SCRUM

Product backlog

- **ID** – уникальный идентификатор – просто порядковый номер.
- **Название** – краткое описание истории.
- **Важность** (**Importance**) – степень важности данной задачи, по мнению product owner'а.
- **Как продемонстрировать** (**how to demo**) – краткое пояснение того, как завершённая задача будет продемонстрирована в конце спринта.
- **Примечания** – любая другая информация.

Дополнительные поля для user story

- Категория (track).
- Компоненты (components) – указывает, какие компоненты будут затронуты при реализации истории.
- Инициатор запроса (requestor).
- ID в системе учёта дефектов (bug tracking ID).

Подготовка к планированию спринта

- Product backlog должен существовать!
- У каждого продукта должен быть один product backlog и один product owner.
- Все наиболее важные задачи должны быть классифицированы по уровню важности, а их числовые значения не должны совпадать.
- Product owner должен понимать каждую историю.

Итоги планирования спринта

- Цель спринта.
- Список участников команды.
- Sprint backlog.
- Дата демонстрации.
- Место и время проведения ежедневного Scrum'а.

Почему без product owner'а не обойтись

- Команде и product owner’у просто необходимо планировать вместе, потому что каждая user story имеет три параметра, которые очень тесно связаны между собой.

Почему качество не обсуждается

- Жертвовать внутренним качеством – это практически всегда очень и очень плохая идея. Сэкономленное время ничтожно мало по сравнению с той ценой, которую вам придётся заплатить как в ближайшем будущем, так и в перспективе. Как только качество вашего кода ухудшится, восстановить его будет очень тяжело.

Распорядок встречи по планированию спринта

- 13:00 – 13:30. Product owner разъясняет цель спринта и рассказывает про бизнес-процессы из product backlog'а. Обсуждается время и место проведения демо.
- 13:30 – 15:00. Команда проводит оценку времени, которое потребуется на разработку бизнес-процессов и, при необходимости дробит их на более мелкие.
- 15:00 – 16:00. Команда определяет набор user story, которые войдут в следующий спринт.
- 16:00 – 17:00. Договариваемся о времени и месте проведения ежедневного Scrum'а. После чего приступаем к разбиению user story на задачи.

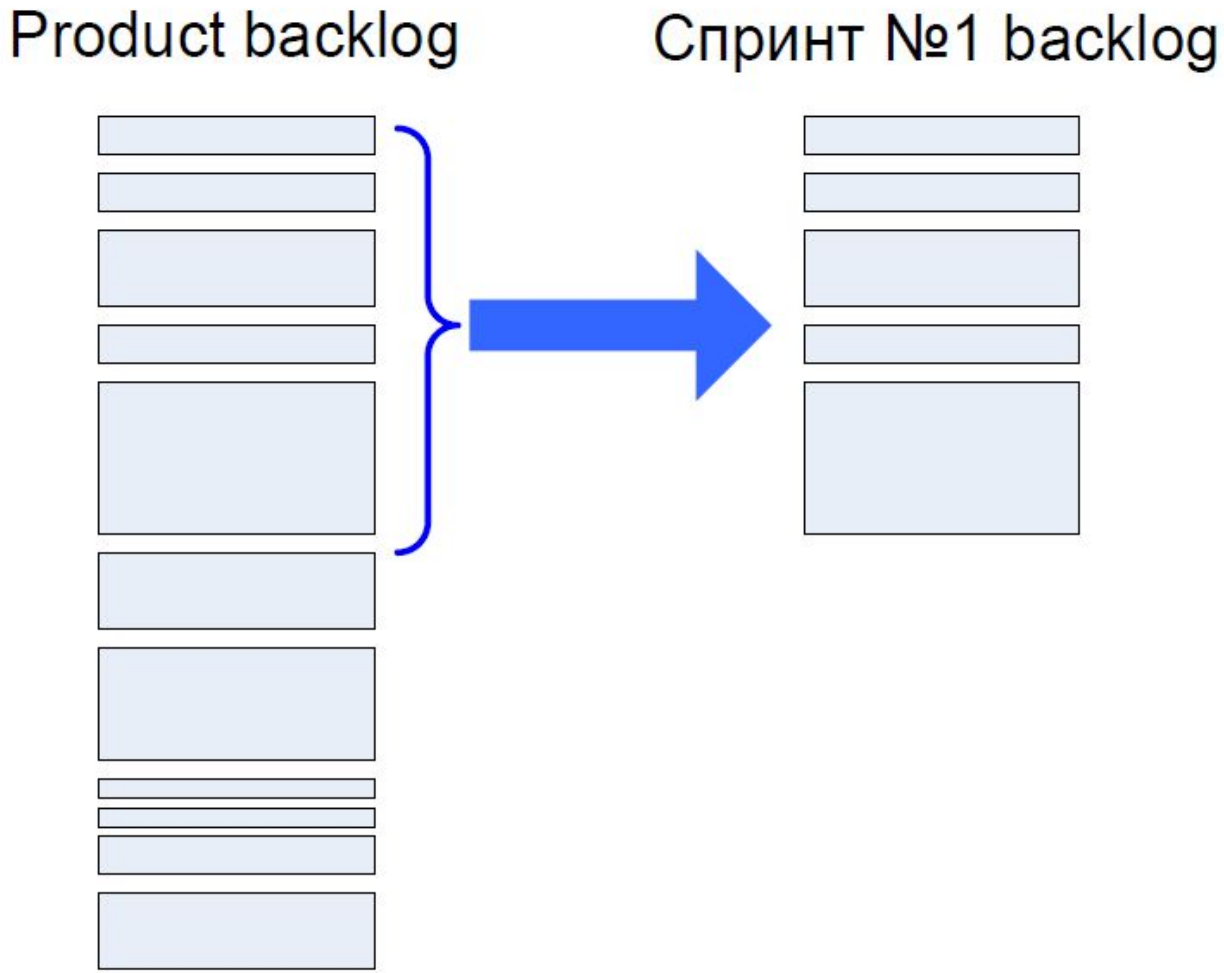
Определение длины спринта

- Одна из основных задач планирования спринта – это определение даты демо. А это значит, что вам придётся определиться с длиной спринта.

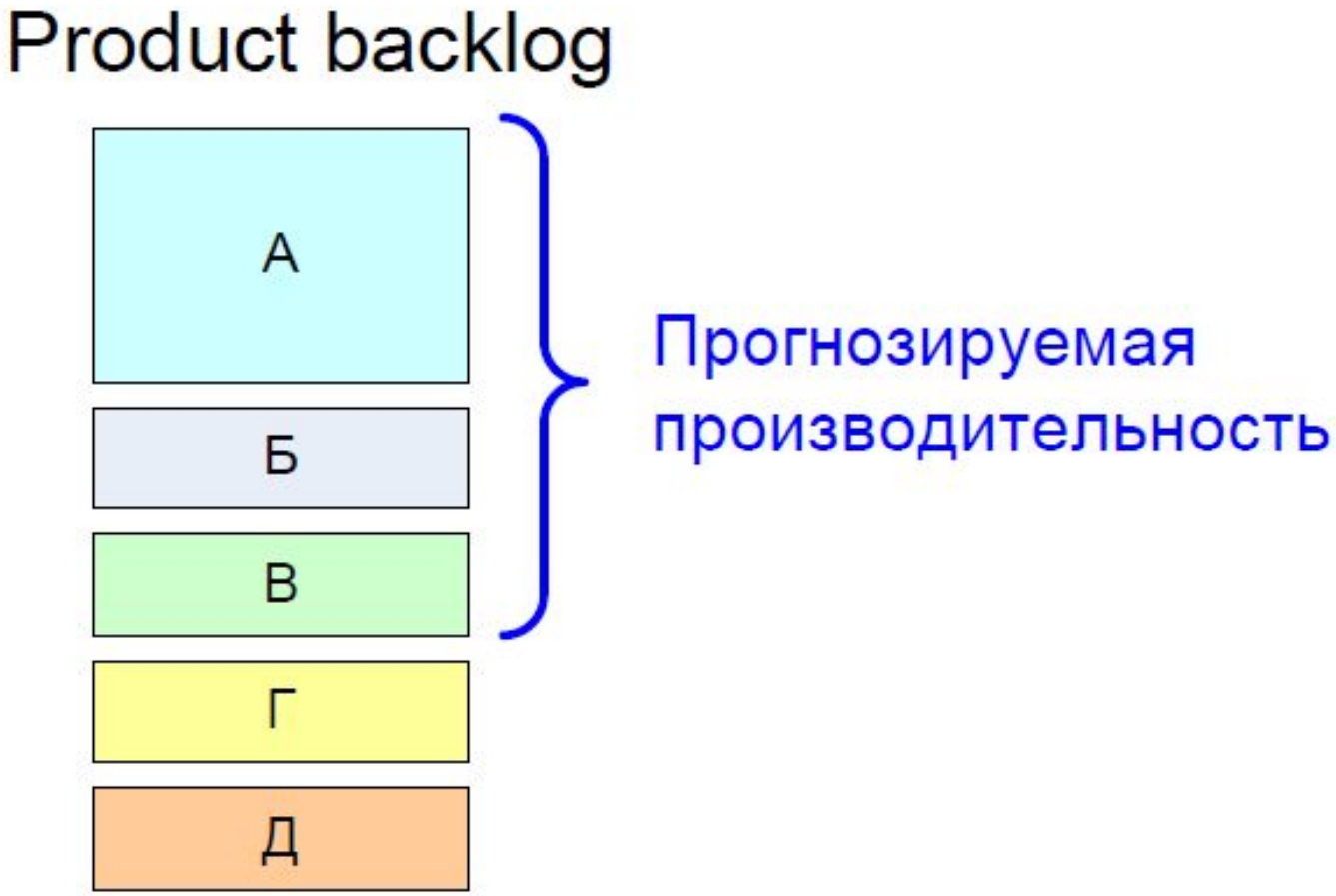
Определение цели спринта

- Цель спринта должна отвечать на главный вопрос “Зачем мы работаем над этим спринтом? Почему мы все просто не уйдём в отпуск?”.

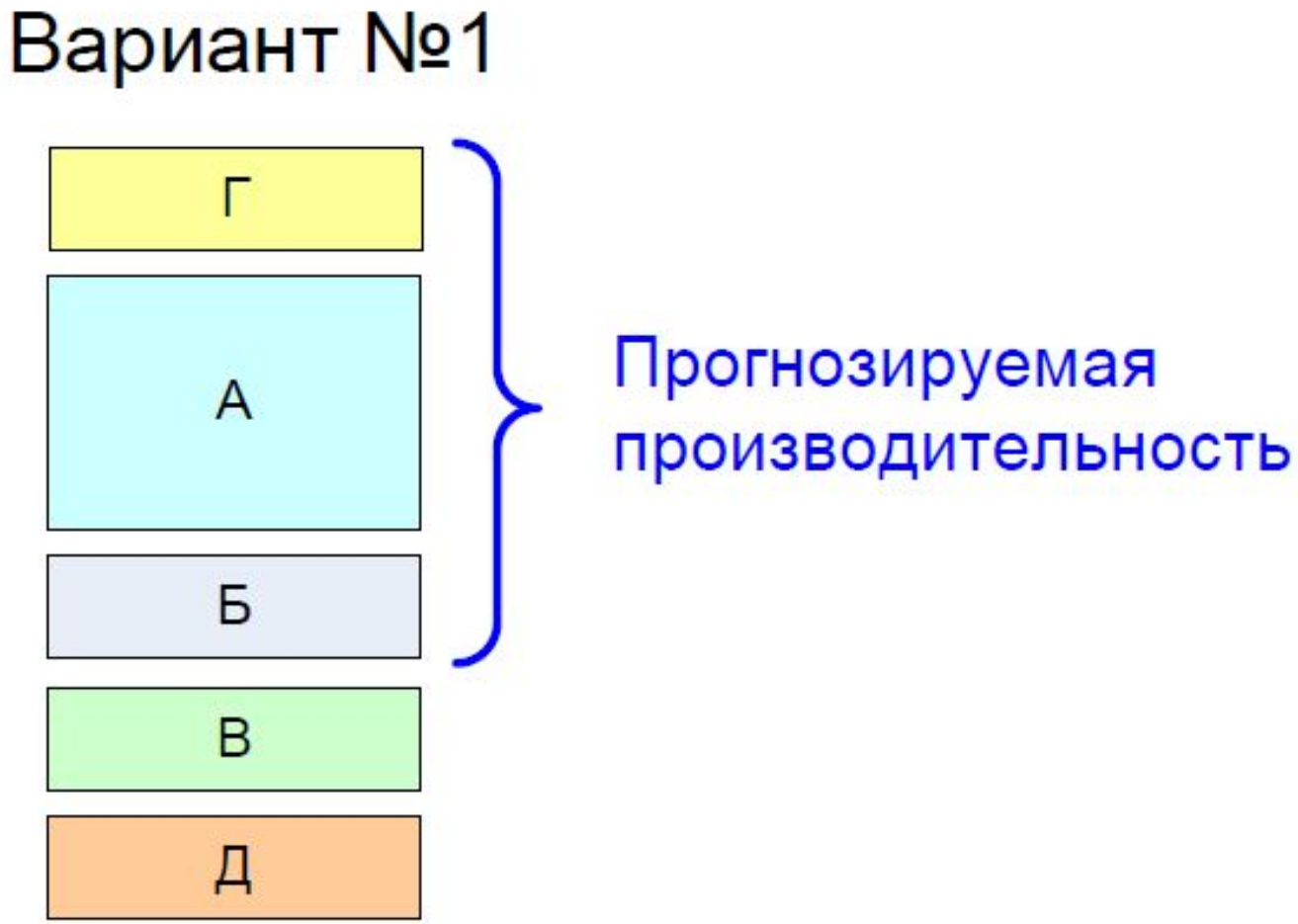
Выбор историй, которые войдут в спринт



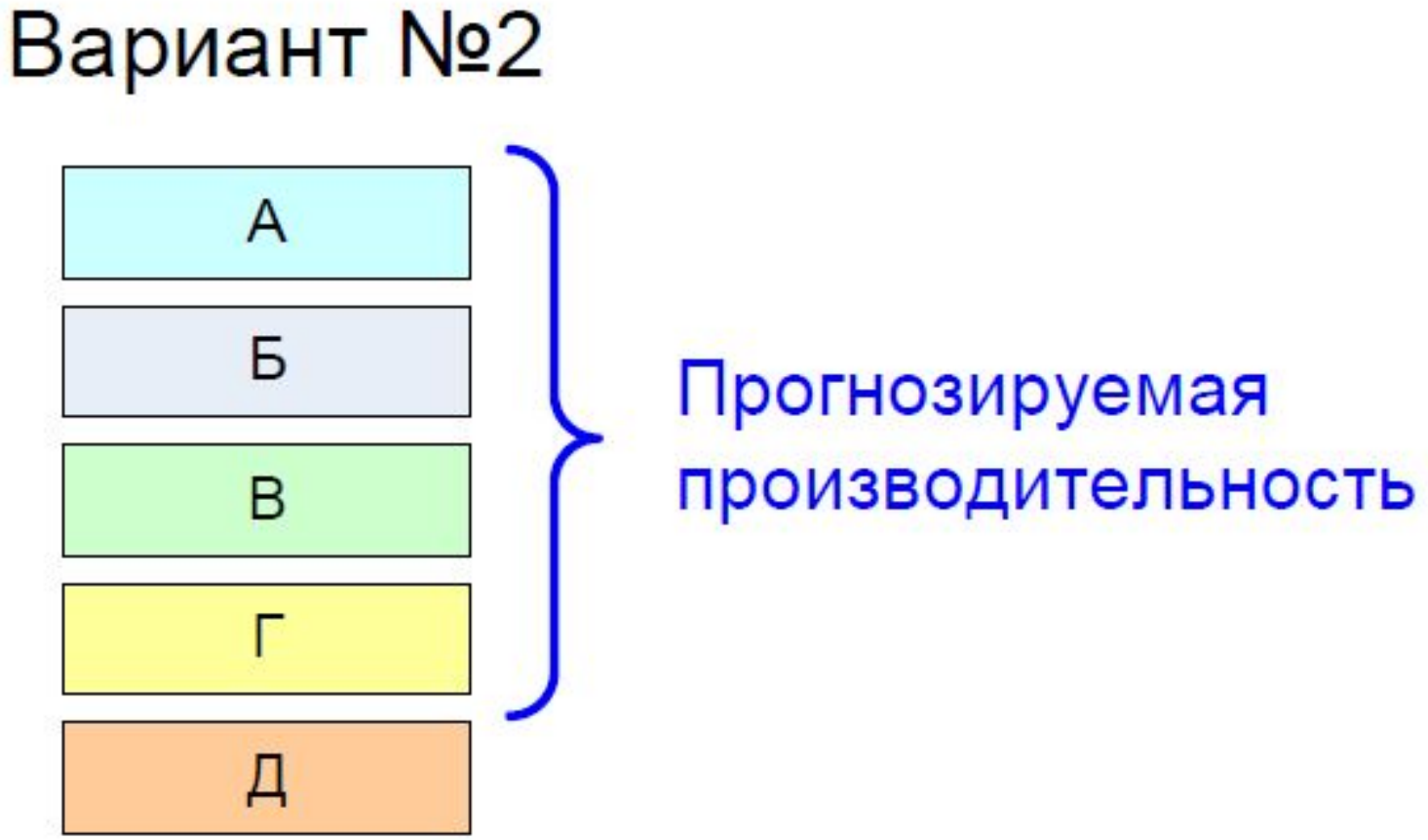
Как product owner может влиять на то, какие истории попадут в спринт?



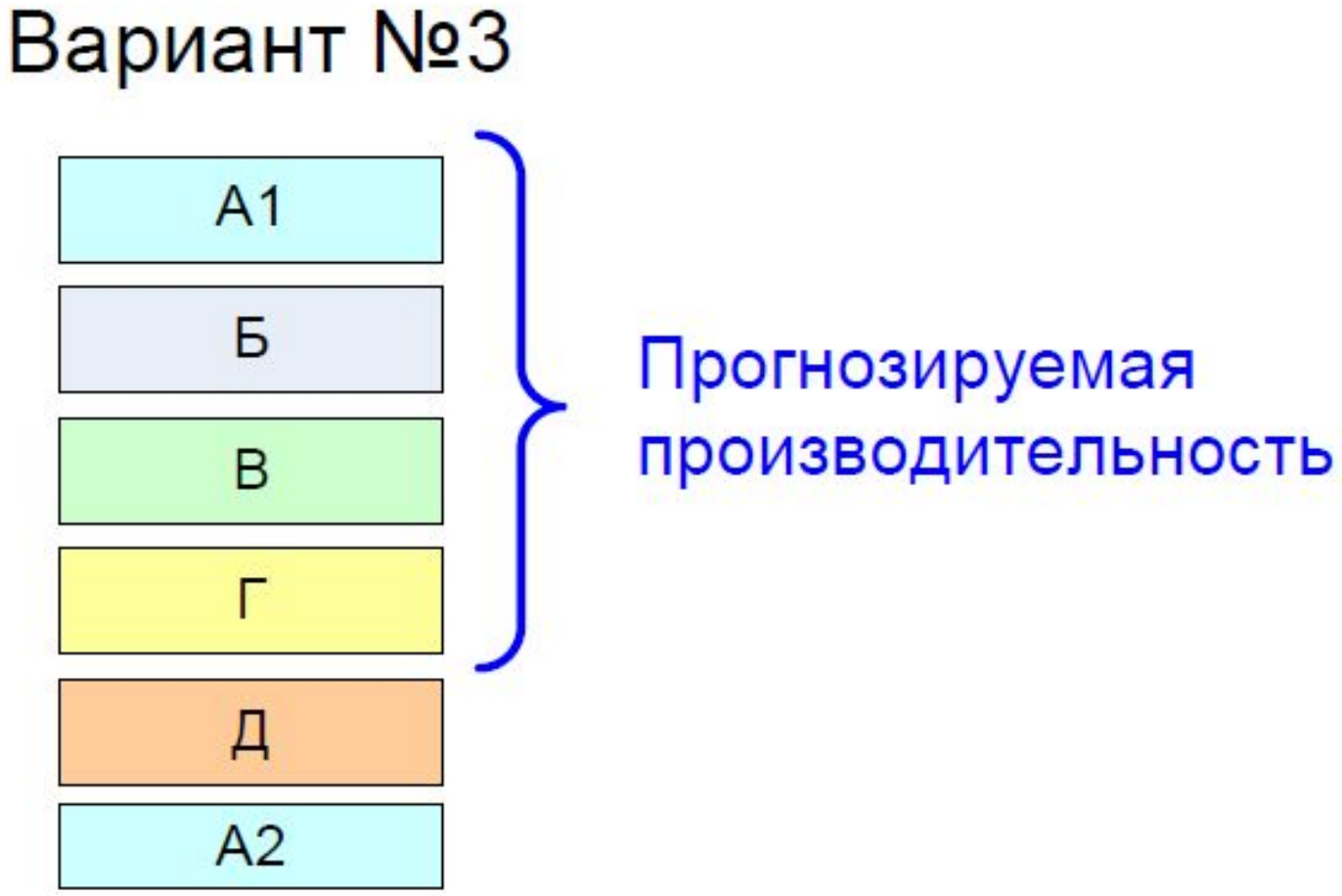
Как product owner может влиять на то, какие истории попадут в спринт?



Как product owner может влиять на то, какие истории попадут в спринт?



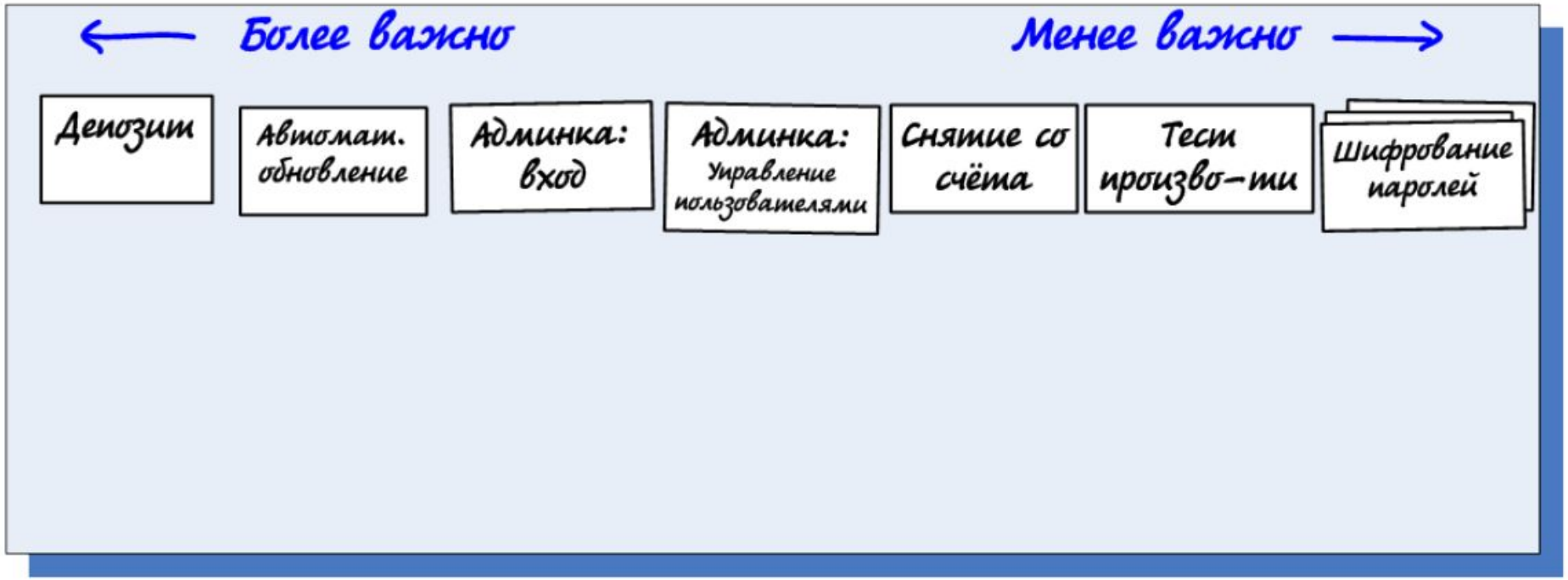
Как product owner может влиять на то, какие истории попадут в спринт?



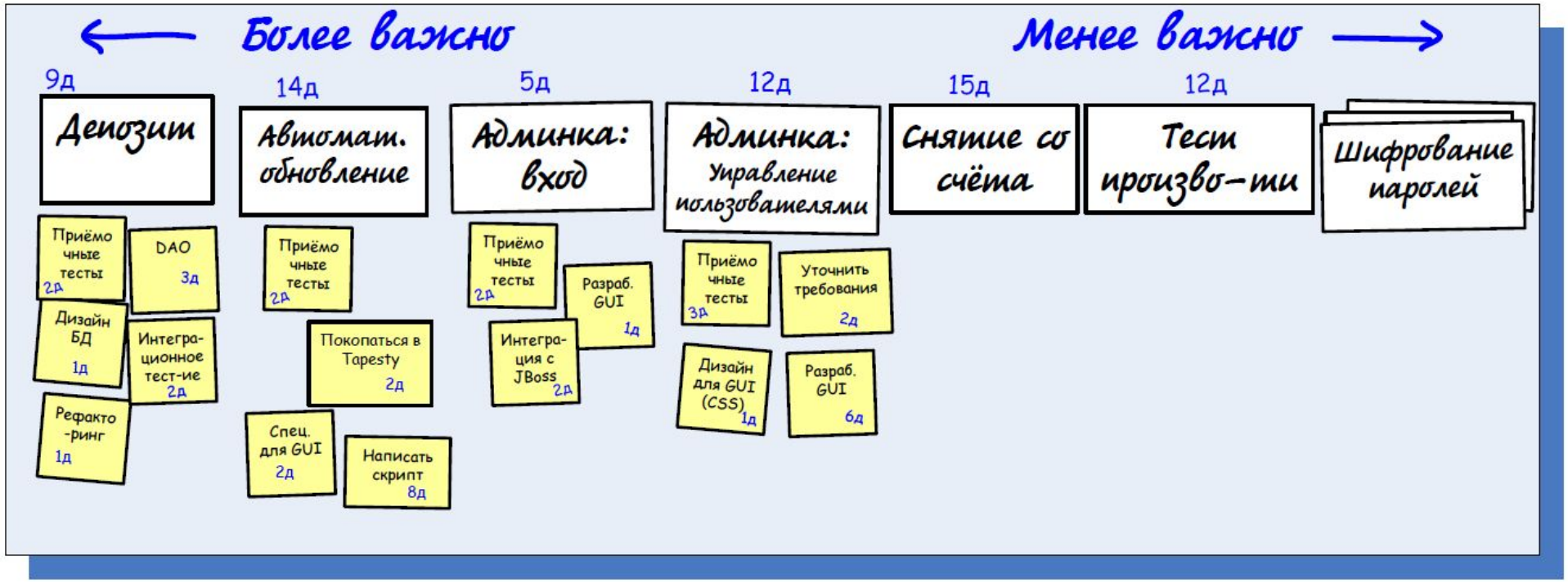
Как команда принимает решение о том, какие истории включать в спринт?

- на основе интуиции
- на основе подсчёта производительности

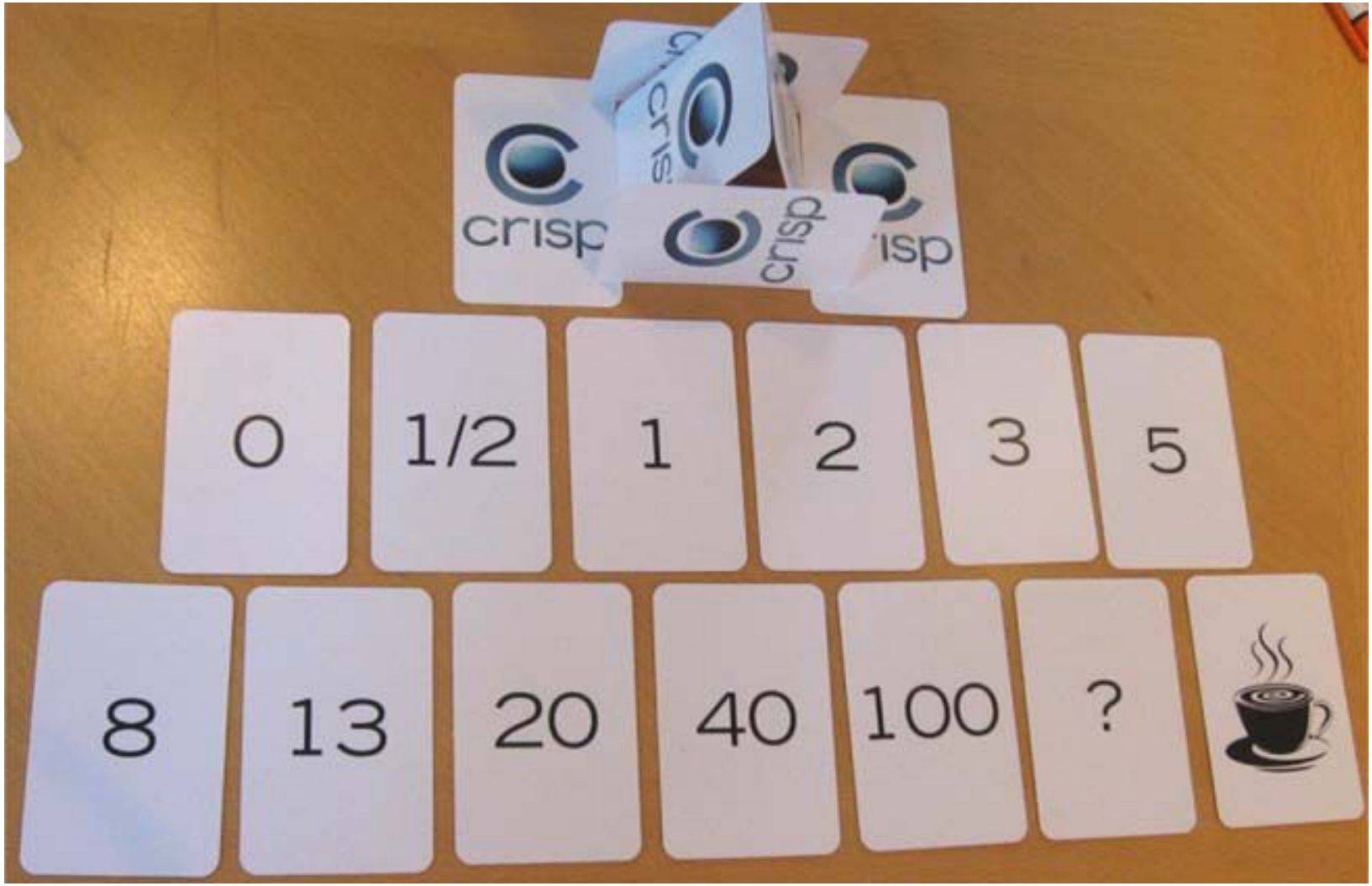
Почему стоит использовать учетные карточки



Разбиение историй на задачи



Оценка трудозатрат с помощью игры в planning poker



Уточнение описаний историй

- Нет ничего ужасней, чем ситуация, когда команда с пафосом демонстрирует новую функциональность продукта, а product owner тяжко вздыхает и говорит: “ну да – всё это красиво, вот только *не то, что я просил!*”

Разбиение историй на более мелкие истории

- Истории должны быть не слишком маленькими, но и не слишком большими (в смысле оценок).

Разбиение историй на задачи

Выбор времени и места для ежедневного Scrum'а

- Все часто забывают, что на планировании спринта, помимо всего прочего, необходимо выбрать время и место проведения ежедневного Scrum'а. Без этого ваш спринт обречён на неудачный старт. Ведь первый ежедневный Scrum – это, по большей части, ввод мяча в игру, когда каждый решает с чего начать работу.

Когда пора остановиться

- **Приоритет №1:** Цель спринта и дата демонстрации.
- **Приоритет №2:** Список историй, которые команда включила в sprint backlog.
- **Приоритет №3:** Оценки для каждой истории из sprint backlog'а.
- **Приоритет №4:** Поле "Как продемонстрировать" для каждой истории из sprint backlog'а.
- **Приоритет №5:** Расчёты производительности и ресурсов в качестве "испытания реальностью" для плана на спринт.
- **Приоритет №6:** Определённое время и место проведения ежедневного Scrum'а.
- **Приоритет №7:** Истории, разбитые на задачи.

Технические истории

- Всё, что должно быть сделано, но невидимо для заказчика, не относится ни к одной user story, и не даёт прямой выгоды product owner'у.

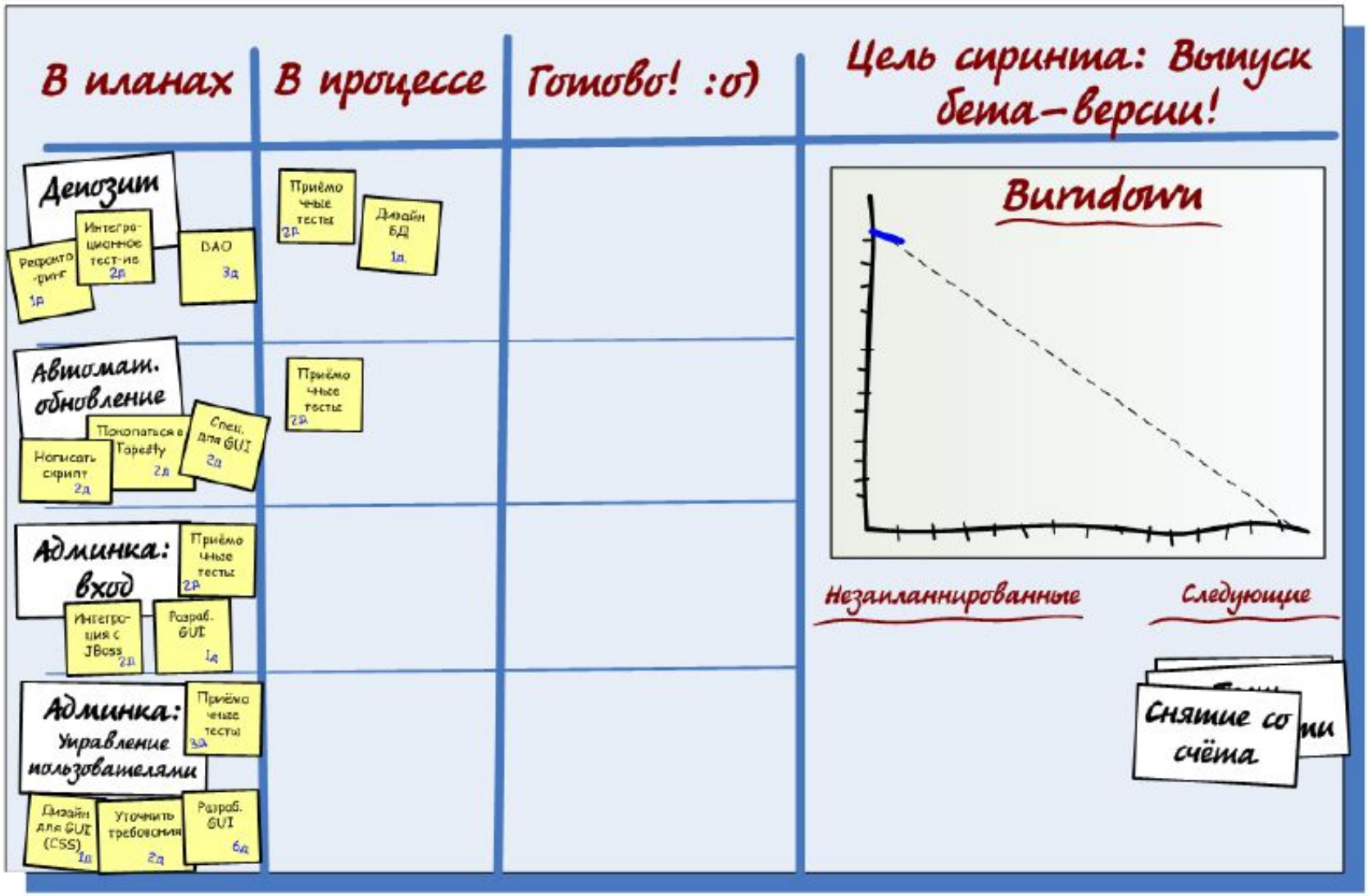
Использование учёта дефектов для ведения product backlog'а

- Product owner распечатывает самые высокоприоритетные задачи из Jira, выносит их на планирование спринта и вешает их на стенку с другими историями.
- Product owner создаёт истории, соответствующие задачам из Jira.
- Работы по исправлению ошибок не включаются в спринт, то есть команда определяет довольно низкий фокус-фактор.
- Заносим product backlog в Jira. Считаем баги обычными историями.

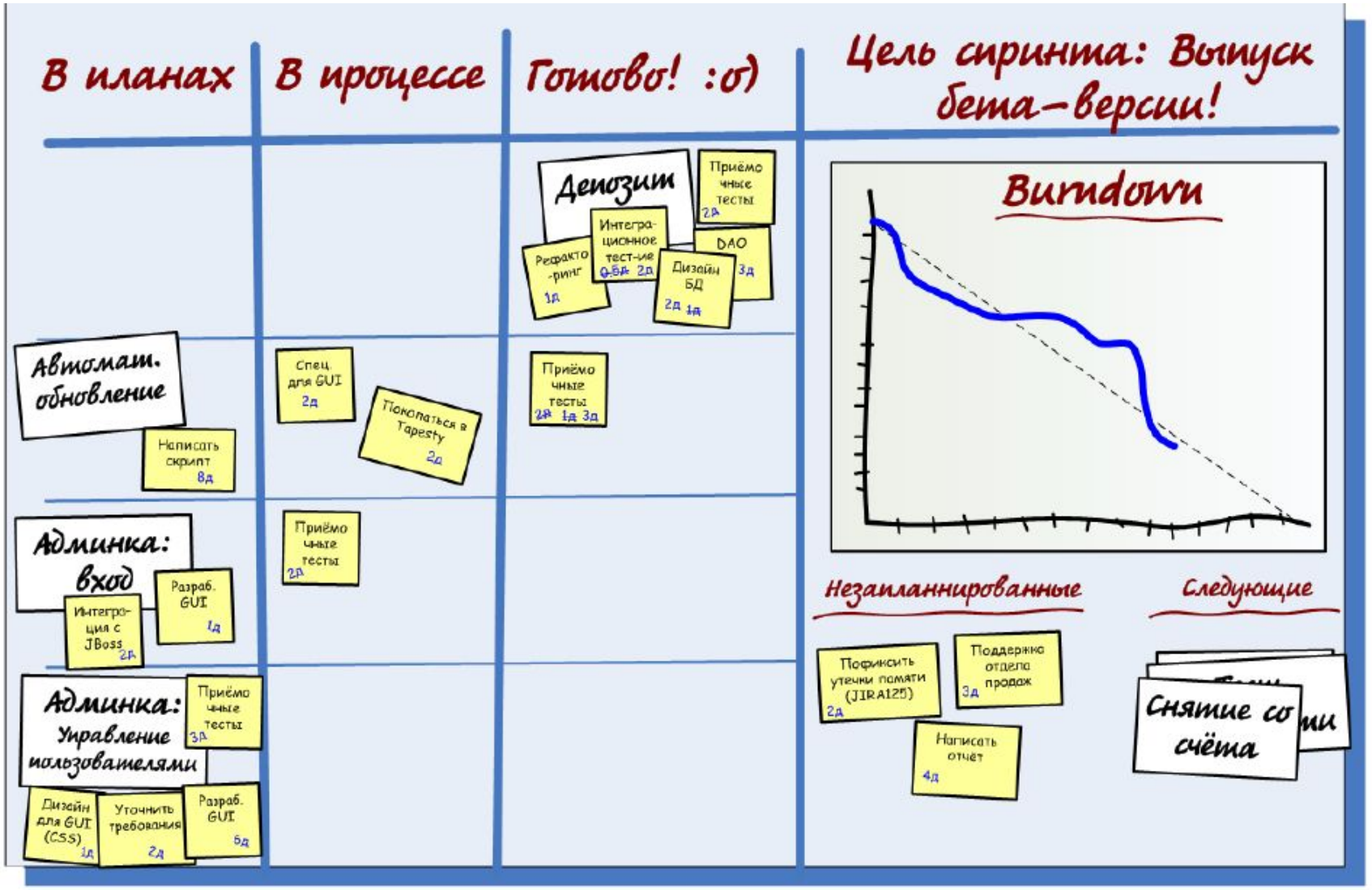
Как донести информацию о спринте до всех в компании

- Важно информировать всю компанию о том, что происходит в вашей команде. Если этого не делать, то остальные начнут жаловаться, или – что ещё хуже – придумывать всякие ужасы про вас.

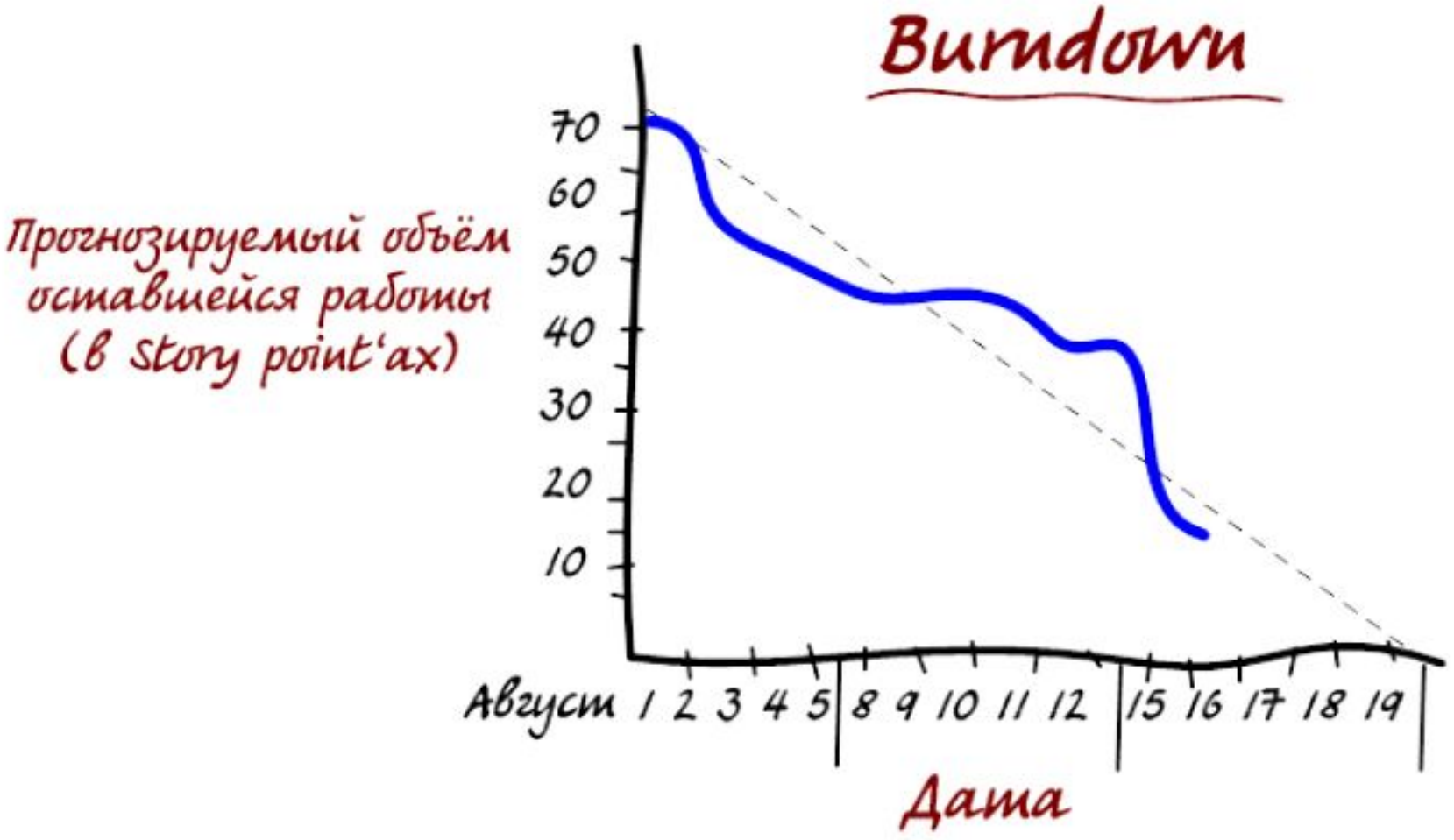
Формат sprint backlog'а



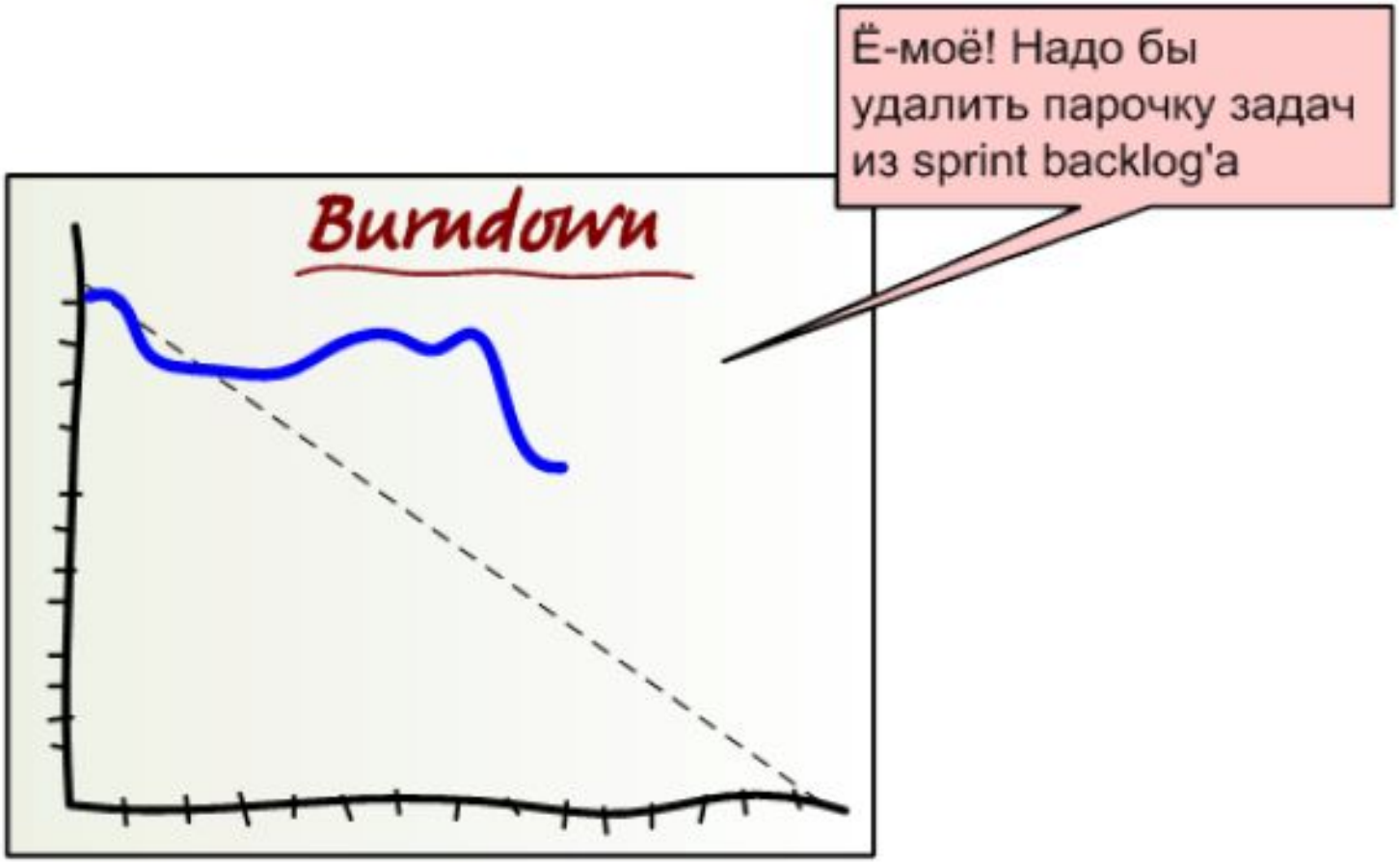
Пример доски задач через пару дней



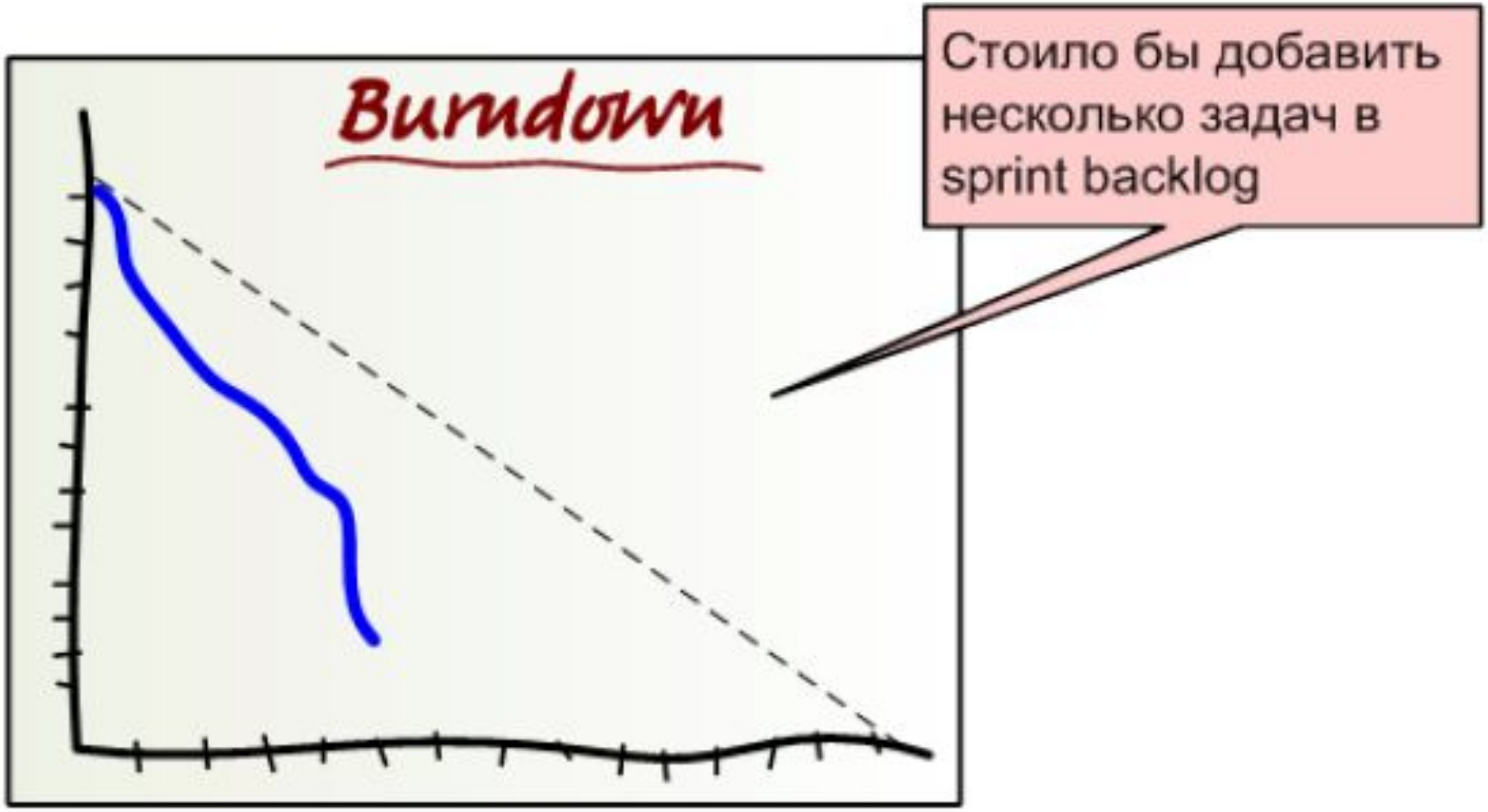
Как работает burndown-диаграмма



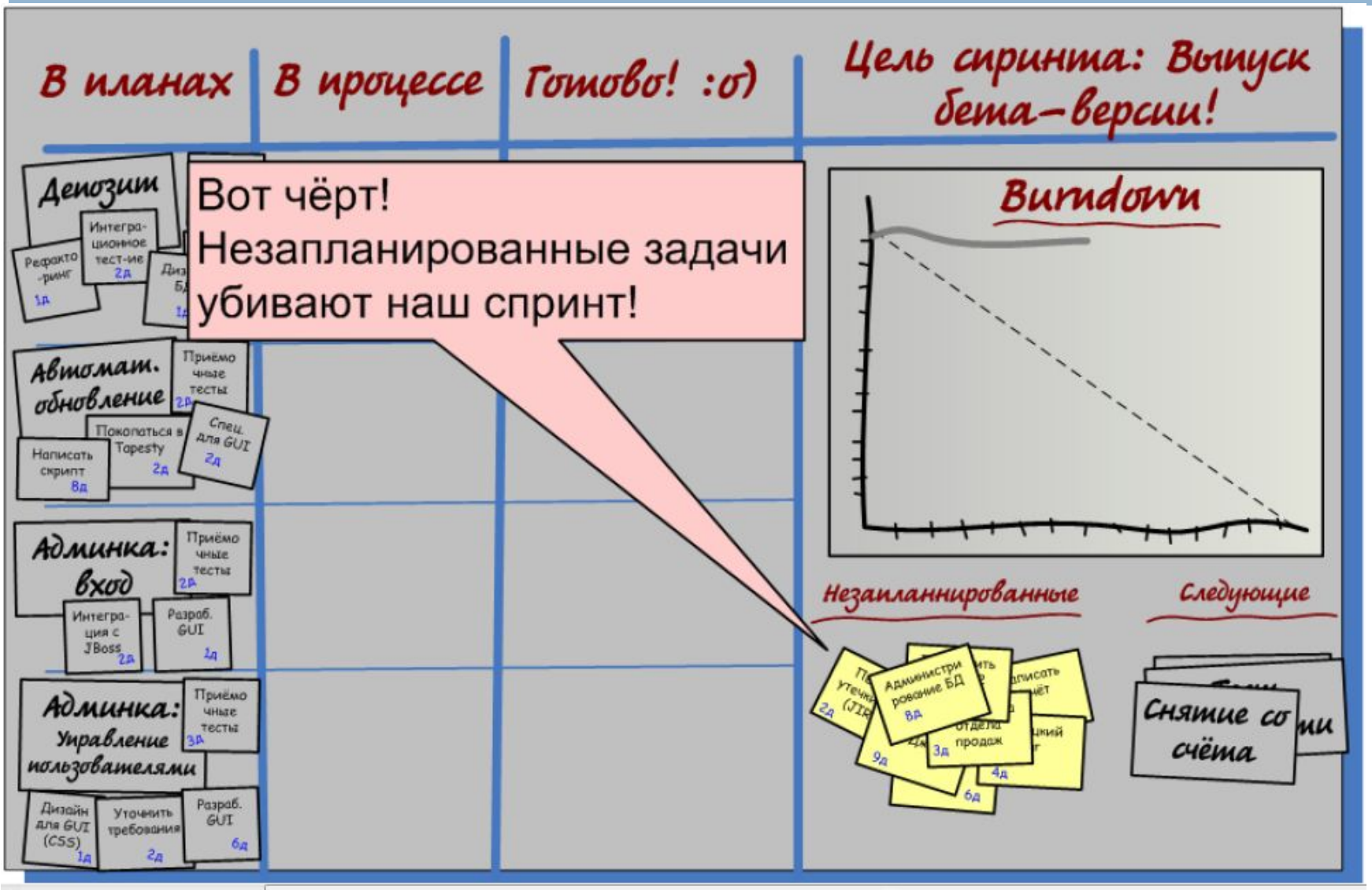
Тревожные сигналы на доске задач



Тревожные сигналы на доске задач



Тревожные сигналы на доске задач

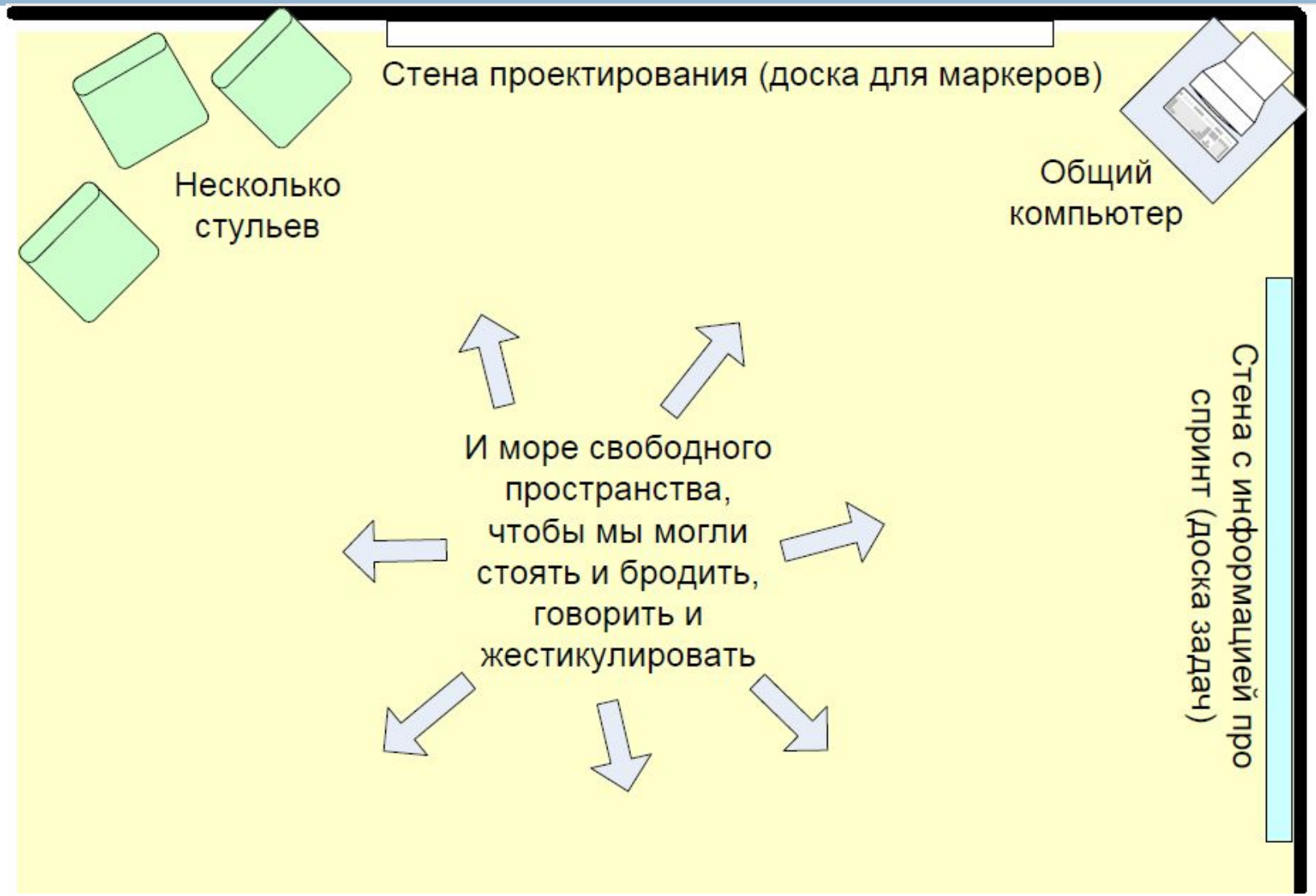


Тревожные сигналы на доске задач

Как отслеживать изменения

- Лучший вариант отслеживания изменений, при данном подходе – это делать фотографию доски задач каждый день.

Как обустроить комнату для команды



Усадите команду вместе

- В пределах слышимости
- В пределах видимости
- Автономно

Не подпускайте product owner'а слишком близко

- Product owner должен находиться настолько близко к команде, чтобы в случае возникновения вопросов, команда могла бы спросить его лично, и чтобы он имел возможность на своих двоих подойти к доске задач. Но он не должен сидеть в одной комнате с командой

Ежедневный Scrum:обновление доски задач

- По мере того, как каждый член команды рассказывает о том, что он сделал за вчерашний день и чем будет заниматься сегодня, он перемещает стикеры на доске задач.
- Как только рассказ касается какого-то незапланированного задания, то для него клеится новый стикер.
- При обновлении временных оценок, на стикере пишется новая оценка, а старая зачеркивается.

Как быть с опоздавшими

- Некоторые команды заводят специальную копилку. Если вы опоздали, даже на минуту, вы кидаете в копилку определённую сумму. Без вариантов. Даже если вы позвонили перед началом ежедневного Scrum'а и предупредили, заплатить всё равно придётся.

Что делать с теми, кто не знает чем себя занять

- Пристыдить
- По старинке
- Моральное давление
- Закабалить

Почему каждый спринт должен оканчиваться демонстрацией

- Положительная оценка работы *воодушевляет команду* .
- Все остальные узнают, чем занимается ваша команда.
- На демо заинтересованные стороны обмениваются жизненно важными отзывами.
- Демо проходит в дружеской атмосфере, поэтому разные команды могут свободно общаться между собой и обсуждать насущные вопросы.
- Проведение демо заставляет команду *действительно доделывать задачи и выпускать их*.

Подготовка к демо

- Постарайтесь как можно более чётко озвучить цель данного спринта.
- Не тратьте много времени на подготовку демо, особенно на создание эффектной презентации.
- Следите, чтобы демо проходило в быстром темпе.
- Пусть ваше демо будет бизнес-ориентированным, забудьте про технические детали.
- Если это возможно, дайте аудитории самой попробовать поиграть с продуктом.
- Не нужно показывать кучу исправлений мелких багов и элементарных фич.

Почему нужно проводить ретроспективы

- Ретроспективы позволяют команде не наступать на одни и те же грабли снова и снова.

Как проводить ретроспективы

- Продолжительность 1-3 часа
- Присутствуют: вся команда и product owner.
- Один из членов команды выступает в качестве секретаря.
- Scrum Master показывает sprint backlog и подводит итоги спринта.
- Серия обсуждений. Каждый говорит, что было плохо и что было хорошо.
- Прогнозируемая производительность сравнивается с фактической.
- Scrum Master обобщает все конкретные предложения по поводу того, что можно улучшить в следующем спринте.

Как учиться на чужих ошибках

- Возможные способы решения проблем, найденных командой на ретроспективе, могут оказаться полезными не только для неё самой, но и для остальных. Как же собрать все эти результаты? Один человек принимает участие во всех ретроспективах в роли "связующего звена".

Типичные проблемы, которые обсуждают на ретроспективах

- Нам надо было больше времени потратить на разбиение историй на подзадачи.
- Очень часто беспокоят извне.
- Мы взяли огромный кусок работы, а закончили только половину.
- У нас в офисе бардак и очень шумно.

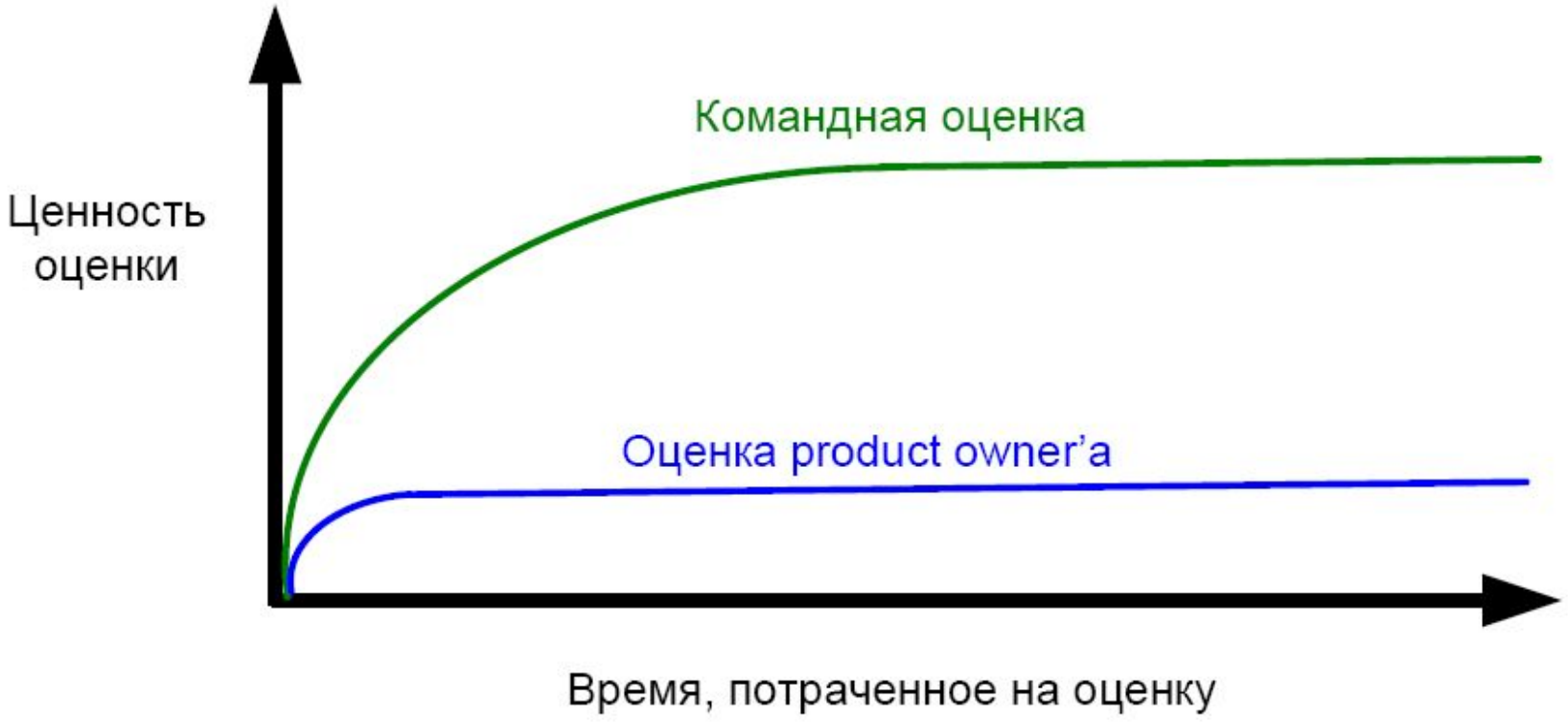
Отдых между спринтами

- В реальной жизни невозможно постоянно бежать как спринтер. Между забегами вам в любом случае нужен отдых. Если вы бежите с постоянной максимальной скоростью, то, по сути, вы просто бежите трусцой.

Планирование релизов: определение приемочной шкалы

Важность	Название
130	Банан
120	Яблоко
115	Апельсин
110	Гуава
100	Груша
95	Изюм
80	Арахис
70	Пончик
60	Лук
40	Грейпфрут
35	Папайя
10	Черника
10	Персик

Оценка наиболее важных историй



Оценка наиболее важных историй

Важность	Название	Оценка
130	Банан	12
120	Яблоко	9
115	Апельсин	20
110	Гуава	8
100	Груша	20
95	Изюм	12
80	Арахис	10
70	Пончик	8
60	Лук	10
40	Грейпфрут	14
35	Папайя	4
10	Черника	
10	Персик	

Прогнозируемая производительность

- Определяем сколько story point-ов может выполнить команда за спринт.

План релиза

Важность	Название	Оценка
		Спринт 1
130	Банан	12
120	Яблоко	9
115	Апельсин	20
		Спринт 2
110	Гуава	8
100	Груша	20
95	Изюм	12
		Спринт 3
80	Арахис	10
70	Пончик	8
60	Лук	10
40	Грейпфрут	14
		Спринт 4
35	Папайя	4
10	Черника	
10	Персик	

Сочетание Scrum и XP

- Scrum решает вопросы управления и организации, тогда как XP специализируется на инженерных практиках.

Парное программирование

- Парное программирование улучшает качество кода.
- Частая смена пар даёт хороший результат.
- Парное программирование действительно способствует распространению знаний внутри команды.
- Ревью кода – хорошая альтернатива парному программированию.
- Не навязывайте парное программирование людям. Вдохновите их, дайте необходимые инструменты и позвольте самим дойти до этого.

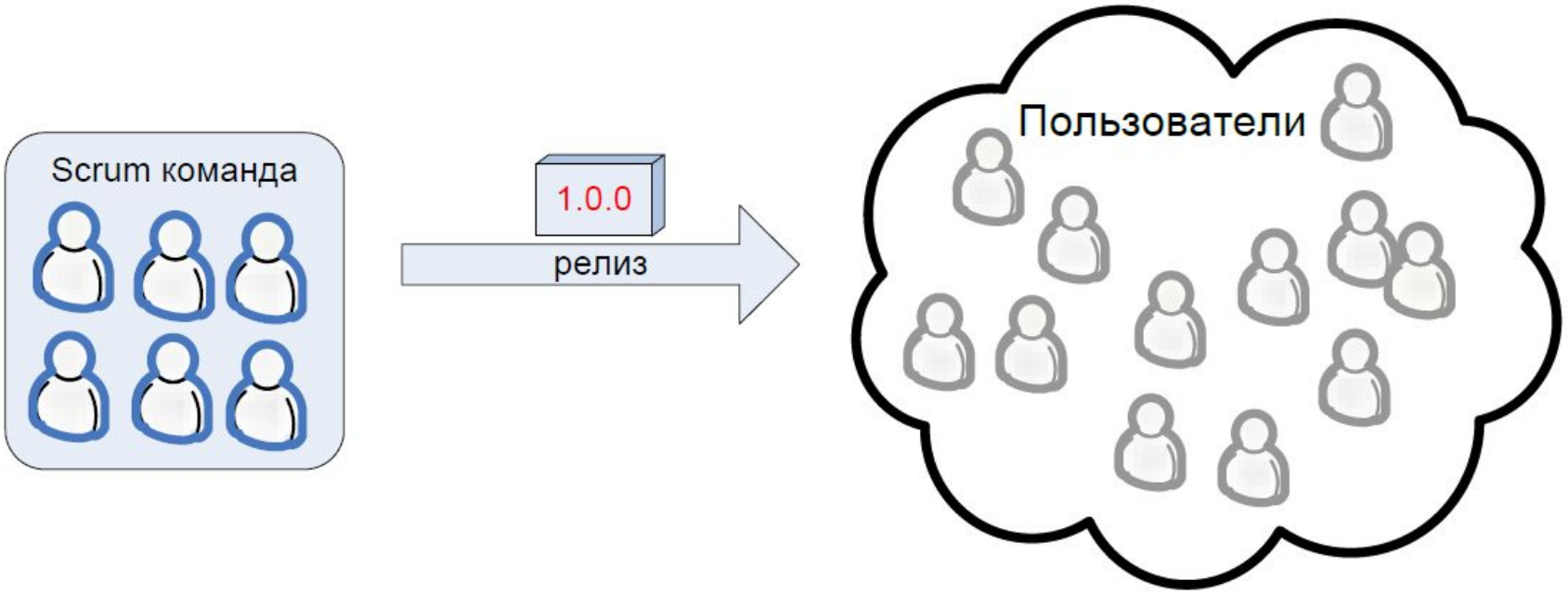
Разработка через тестирование

- Разработка через тестирование – это *непросто*.
- TDD оказывает глубокое положительное влияние на дизайн системы.
- Чтобы TDD стало приносить пользу в новом проекте, необходимо приложить немало усилий.
- Потрать достаточно времени, но сделай так, чтобы писать тесты *было просто*.

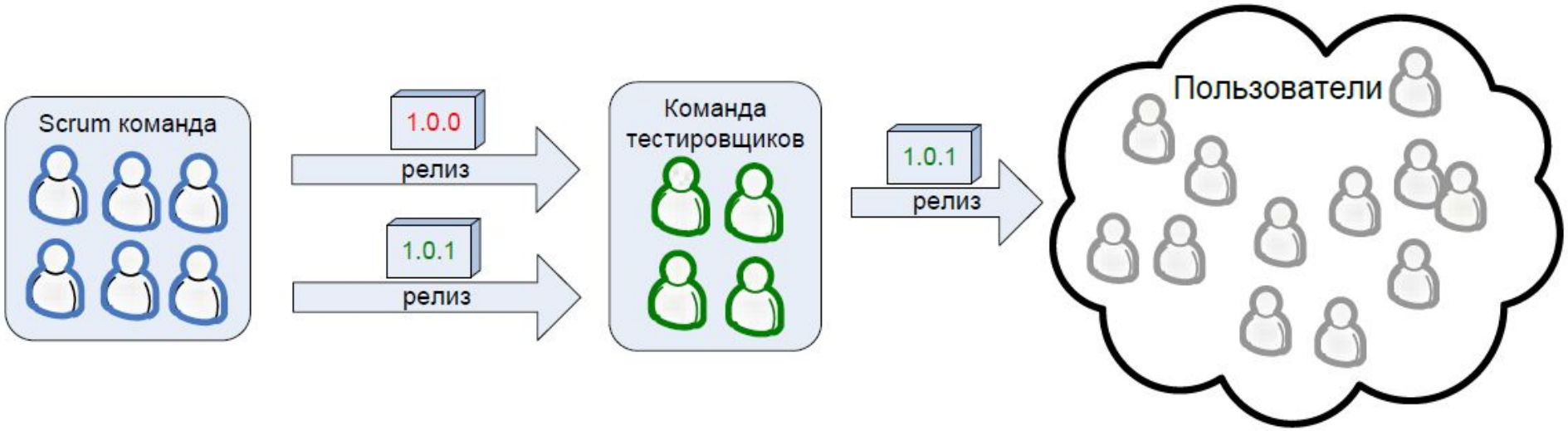
Остальные практики XP

- Эволюционный дизайн
- Непрерывная интеграция (Continuous integration)
- Совместное владение кодом (Collective code ownership)
- Информативное рабочее пространство
- Стандарты кодирования
- Устойчивый темп / энергичная работа

Идеальный вариант



Реальность


Scrum команда
1.0.0
релиз
1.0.1
релиз
Команда тестировщиков
1.0.1
релиз
Пользователи

Минимизация фазы приемочного тестирования

- Максимально улучшить качество исходного кода, создаваемого Scrum-командой.
- Максимально увеличить эффективность ручного тестирования

Повышение качества с помощью тестировщика в команде

Чем занимается тестировщик, когда нечего тестировать?

- Установить и настроить тестовое окружение.
- Уточнить требования.
- Детально обсудить процесс установки.
- Написать документы по установке.
- Пообщаться с подрядчиками.
- Улучшить скрипты автоматизированной сборки.
- Последующее разбиение историй на задачи.
- Собрать ключевые вопросы от разработчиков и проследить, чтобы они не

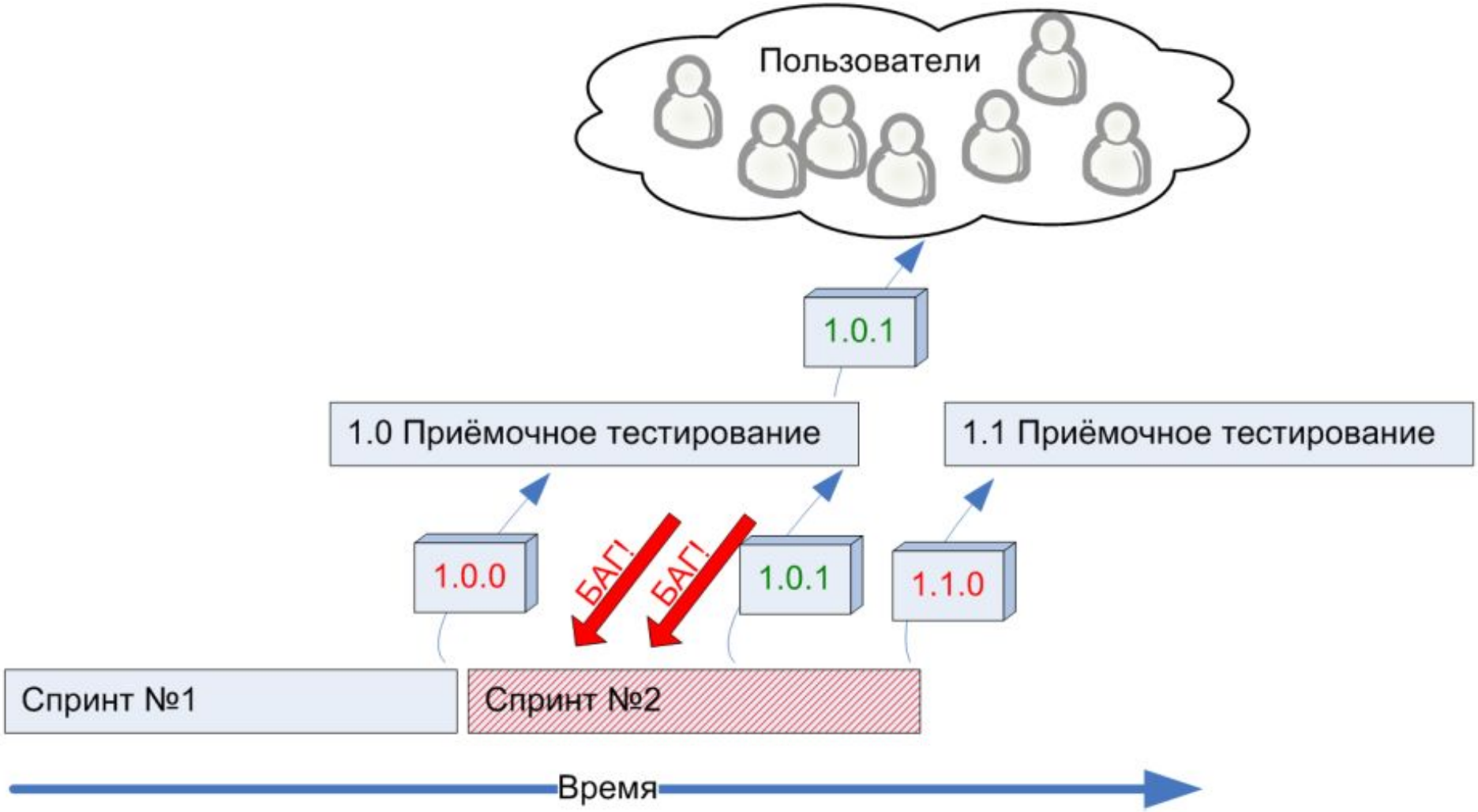
Повышайте качество – делайте меньше за спринт!

- Почти всегда получается дешевле сделать меньше, но качественнее, чем больше, но потом в панике латать дыры.

Стоит ли делать приёмочное тестирование частью спринта?

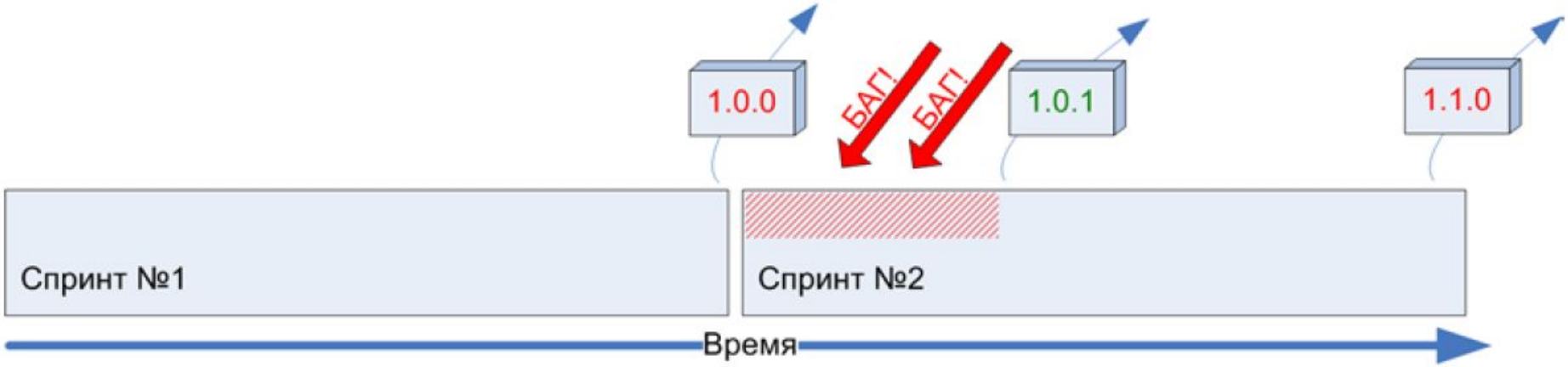
- Спринт ограничен во времени. Приёмочное тестирование (которое включает отладку и повторный выпуск продукта), довольно сложно втиснуть в чёткие временные рамки.
- Если две Scrum-команды работают над одним продуктом, тогда ручное приёмочное тестирование необходимо проводить, собрав результаты работы обеих команд.

Соотношение спринтов и фаз приемочного тестирования



Не начинать новые истории, пока старые не будут готовы к реальному использованию

Начинать реализовывать новые истории, но наивысшим приоритетом ставить доведение старых до ума



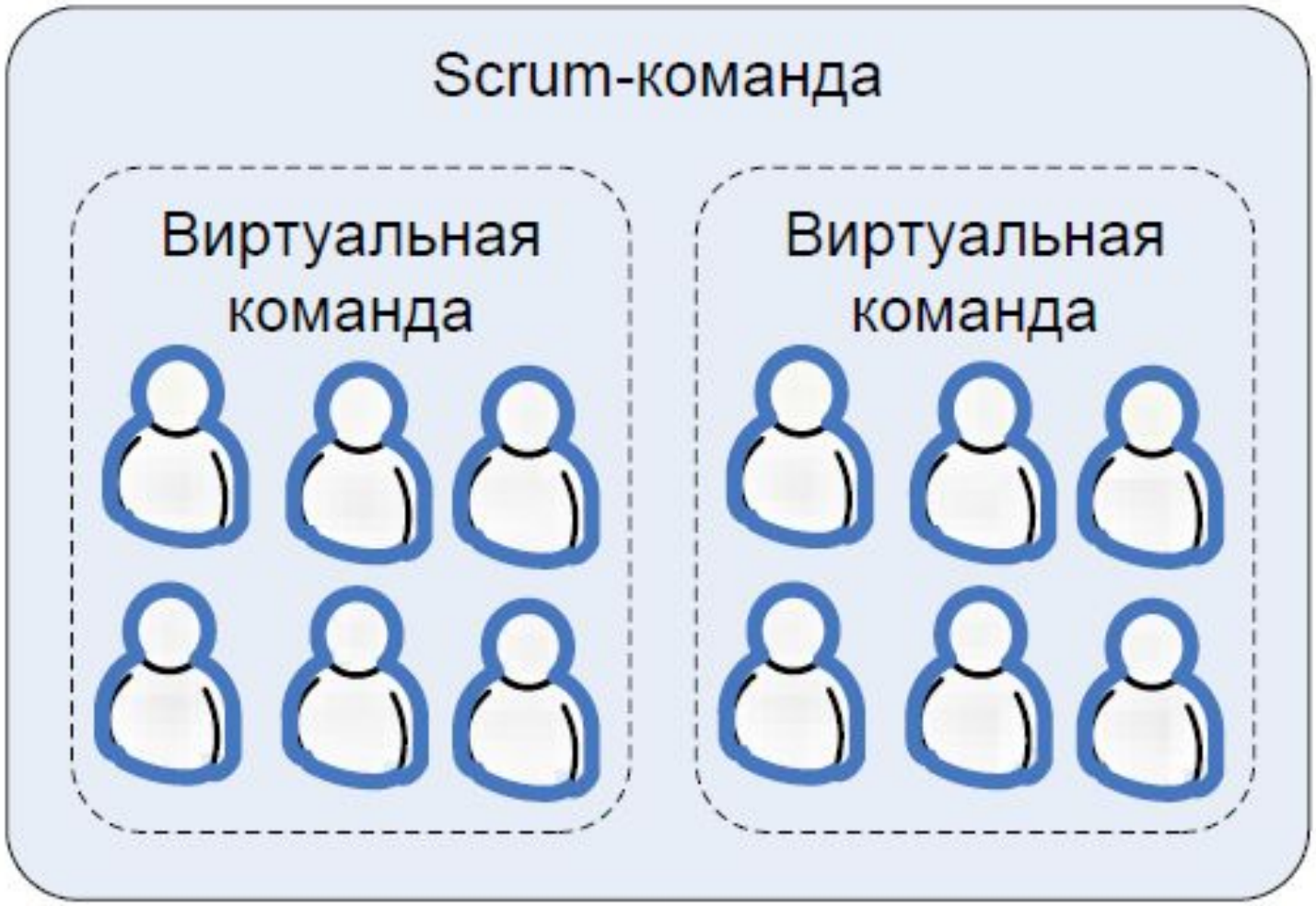
Ограничения системы

- Заставить разработчиков потестировать.
- Внедрить новые инструменты и скрипты, которые упростят тестирование.
- Добавить больше автоматизации.
- Сделать длиннее спринт и включить приемочное тестирование в спринт.
- Выделить несколько "тестовых спринтов", где вся команда будет работать над приемочным тестированием.
- Нанять больше тестировщиков.

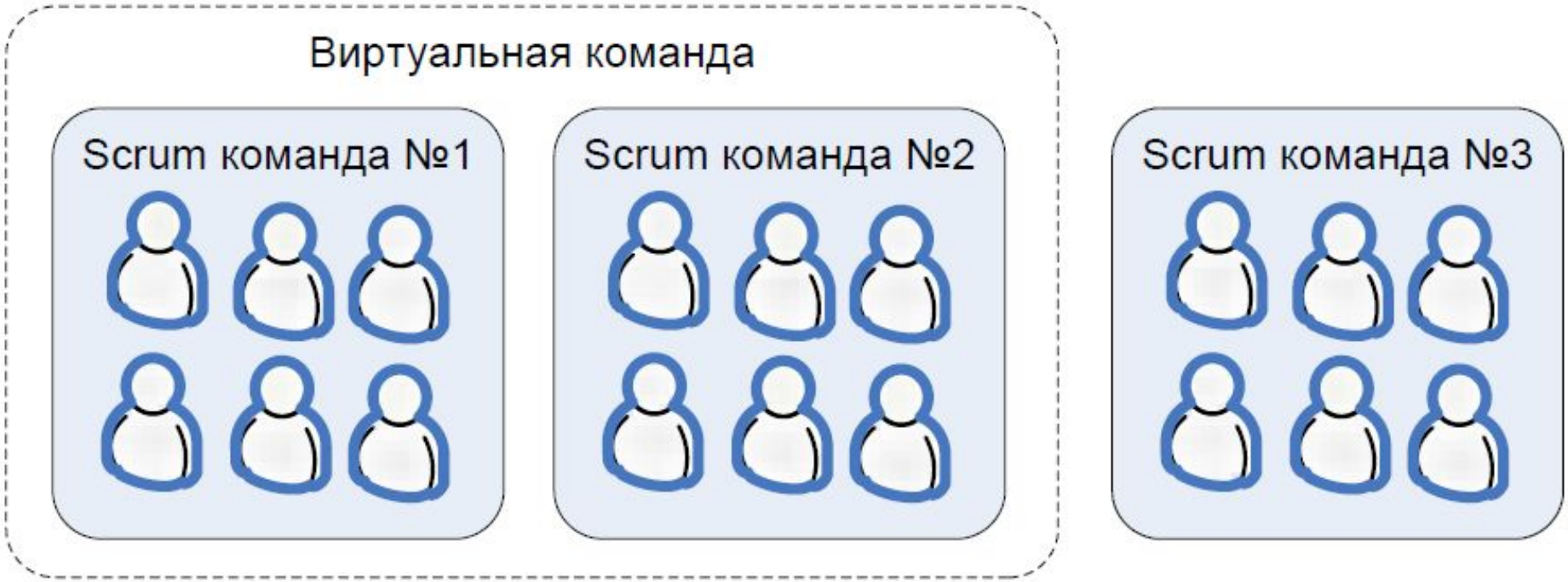
Сколько сформировать команд

- Если настолько сложно работать по Scrum'у с несколькими командами, то зачем вообще заморачиваться? Почему просто не собрать всех в одну команду?

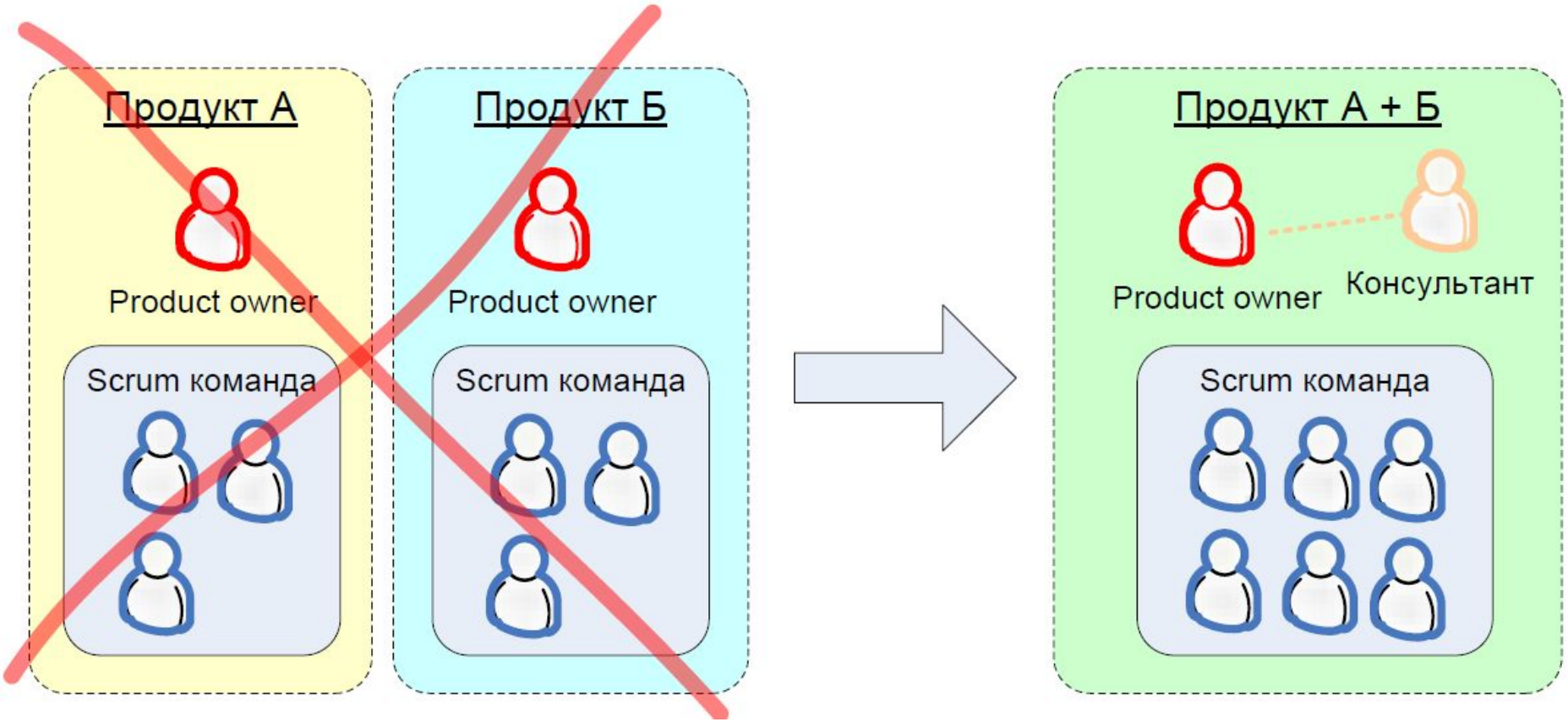
Виртуальные команды



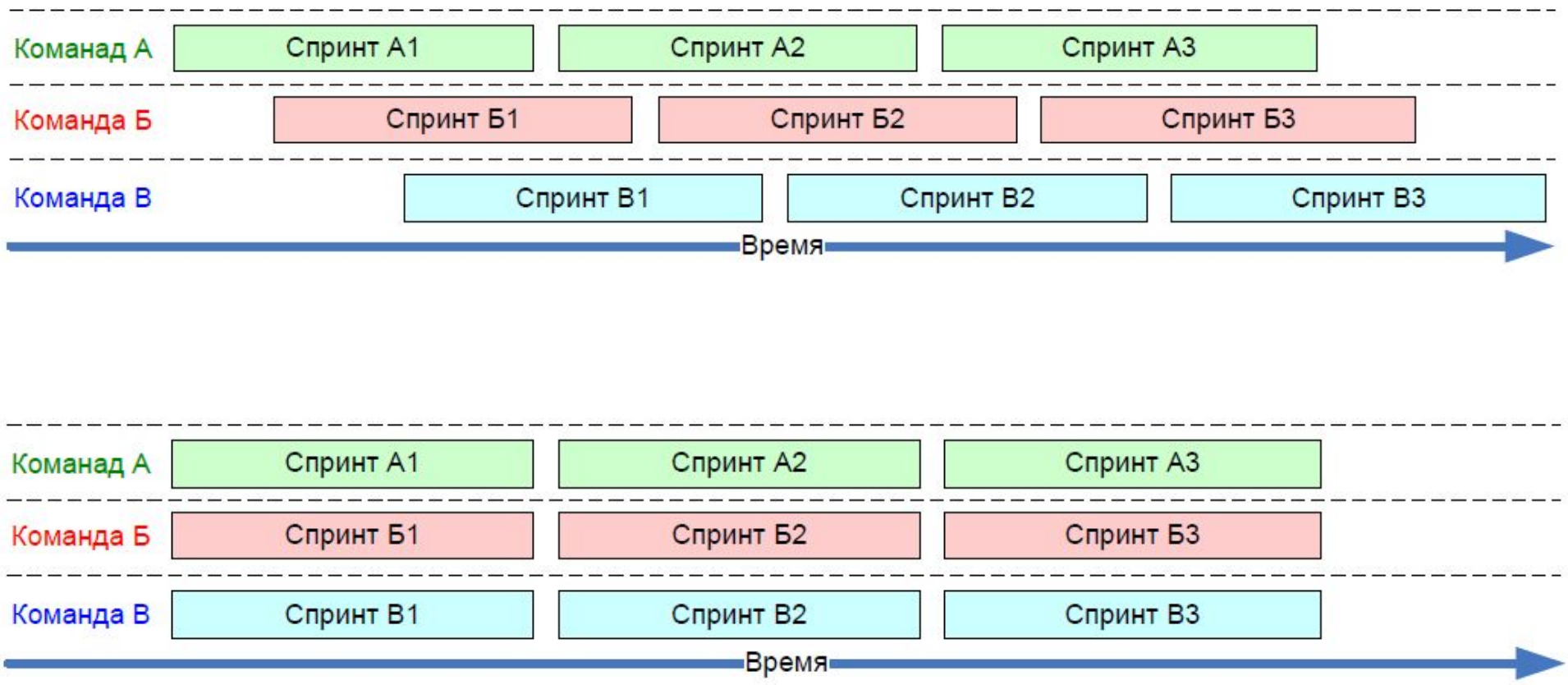
Виртуальные команды



Оптимальный размер команды



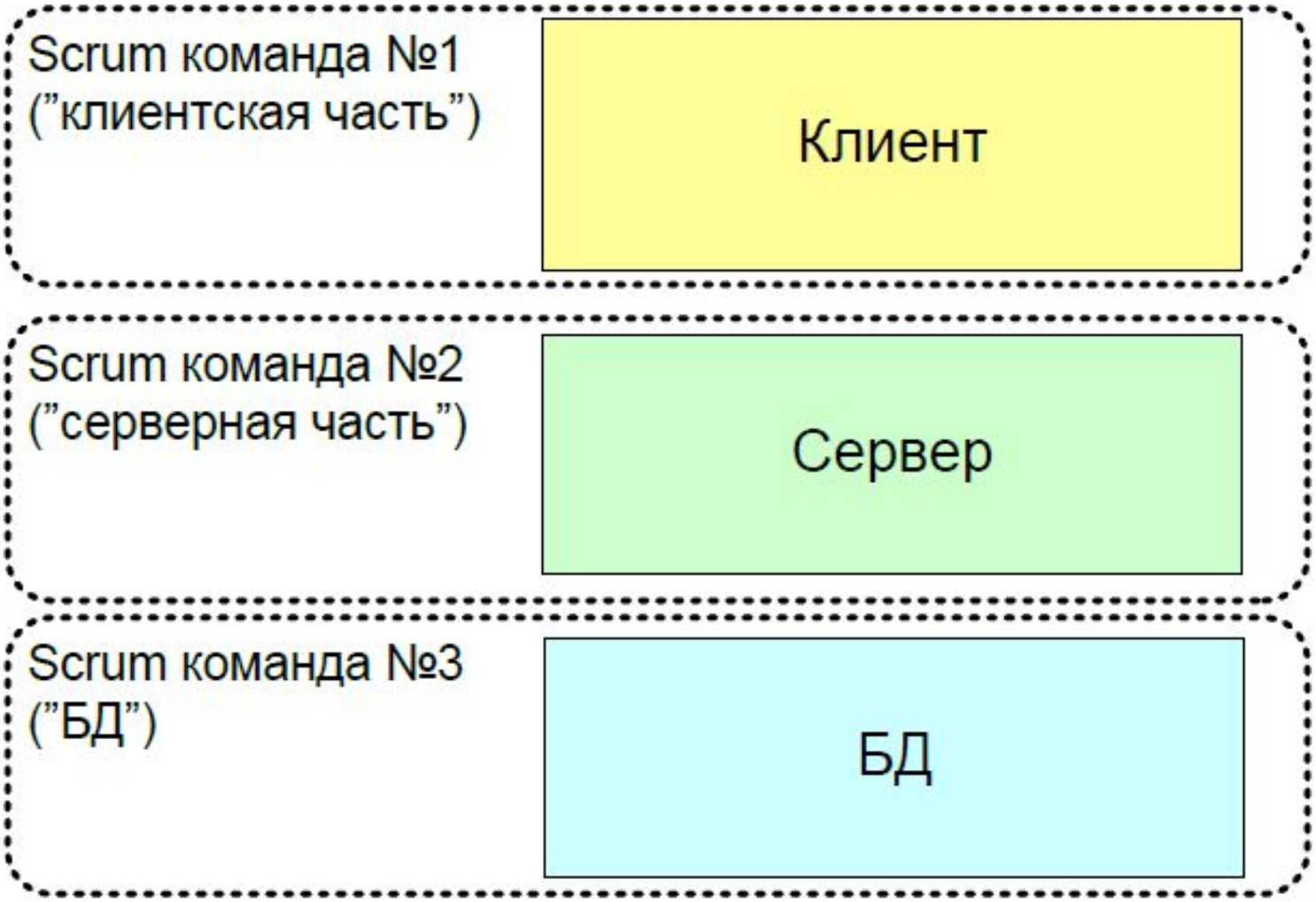
Синхронизировать спринты или нет?



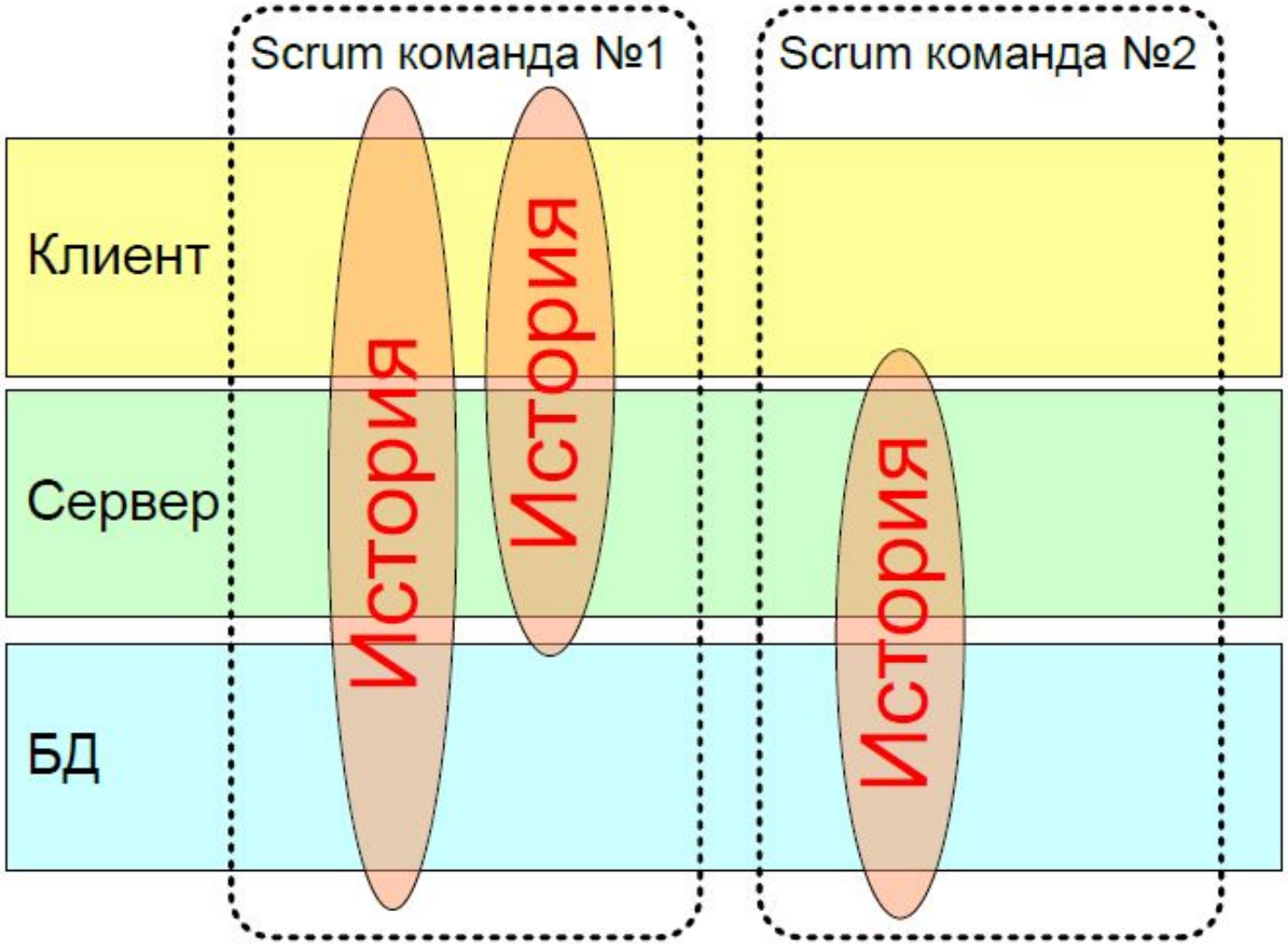
Как распределить людей по командам

- Позволить специально назначенному человеку провести распределение.
- Позволить командам каким-то способом самоорганизоваться.

Команды, специализирующиеся на компонентах



Универсальные команды

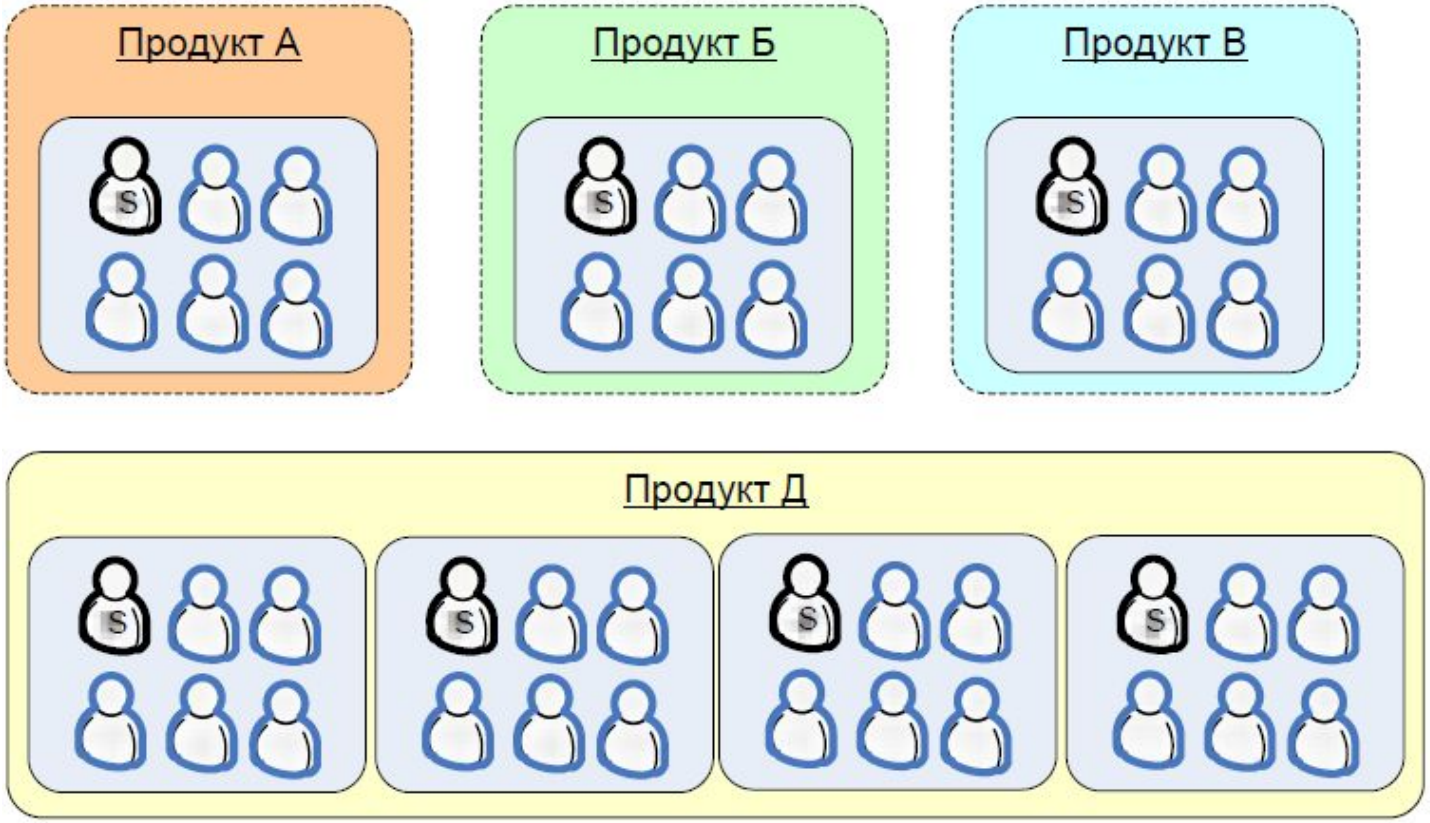


Стоит ли изменять состав команды между спринтами?

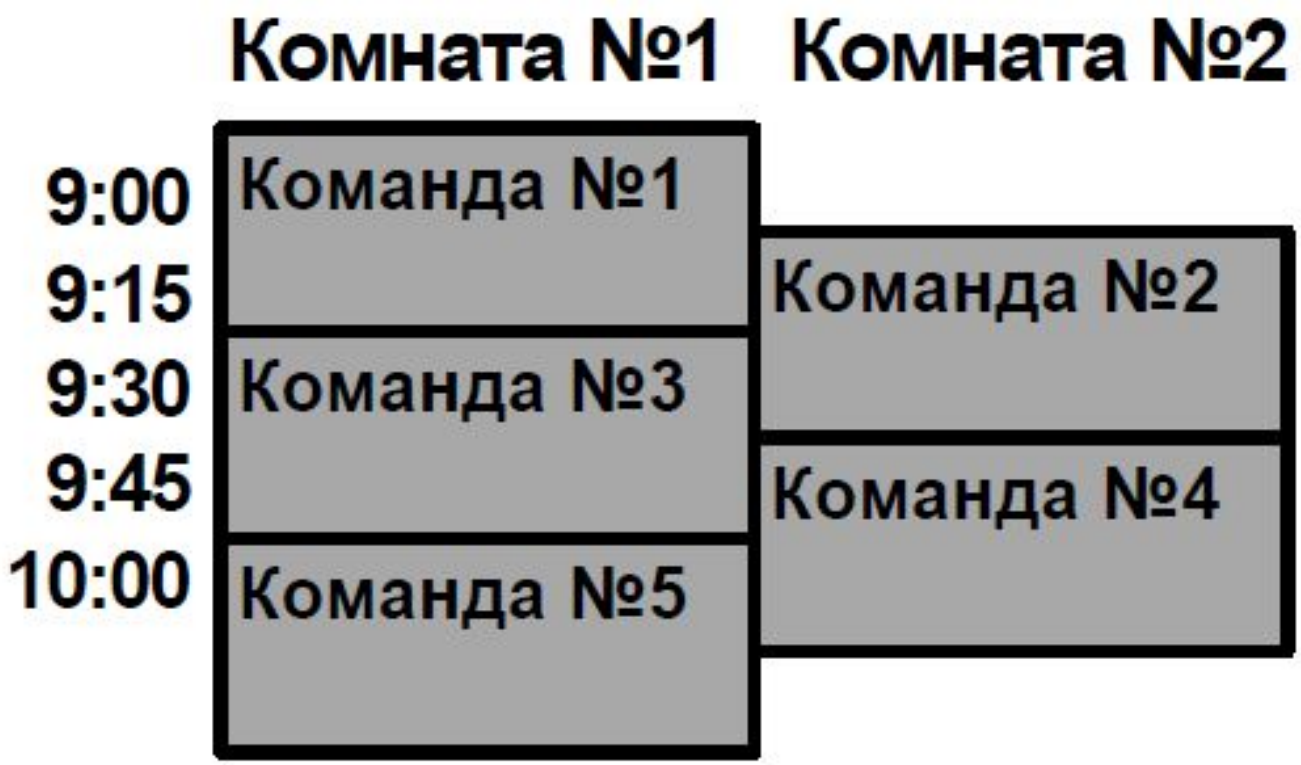
- Если вы решили изменить состав команды, учитывайте все последствия. Будут ли это долговременные или кратковременные изменения? Если кратковременные, стоит их пропустить. На долговременные изменения можно пойти.

Scrum-of-Scrums

- Scrum-of-scrums – это регулярные встречи, цель которых – обсуждение различных вопросов между Scrum-мастерами.



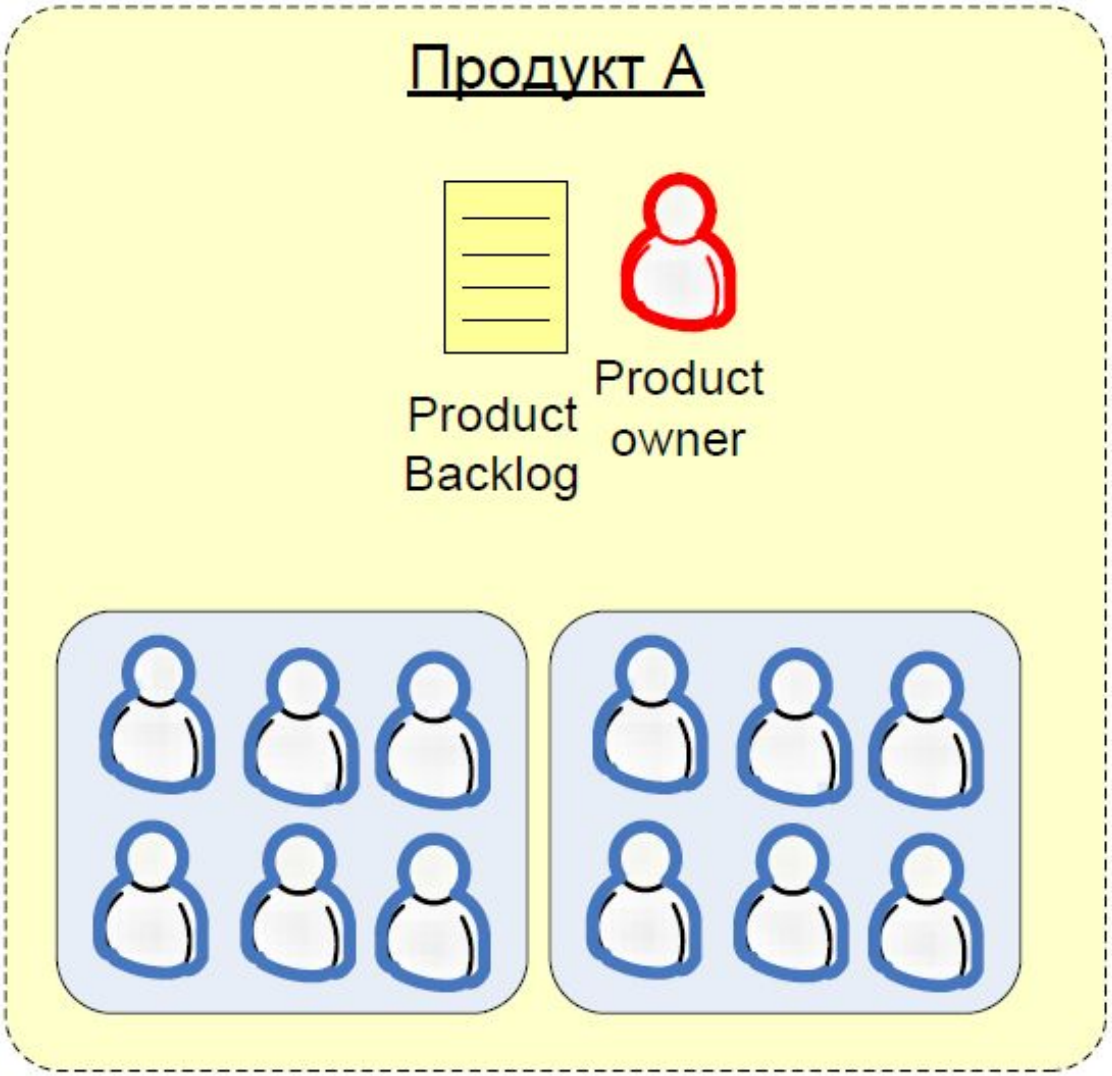
Чередование ежедневных Scrum'ов



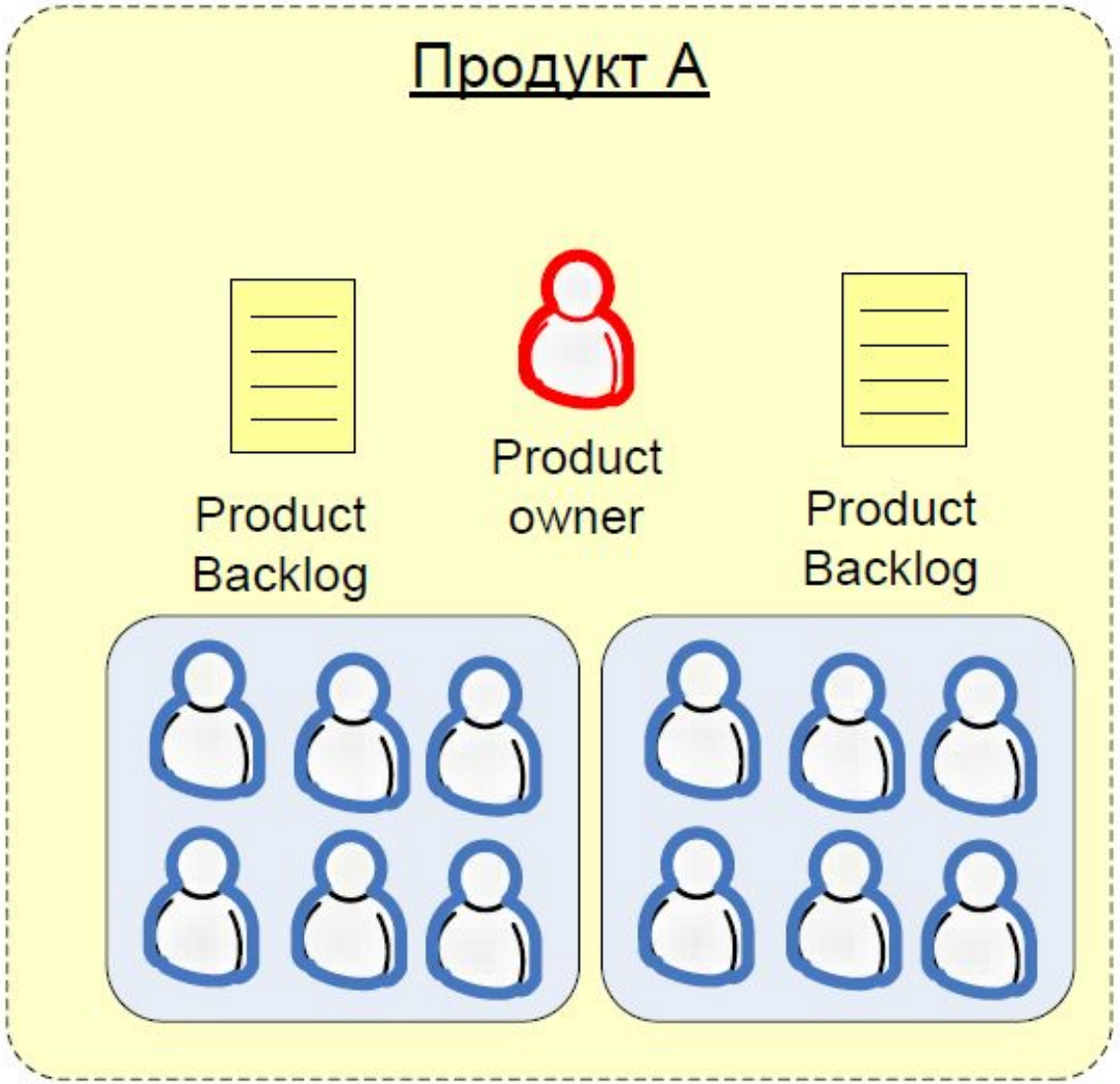
«Пожарные» команды

- Тушить пожары.
- Прикрывать Scrum-команду от всяких раздражителей, вроде неожиданных запросов на изменение функционала, которые непонятно откуда берутся.

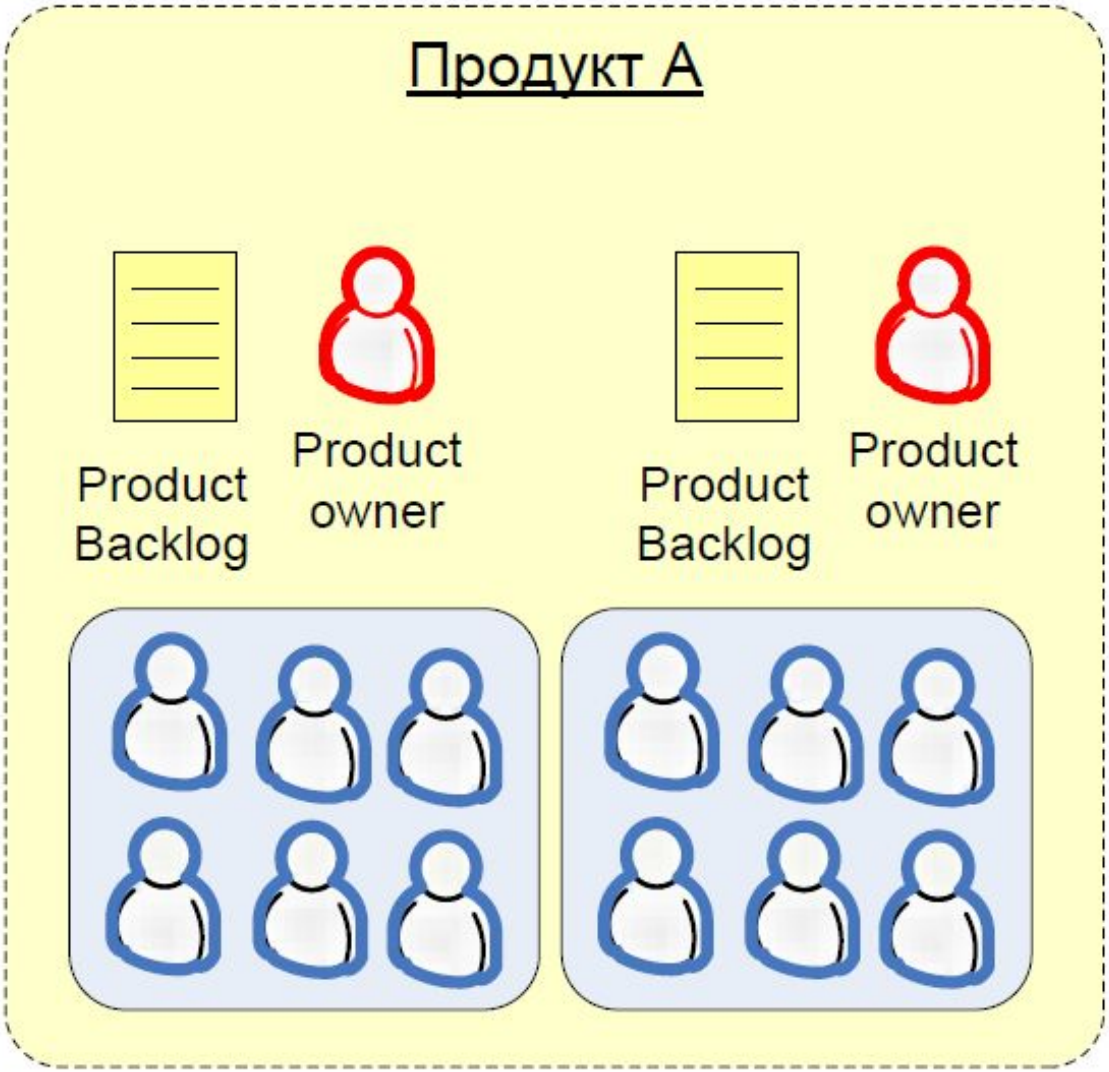
Разбивать product backlog или нет? Один product owner – один backlog



Разбивать product backlog или нет? Один product owner – несколько backlog'ов



Разбивать product backlog или нет? Несколько product owner'ов – несколько backlog'ов

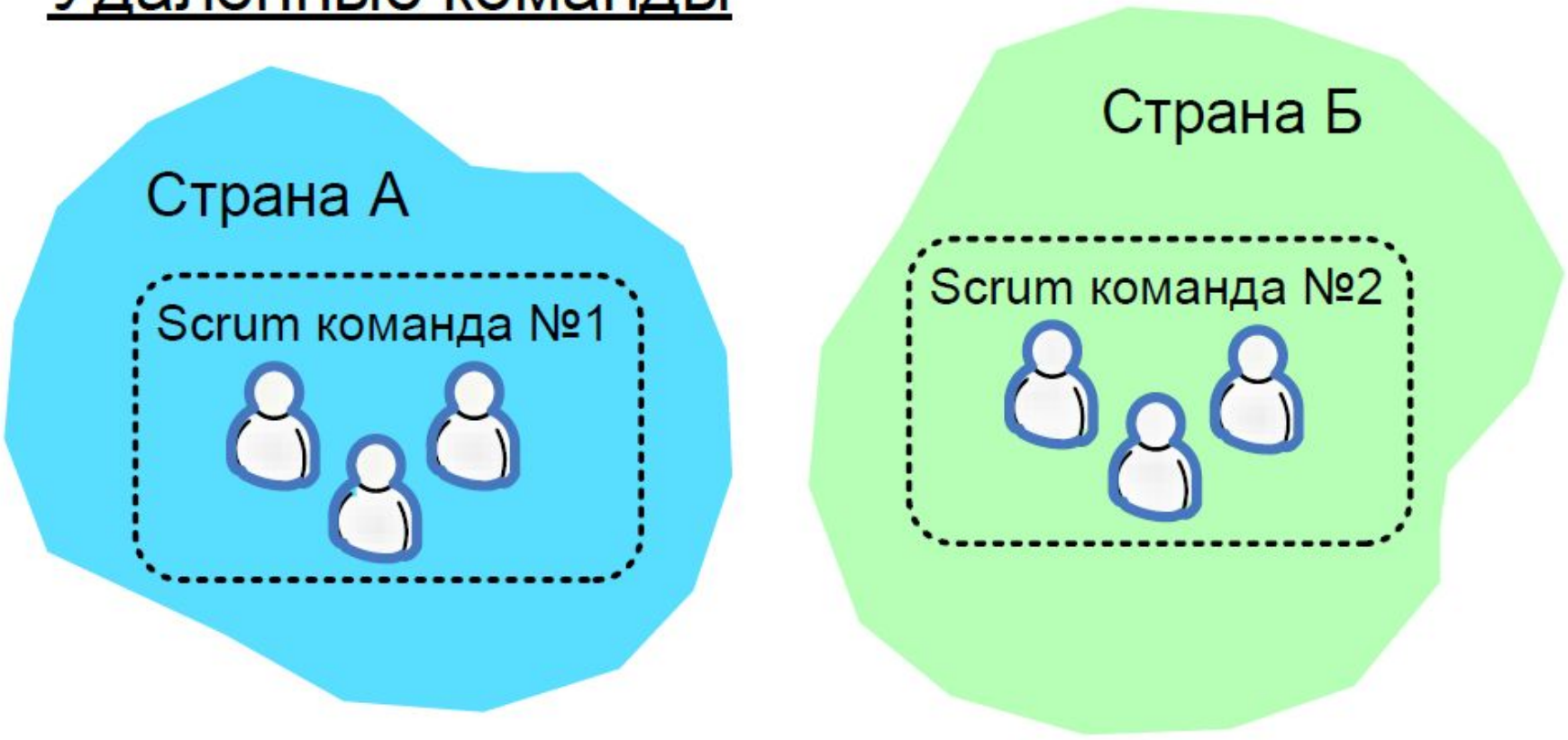


Параллельная работа с кодом

- Всегда поддерживайте основную ветку проекта в рабочем состоянии.
- Помечайте каждый релиз тэгом.
- Создавайте новые ветки кода только тогда, когда это действительно необходимо.
- Используйте ветвление для разделения кода *разных стадий разработки*.
- Чаще синхронизируйтесь.

Управление географически распределенными командами

Удалённые команды



Управление географически распределенными командами

Удалённые участники команд