Лекция 10

MSF

Microsoft Solutions Framework

Литература

- 1. Microsoft Solutions Framework, Process Model, Version 3.1. 2002.
- http://www.microsoft.com/msf (Русский перевод: Microsoft Solutions Framework.
- Модель процессов, версия 3.1. 41 с.)
- 2. Microsoft Solutions Framework, Team Model, Version 3.1. 2002.
- http://www.microsoft.com/msf (Русский перевод: Microsoft Solutions Framework.
- Модель команды, версия 3.1.)
- 3. Microsoft Solutions Framework, Project Management, Version 3.1. 2002.
- http://www.microsoft.com/msf (Русский перевод: Microsoft Solutions Framework.
- Упрвление проектами, версия 3.1.).
- 1. Manifesto for Agile Software Development. http://agilemanifesto.org/.
- 2. M.Fowler. The New Methodology. 2003. http://www.martinfowler.com/articles/
- newMethodology.html. (Русский перевод: М.Фаулер. Новые методологии
- программирования. 2001. http://www.maxkir.com/ sd/newmethRUS.html).
- 3. K.Beck, C. Andres. Extreme Programming Explained. 2004. Paperback. 118 p.
- 4. М.Борисов. Scrum: гибкое управление разработкой. 30/05/2007 №04.
- http://www.osp.ru/os/2007/04/4220063/.
- 5. Wikipedia http://en.wikipedia.org/wiki/Scrum_(development). Создание Project Backlog Sprint lanning Meeting Sprint Sprint Review Meeting

История и текущий статус

- В 90-х годах компания Microsoft, стремясь достичь максимальной отдачи от
- реализации заказных IT-решений и в целях улучшения работы с субподрядчиками
- обобщила свой опыт по разработке, внедрению, сопровождению и консалтингу ПО,
- создав методологию MSF.
- В 2002 году вышла версия MSF 3.1, состоявшая из пяти документов-руководств:
- □ модель процессов (process model),
- □ модель команды (team model),
- □ модель управления проектами (project management),
- □ дисциплина управления рисками (risk management),
- □ управление подготовкой (readiness management).

- ***IT решение – понимается как скоординированная поставка набора элементов***
- (таких как программные средства, документация, обучение и сопровождение),
- необходимых для удовлетворения бизнес-потребности конкретного заказчика. *Причем*
- *под его разработкой понималась создание ПО, обучение пользователей и полная передача продукта команде сопровождения.*
- *Задача наладки полноценного сопровождения IT-* решения - важная составляющая успешности проекта.

Основными новшествами MSF является следующее

- 1. Акцент на внедрении IT-решения.
- 2. Модель процесса, объединяющая спиральную и водопадную модели.
- 3. Особая организация команды – не иерархическая, а как группа равных, но выполняющих разные функции (роли) работников.
- 4. Техника управления компромиссами.

Основные принципы

- 1. *Единое видение проекта. Успех коллективной работы над проектом немыслим* без наличия у членов проектной группы и заказчика единого видения (shared vision), т.е. четкого, и, самое главное, одинакового, понимания целей и задач проекта.
- Как проектная группа, так и заказчик изначально имеют собственные
- предположения о том, что должно быть достигнуто в ходе работы над
- проектом. Лишь наличие единого видения способно внести ясность и
- обеспечить движение всех заинтересованных в проекте сторон к общей цели.
- Формирование единого видения и последующее следование ему являются столь важными, что модель процессов MSF выделяет для этой цели специальную фазу – «Выработка концепции», которая заканчивается соответствующей вехой.

Основные принципы

- 2. *Гибкость – готовность к переменам. Традиционная дисциплина управления* проектами и каскадная модель исходят из того, что все требования могут быть четко сформулированы в начале работы над проектом, и далее они не будут существенно изменяться.
- В противоположность этому MSF основывается на принципе непрерывной изменяемости условий проекта при неизменной эффективности управленческой деятельности.
- 3. *Концентрация на бизнес-приоритетах. Независимо от того, нацелен ли*
- разрабатываемый продукт на организации или индивидуумов, он должен
- удовлетворить определенные нужды потребителей и принести в некоторой
- форме выгоду или отдачу. В отношении индивидуумов это может означать,
- например, эмоциональное удовлетворение – как в случае компьютерных игр.
- Что же касается организаций, то неизменным целевым фактором продукта
- является бизнес-отдача (business value). Обычно продукт не может

Основные принципы

- ***Поощрение свободного общения.*** *Исторически многие организации строили* свою деятельность на основе сведения информированности сотрудников к **минимуму, необходимому для исполнения работы** (need-to-know).
- Зачастую такой подход приводит к недоразумениям и снижает шансы команды на достижение успеха. Модель процессов MSF предполагает открытый и честный **обмен информацией как внутри команды, так и с ключевыми заинтересованными лицами.**
- Свободный обмен информацией не только сокращает риск возникновения недоразумений, недопонимания и неоправданных затрат, но и обеспечивает максимальный вклад всех участников **проектной группы в снижение существующей в проекте неопределенности.** По этой причине модель процессов MSF предлагает проведение анализа хода работы над проектом в определенных временных точках.
- **Документирование** результатов делает ясным прогресс, достигнутый в работе над проектом - как для проектной команды, так и для заказчика и других заинтересованных в проекте сторон.

Основные принципы

- Главная особенность модели команды в MSF является то,
- что она «плоская», то есть не имеет официального лидера. Все отвечают за проект в равной степени, уровень заинтересованности каждого в результате очень высок, а коммуникации внутри группы четкие, ясные, дружественные и ответственные. Конечно, далеко не каждая команда способна так работать – собственно, начальники для того и нужны, чтобы нести основной груз ответственности за проект и, во многом, освободить от него других.
- Демократия в команде возможна при высоком уровне осознанности и
- заинтересованности каждого, а также в ситуации равности в профессиональном уровне (пусть и в разных областях – см. различные ролевые кластеры в команде, о которых речь пойдет ниже).
- С другой стороны, в реальном проекте, в рамках данной модели команды,
- можно варьировать степень ответственности, в том числе вплоть до выделения, при необходимости, лидера.

Особенностей отношений внутри команды

- Одной из особенностей отношений внутри команды является высокая культура дисциплины обязательств:
- • готовность работников принимать на себя обязательства перед другими;
- • четкое определение тех обязательств, которые они на себя берут;
- • стремление прилагать должные усилия к выполнению своих обязательств;
- • готовность честно и незамедлительно иформировать об угрозах выполнению своих обязательств.

Ролевые кластеры.

- MSF основан на постулате о семи качественных целях, достижение которых определяет успешность проекта. Эти цели обуславливают модель проектной группы и образуют ***ролевые кластеры (или просто роли) в проекте.***
- ***В каждом*** ролевом кластере может присутствовать по одному или несколько специалистов, некоторые роли можно соединять одному участнику проекта. **Каждый ролевой кластер** представляет уникальную точку зрения на проект, и в то же время никто из членов проектной группы в одиночку не в состоянии успешно представлять все возможные взгляды, отражающие качественно различные цели.
- Для разрешения этой дилеммы команда соратников (команда равных, team of peers), работающая над проектом, должна иметь четкую форму отчетности перед заинтересованными сторонами (stakeholders) при распределенной ответственности за достижение общего успеха.
- В MSF следующие ролевые кластеры (часто их называют ролями) – см. рис.

Ролевые кластеры

Ролевые кластеры

- ***Управление продуктом (product management). Основная задача этого ролевого*** кластера – обеспечить, чтобы заказчик остался довольным в результате выполнения проекта.

  Этот ролевой кластер действует по отношению к проектной группе как представитель заказчика и зачастую формируется из сотрудников оганизации-заказчика. Он представляет бизнес-сторону проекта и обеспечивает его согласованность со стратегическими целями заказчика. В него же входит контроль за полным пониманием интересов бизнеса при принятии ключевых проектных решений.

- • ***Управление программой (prograт management) обеспечивает управленческие*** функции – отслеживание планов и их выполнение, ответственность за бюджет, разрешение проблем и трудностей процесса, создание условий, при которых команда может работать эффективно, испытывая минимум бюрократических преград.

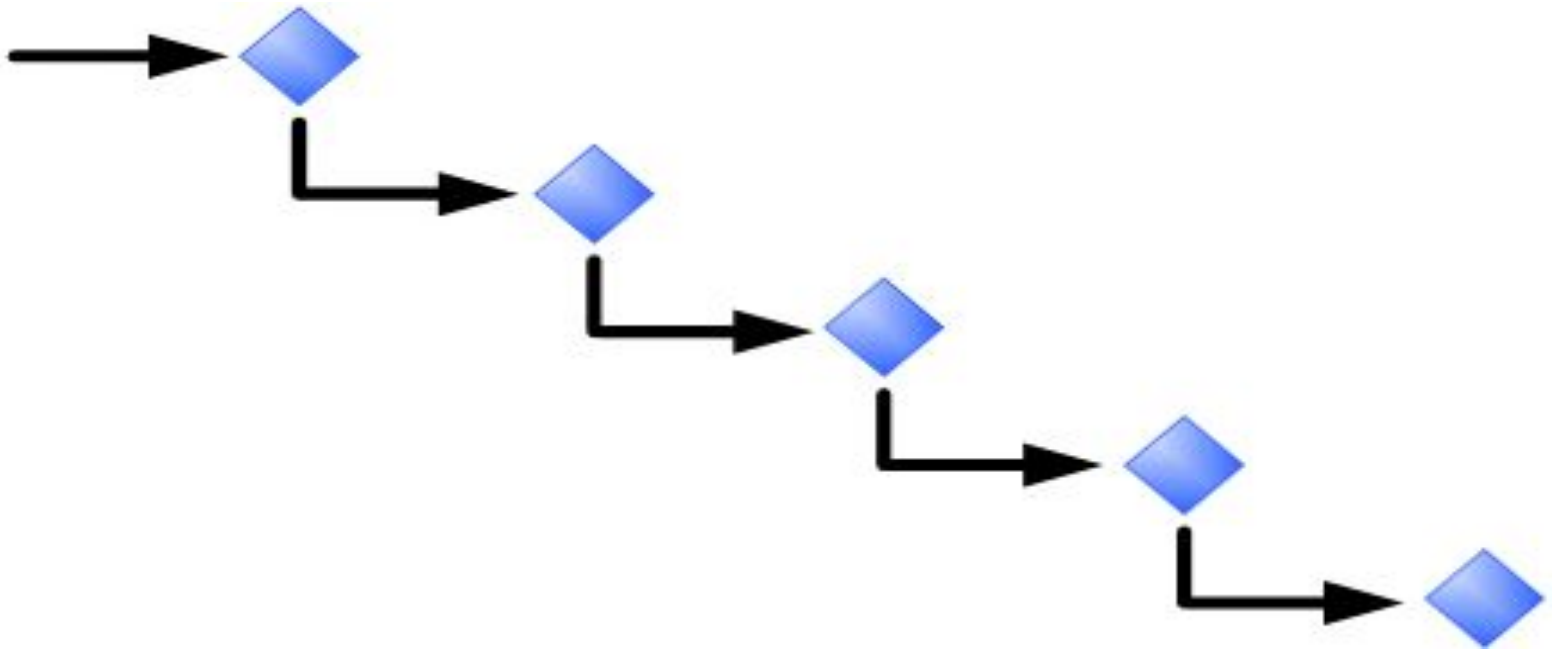
Ролевые кластеры

- • ***Разработка (development). Этот ролевой кластер занимается, собственно,*** программированием ПО.
- • ***Тестирование (test) – отвечает за тестирование ПО.***
- • ***Удовлетворение потребителя (user experience). Дизайн удобного***
- пользовательского интерфейса и обеспечение удобства эксплуатации ПО
- (эргономики), обучение пользователей работе с ПО, создание пользовательской документации.
- • ***Управление выпуском (release management). Непосредственно ответственен за*** беспрепятственное внедрение проекта и его функционирование, берет на себя связь между разработкой решения, его внедрением и последующим сопровождением, обеспечивая информированность членов проектной группы о последствиях их решений.
- • ***Архитектура (Architecture).7 Организация и выполнение высокоуровневого*** проектирования решения, создание функциональной спецификации ПО и управление этой спецификацией в процессе разработки, определение рамок проекта и ключевых компромиссных решений.

Масштабирование команды MSF. Наличие 7 ролевых кластеров не означает, что команда должна состоять строго из 7 человек. Один сотрудник может объединять несколько ролей.

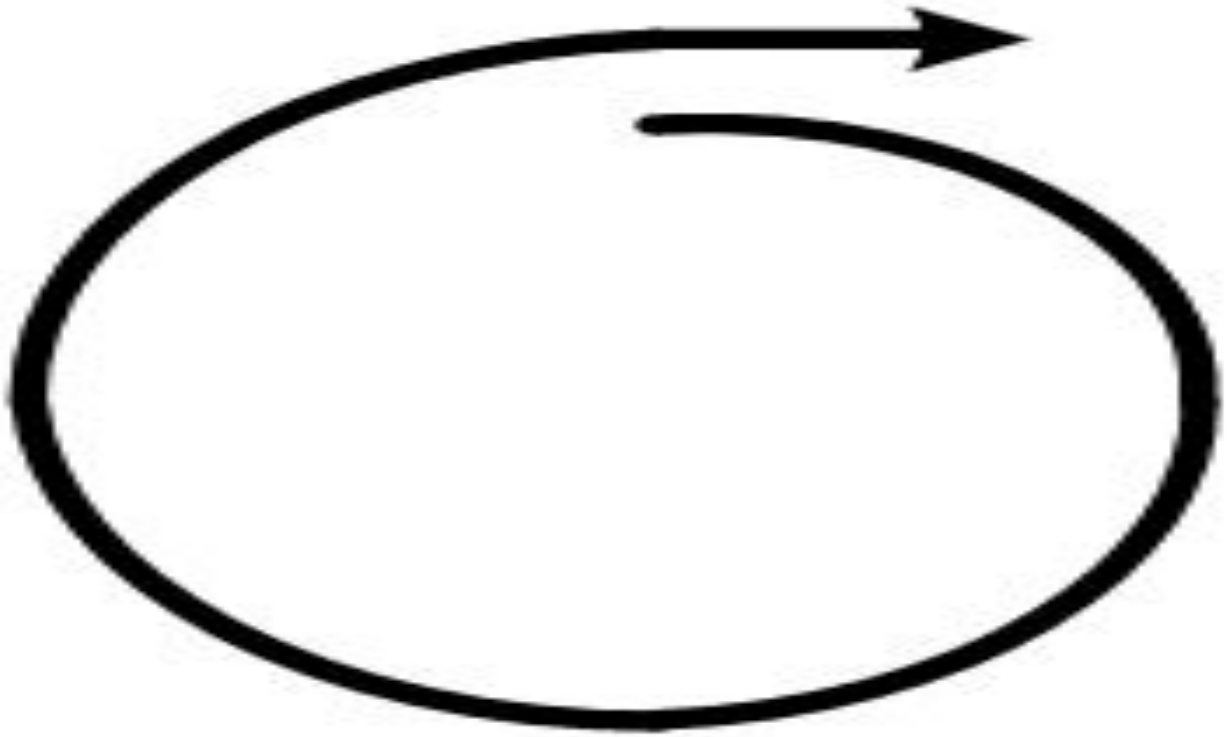
Модель процесса

- Водопадная модель – фазы работ и вехи

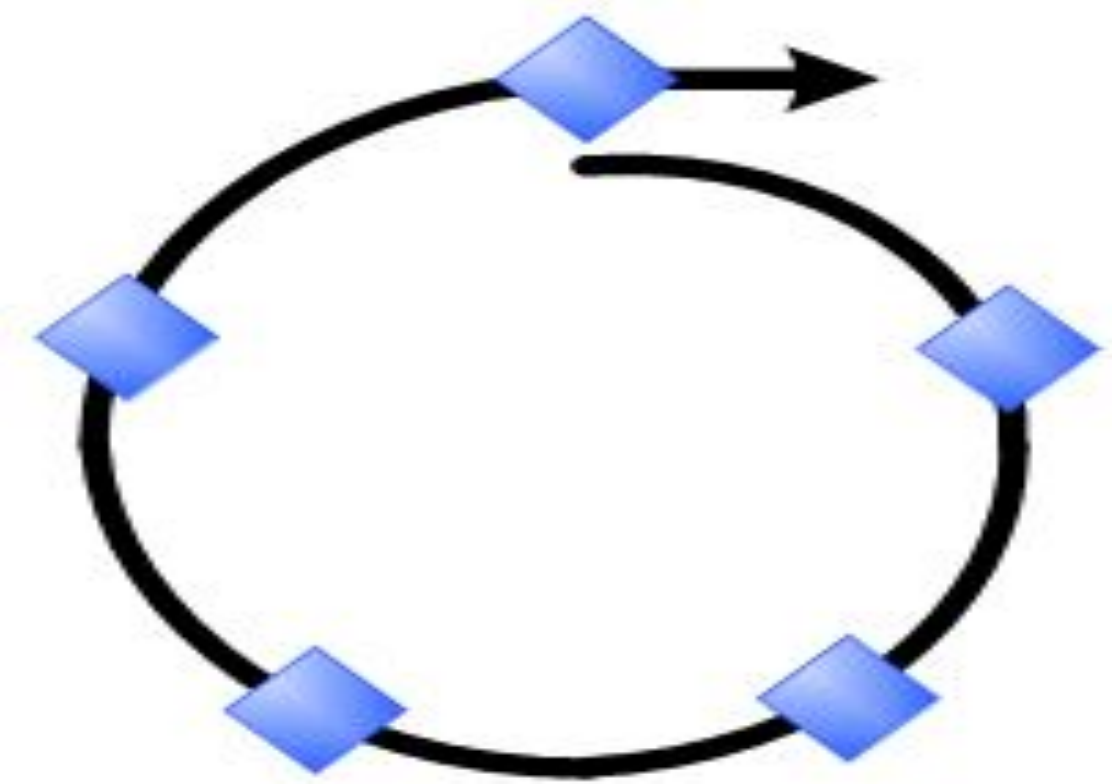


Водопадная модель – постоянное уточнение требований, активное взаимодействие
с заказчиком.
Рис. 8.3

Спиральная модель



В MSF объединяются водопадная и спиральная модели: сохраняются преимущества упорядоченности водопадной модели, не теряя при этом гибкости и творческой ориентации модели спиральной.



Управление компромиссами

Хорошо известна взаимозависимость между ресурсами проекта (людскими и финансовыми), его календарным графиком (временем) и реализуемыми возможностями (рамками). Эти три переменные образуют треугольник

Управление компромиссами

- После достижения равновесия в этом треугольнике изменение на любой из его
- сторон для поддержания баланса требует модификаций на другой (двух других) сторонах и/или на изначально измененной стороне.

	Фиксируется	Согласовывается	Принимается
Ресурсы	✓		
Время		✓	
Возможности			✓

Гибкие» (agile) методы разработки ПП

- Основными положениями гибких методов, закрепленных в Agile Manifesto в 2007 году являются следующие:
- индивидуальное взаимодействие вместо процессов и программных средств;
- работающее ПО вместо сложной документации;
- взаимодействие е с заказчиком вместо жестких контрактов;
- реакция на изменения вместо следования плану.

- Фактически, гибкие методологии говорят о небольших, самоорганизующихся
- командах, состоящих из высококвалифицированных и энергичных людей,
- ориентированных на бизнес, то есть, например, разрабатывающих свой собственный продукт для выпуска его на рынок.
- У этого подхода есть, очевидно, свои плюсы и свои
- Минусы.

Extreme Programming

- Самым известным гибким методом является Extreme Programming (известное
- сокращенное название – XP). Он был создан талантливым специалистом в области программной инженерии Кентом Беком в результате его работы в 1996-1999 годах над системой контроля платежей компании “Крайслер”.
- Модель процесса по XP выглядит как частая последовательность выпусков (releases) продукта, столь частых, сколь это возможно. Но при этом обязательно, чтобы в выпуск входила новая целиковая функциональность

Основные принципы организации процесса по XP.

- 1. Планирование (Planning Game), основанное на принципе, что разработка ПО является диалогом между возможностями и желаниями, при этом изменятся и то и другое.
- 2. Простой дизайн (Simple Design) – против избыточного проектирования.
- 3. Метафора (Metaphor) – суть проекта должна умещаться в 1-2 емких фразах или в некотором образе.
- 4. Рефакторинг (Refactoring) – процесс постоянного улучшения (упрощения) структуры ПО, необходимый в связи с добавлением новой функциональности.
- 5. Парное программирование (Pair Programming) – один программирует, другой думает над подходом в целом, о новых тестах, об упрощении структуры программы и т.д.
- 6. Коллективное владение кодом (Collective Ownership).
- 7. Участие заказчика в разработке (On-site Customer) – представитель заказчика входит в команду разработчика.
- 8. Создание и использование стандартов кодирования (Coding Standards) в проекте – при написании кода (создаются и) используются стандарты на имена
- идентификаторов, составление комментариев и т.д.
- 9. Тестирование – разработчики сами тестируют свое ПО, перемежая этот процесс с разработкой. При этом рекомендуется создавать тесты до того, как будет реализована соответствующая функциональность. Заказчик создает функциональные тесты.
- 10. Непрерывная интеграция. Сама разработка представляется как последовательность выпусков.
- 11. 40-часовая рабочая неделя.

Scrum

- **История.**

- В 1986 японские специалисты Hirotaka Takeuchi и Ikujiro Nonaka
- опубликовали сообщение о новом подходе к разработке новых сервисов и продуктов (не обязательно программных).
- Основу подхода составляла сплоченная работа небольшой универсальной команды, которая разрабатывает проект на всех фазах. Приводилась аналогия из регби, где вся команда двигается к воротам противника как единое целое, передавая (пасуя) мяч своим игрокам как вперед, так и назад. В начале 90-х годов данный подход стал применяться в программной индустрии и обрел название Scum (термин из регби, означающий - схватка)

Общее описание.

- **Метод Scrum позволяет гибко разрабатывать проекты**
- небольшими командами (7 человек плюс/минус 2) в ситуации изменяющихся требований.
- При этом процесс разработки итеративен и предоставляет большую свободу команде.
- Кроме того, метод очень прост – легко изучается и применяется на практике.

Схема метода Scrum

- Создание
- требований к
- Продукту

- Планирова-
- ние итерации

- Выполнение
- итерации
- (2-4 недели)

- Анализ результатов,
- пересмотр требований

Роли.

- **В Scrum есть всего три вида ролей.**
- *Владелец продукта (Product Owner) – это менеджер проекта, который представляет* в проекте интересы заказчика. В его обязанности входит разработка начальных требований к продукту (Product Backlog), своевременное их изменение требований, назначение приоритетов, дат поставки и пр. Важно, что он совершенно не участвует в выполнении самой итерации.
- *Scrum-мастера (Scrum Master) обеспечивает максимальную работоспособность* и продуктивную работу команды – как выполнение Scrum-процесса, так и решение хозяйственных и административных задач. В частности, его задачей является ограждение команды от всех воздействий извне во время итерации.
- *Scrum-команда (Scrum Team) – группа, состоящая из пяти–девяти* самостоятельных, инициативных программистов. Первой задачей команды является постановка для итерации реально достижимых и приоритетных для проекта в целом задач (на основе Project Backlog и при активном участии владельца продукта и Scum-мастера).
- Второй задачей является выполнение этой задачи во что бы то ни стало, в отведенные сроки и с заявленным качеством. Важно, что команда сама участвует в постановке задачи и сама же ее выполняет.

Практики.

- **В Scrum определены следующие практики.**
- Sprint Planning Meeting. Проводится в начале каждого Sprint. Сначала Product Owner, Scurm-мастер, команда, а также представители заказчика и пр. Заинтересованные лица определяют, какие требования из Project Backlog наиболее приоритетные и их следует реализовывать в рамках данного Sprint. Формируется Sprint Backlog. Далее Scurm-мастер и Scrum-команда определяют то, как именно будут достигнуты определенные выше цели из Sprint Backlog. Для каждого элемента Sprint Backlog определяется список задач и оценивается их трудоемкость.
- *Daily Scrum Meeting – пятнадцатиминутное каждодневное совещание, целью* которого является достичь понимания того, что произошло со времени предыдущего совещания, скорректировать рабочий план сообразно реалиям сегодняшнего дня и обозначить пути решения существующих проблем. Каждый участник Scrum-команды отвечает на три вопроса: что я сделал со времени предыдущей встречи, мои проблемы, что я буду делать до следующей встречи? В этом совещании может принимать участие любое заинтересованное лицо, но только участники Scrum-команды имеют право принимать решения. Правило обосновано тем, что они давали обязательство реализовать цель итерации, и только это дает уверенность в том, что она будет достигнута. На них лежит ответственность за их собственные слова, и, если кто-то со стороны вмешивается и принимает решения за них, тем самым он снимает ответственность за результат с участников команды. Такие встречи поддерживают дисциплину обязательств в Scrum-команде, способствуют удержанию фокуса на целях итерации, помогают решать проблемы «в зародыше». Обычно такие совещания проводятся стоя, в течение 15-20 минут.

Практики.

- *Sprint Review Meeting. Проводится в конце каждого Sprint. Сначала Scrum-команда* демонстрирует Product Owner сделанную в течение Sprint работу, а тот в свою очередь ведет эту часть митинга и может пригласить к участию всех заинтересованных представителей заказчика.
- Product Owner определяет, какие требования из Sprint Backlog были выполнены, и обсуждает с командой и заказчиками, как лучше асставить приоритеты в Sprint Backlog для следующей итерации. Во второй части митинга производится анализ прошедшего спринта, который ведет Scrum-мастер. Scrum-команда анализирует в последнем Sprint положительные и отрицательные моменты совместной работы, делает выводы и принимает важные для дальнейшей работы решения. Scrum- команда также ищет пути для увеличения эффективности дальнейшей работы. Затем цикл повторяется.