

**Scrum** - система разработки ПО , основанная на делении всего процесса на итерации, где в конце каждой из них команда готова предоставить демо версию продукта.

рекомендуется команда не более 10 чел.). В их число входит:

В Scrum выделяют:

-**Scrum команда,**

**-События,**

**-артефакты**

**-метрики** позволяющие оценивать эффективность работы команды и отдельно каждого его члена, создовать репорты для менеджмента

**Scrum команда**

1)-владелец продукта(product owner - ответственный за бизнес часть может быть например представитель заказчика). Занят управлением бэклогом продукта (бэк лог-набор требований которые мы должны реализовать чтобы продукт был полностью готов), product owner - он собирает все эти требования и приводит их в такое состояние чтобы эти требования понимали все участники проекта. -расставляет приоритеты на выполнение этих требований.

2)-scrum master(человек который отвечает за эффективность и правильность применения scrum методологии в команде). Знает всё о скраме, помогает, обучает, направляет команду, но не участвует в разработке. Отвечает чтобы команда работала по скраму.

3)- разработчики/team (имеется в виду и разраб., и тестировщики все кто работает над продуктом). рекомендуется неболее 9 чел. Это команда самоорганизующаяся, все равны. Команда не делится на разраб., тестировщиком., бизнес аналитиков, … все обладают одинаковыми компетенциями, все взаимозаменяемые. Есть понятие все несут коллективную ответственность, если кто то ошибся это дело каждого.

**События**

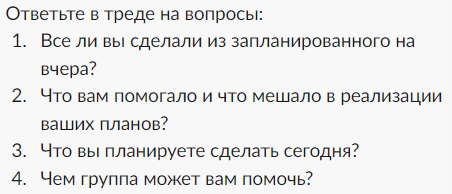
Итерация называется **sprint** - временной отрезок 2-4 недели в течении которого мы создаём версию продукта которая полностью готова для конечного пользователя

Существует 4 организационных моментов в рамказ спринта:

**1)-Планирование спринта (итерации)** собирается вся команда, решается каким будет продукт в конце спринта, какие требования(часть из всех требований) будут реализованы в конце спринта.

**2)-daily meetings / ежедневный скрам (daily stand up), дэйлик**- ежедневный созвон всей команды для планирования дел на день и есть ли у кого проблемы, это даёт возможность оперативно решать возникающие проблемы + эффективно отрабатывать поставленные на этот день задачи + команда может давать советы как что-то решить эффективно.

Пример:



**3)-Sprint review** - подведение итогов спринта, что получилось, что нет, почему получилось, почему нет, все ли задачи были решены

**4)-** **Demo -** Демонстрация нашего инкремента продукт оунеру, как работает инкремент, отдельное мероприятие котое проходит после обзора спринта

**5)-Retrospective meeting** - это когда команда собирается одна, без заказчика и обсуждает всё ли идёт у них хорошо с процессом работы / взаимодействия и т.д. Если были какието ошибки то как их избежать в будущем.

**артефакты**

**артефакты** — это информация (различные понятия), с помощью которой Scrum-команда и заинтересованные стороны подробно описывают разрабатываемый продукт, а также действия по его созданию и деятельность в рамках проекта

**BackLog продукта / user stories** - некий список функциональных требований, он постоянно обновляется. Например мы набросали BackLog =200 требований, это относится ко всему продукту в целом, затем мы набрасываем BackLog для спринта из требований для продукта. ← это называется планирование спринта. Когда мы переносим требования в спринт мы их как бы ещё более детализируем(разбиваем).

**Бэклог спринта-** содержит несколько заданий (требований) из бэклога продукта, которые планируются для выполнения в ходе следующего (или текущего) инкремента (спринта) продукта.

**Уточнение бэклога продукта (product backlog refinement)**- собирается вся команда, продукт овнер представляет новые требования, ему задают вопрсы/уточнения и если оно не понятно команде то он её перерабатывает чтобы было понятно всем. Могут подыматся/решатся и другие моменты.

**Критерии подготовленности (неготовности!)(Definition ready)-** фокусируются на уровне бэклог, мы оцениваем юзер стори которую предлагает нам продукт овнер на степень того как мы их понимаем. Другими словами хорошо ли нам понятны требования описанные продукт овнером. А как это понять? Вспоминаем про набор свойств которыми должно обладать требование (завершонность, атомарность, непротиворечивость/последовательность, недвусмысленность, выполнимость…). Требование - другими словами это юзер стори.

Благодаря этому артефакту заказчику удаётся создавать хорошие юзер стори готовые для разработки, так как в обсуждении участвует вся команда, а не только заказчик где то посидел, надумал а вы вот нате делайте, что на выходе получится не понятно.

**Критерии готовности (definition of done)-** фокусируются на уровне спринта/итерации/релиза,(другими словами мы обсуждаем это во время планирования самого спринта.) и относятся непосредственно к оценке степени готовности функциональности, чтобы понять готова та или иная функциональность или нет. Ещё другими словами - мы выстывляем для себя критерии по которым мы определим что функционал готов.

несколько несвязанных друг с другом примеров:

- у нас написаны юнит тесты и они все успешно прошли

-у нас написаны интеграционные тесты и они все успешно прошли

-количество критичных багов < 10%

**Пользовательские истории** - формулировка намерения описывающая что-то, что система должна делать для пользователя.(какое то требование которое мы должны реализовать). Оно описывается определёнными ключевыми словами :

1) As a (role) Как пользователь Вася

2) I can (Functionality) Я могу делать что-то (например купить телефон)

3) So that (Rationale) Для чего (какаято ценность), чтобы подорить комуто

4) Acceptance criteria-критерии приёмки Как это будет реализовано

**Покер планирование (Planning pocker)-** это оценка юзер сторис, собирается команда и проводится голосование сколько присвоить поинтов(чем больше поинтов тем сложнее сторя) каждой из стори. Сторя = требование. Если по какойто стори большие расхождения после голосования (кто-то дал 1 поинт, кто-то 13), то по этой стори назначается обсуждение, по окончанию которого сторя переоценивается.

Другими словами это привязка сложности к требованию, основанная на предыдущем опыте. Надо отметить, что сами поинты условные, к примеру можно использовать размер одежды: xs, s, m, l, xl, xxl, xxxl. В зависимости от размера можно интуитивно догодатся о серьёзности в реализации той или иной юзер стори (требования)

**Инкремент продукта** - всё то, что было разработано в предыдущие спринты + то, что получили по окончанию текущего (последнего) спринта.

**Метрики**

**Velocity -** скорость скрам команды, выщитывается - берётся количество реализованных стори в каждом спринте и делится на количество спринтов. Полезна для обьективного планирования очередного спринта.

**Capacity** - Чтобы оценить объем работы над Элементом Бэклога Продукта, Скрам-команды обычно используют Стори Поинты. Это условная величина, позволяющая давать Элементам Бэклога относительные веса. Чаще всего для оценки в Стори Поинтах используются числа Фибоначчи (1, 2, 3, 5, 8, 13, …), что позволяет провести оценку достаточно быстро. **Capacity** - вместимость спринта. Количество стори-поинтов, которые были определены для элементов из Sprint Backlog. т.е. получается берётся во внимание не количество сторей которые команда решала в предыдущих спринтах, а именно сложность этих сторей, которая и выражается поинтами. Это опять же делают для планирования сроков завершения.

**Диаграмма сгорания задачь (Burned down chart)-**

**Накопительная диаграмма потока (Cumulative flow diagram)-**

Преимущества:

-Заказчик видит прогресс каждый законченный спринт(2-4 недели).

-Все Имеют обратную связь, и заказчик и команда. Идём ли мы в правильном направлении, можем вносить какие то изменения.

-Малое количество документации

И надо понимать что скрам это не 100% аксиома а какой то оптимальный вариант, соответственно от команды к команде, от продукта к продукту могут быть какие то свои изменения в данной системе.