* by Esther Derby
- ✓ *Agile Testing* by Lisa Crispin
- ✓ *Agile Documentation* by Andreas Rueping

БЛАГОДАРНОСТИ

- ✓ Алексею Трошину, Максиму Дорофееву, Владу Балину за ревью чеклиста!
- ✓ Сообществу AgileRussia за вдохновение и поддержку!