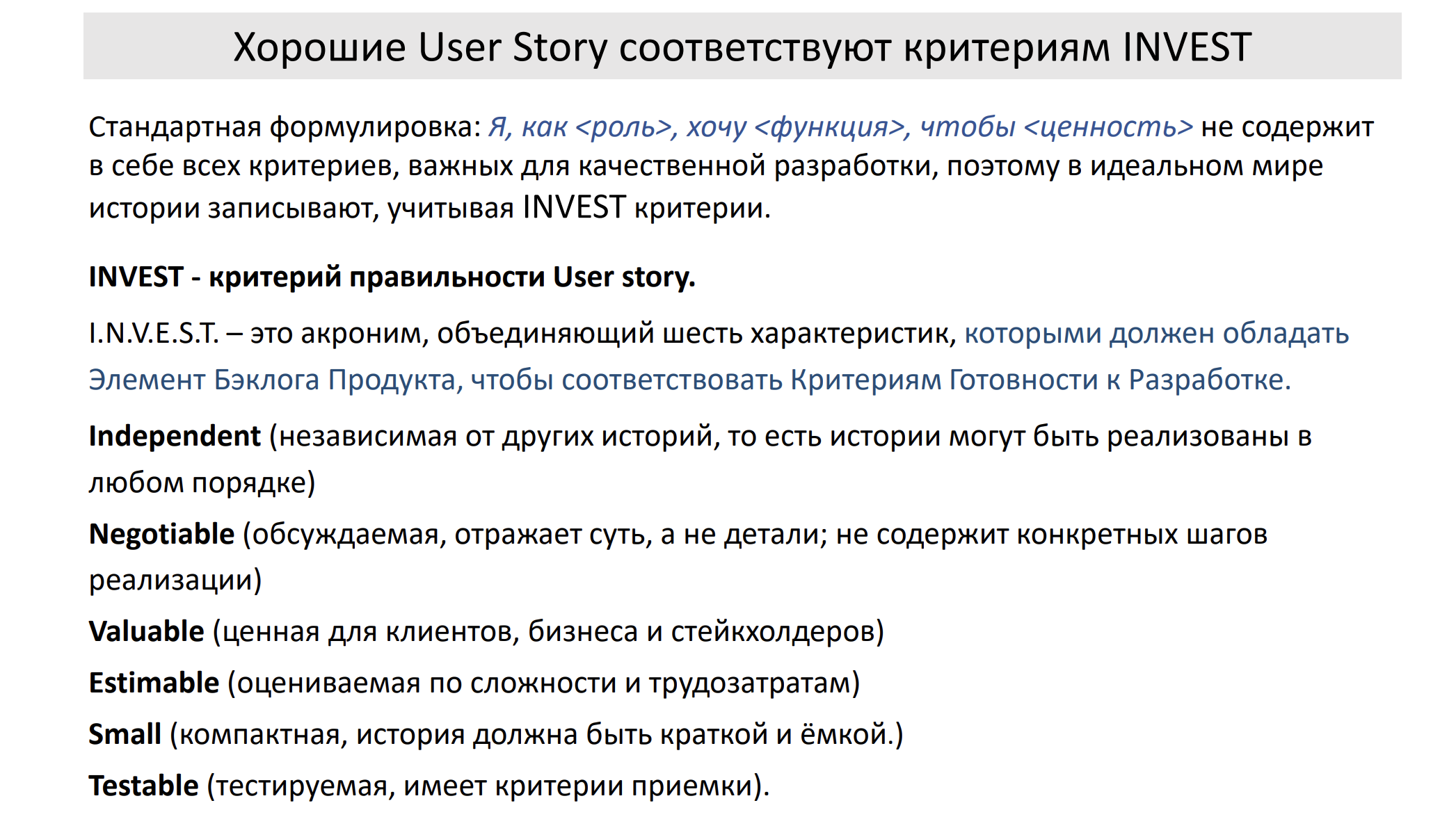
Вопросы:

1. Что означает акроним INVEST? Расскажите подробнее про эти правила.



Акроним **INVEST** используется для описания качеств хорошо сформулированной пользовательской истории или фичи в Agile. Он состоит из следующих компонентов:

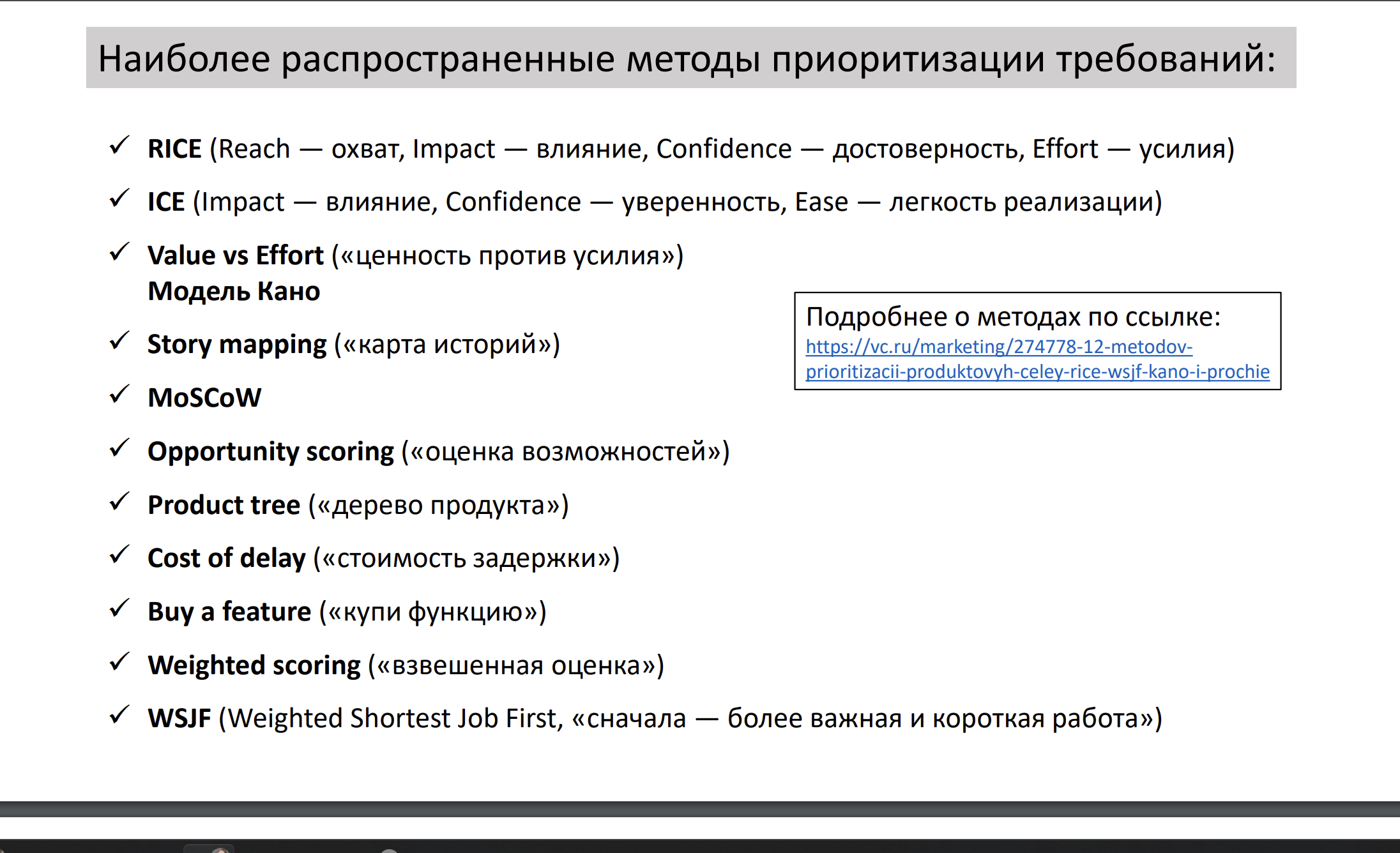
* **I – Independent (Независимый):**  
  История должна быть независимой от других историй, чтобы её можно было планировать, разрабатывать и тестировать отдельно, без тесных связей с другими задачами.
* **N – Negotiable (Поддающийся обсуждению):**  
  История не является жестко зафиксированным требованием. Это своего рода отправная точка, которую можно обсудить и изменить в процессе работы, учитывая потребности заказчика и командные соображения.
* **V – Valuable (Ценный):**  
  Каждая история должна приносить реальную ценность пользователю или бизнесу. Если история не добавляет ценности, её смысл ставится под сомнение.
* **E – Estimable (Оцениваемый):**  
  Историю должно быть можно оценить по времени и усилиям, необходимым для её реализации. Это помогает в планировании спринтов и расстановке приоритетов.
* **S – Small (Небольшой):**  
  История должна быть достаточно маленькой, чтобы её можно было выполнить в рамках одного итерационного цикла. Если задача слишком велика, её разбивают на более мелкие, легко управляемые части.
* **T – Testable (Тестируемый):**  
  Должны быть определены критерии приёмки, по которым можно проверить, что история реализована корректно и соответствует требованиям.

1. Для чего используется техника User Story Mapping?

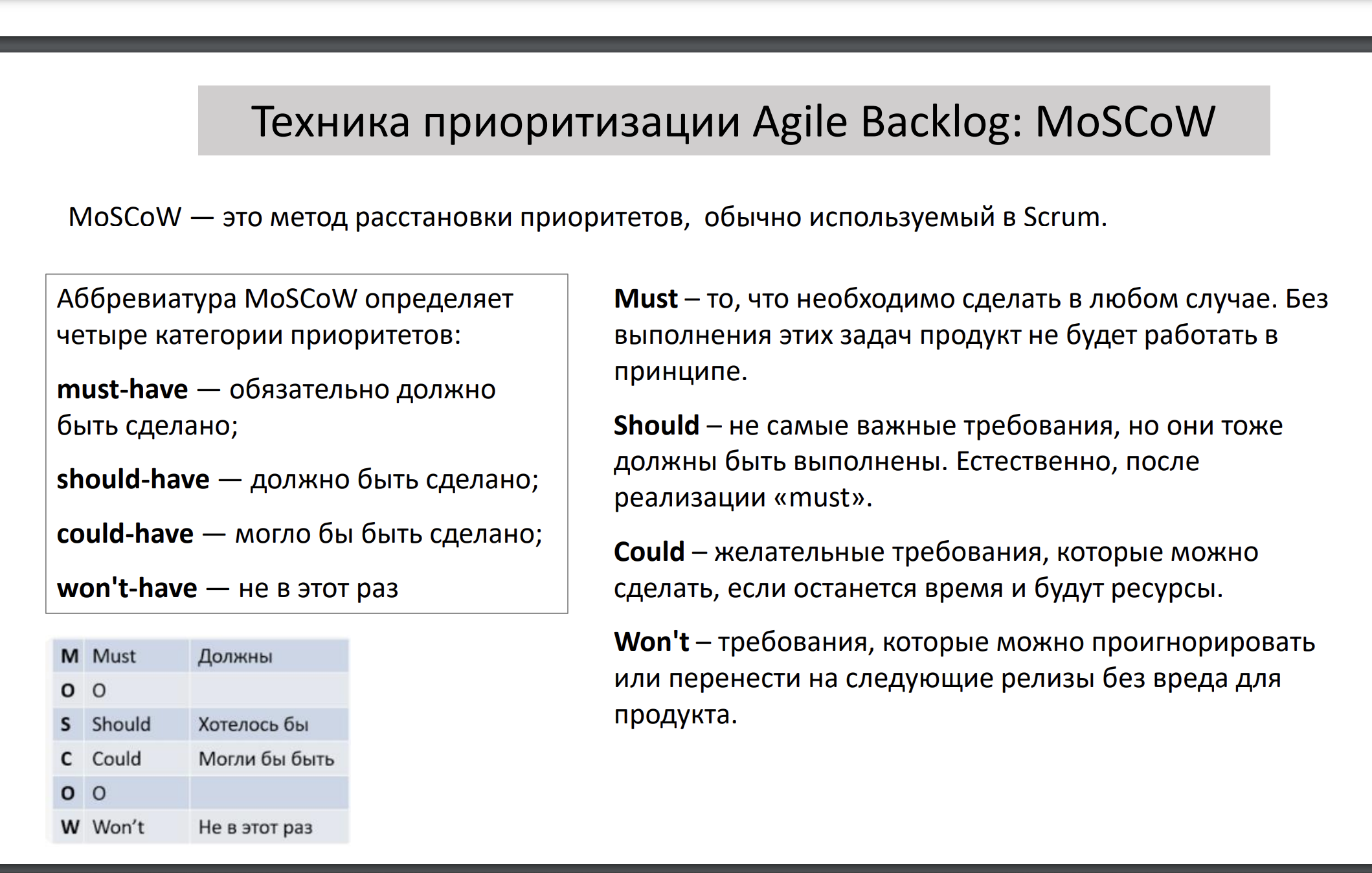


**User Story Mapping** – это техника, используемая для визуального представления пользовательских историй (User Stories) в виде структуры, которая помогает командам **разрабатывать продукт, ориентируясь на пользовательский опыт и бизнес-ценность**.

Разделяет функциональность на **"обязательные"** и **"желательные"** фичи.

1. Расскажите про методы приоритизации требований.
2. 





**Методы приоритизации требований**

Приоритизация требований – это процесс определения **важности и последовательности** реализации функциональности, который помогает команде **фокусироваться на наиболее ценных задачах**.

**Популярные методы приоритизации**

**1. MoSCoW (Must, Should, Could, Won’t)**

Этот метод делит требования на 4 группы:

* **Must have** – обязательные функции, без которых продукт **не может работать**.
* **Should have** – важные, но **не критичные** фичи (их можно отложить).
* **Could have** – желательные функции, которые **улучшат продукт**, но не являются приоритетными.
* **Won’t have** – функции, которые **не будут включены** в данный релиз.

📌 **Пример:**  
Разрабатываем мобильное приложение для доставки еды:  
✅ **Must have** – заказ еды, оплата онлайн.  
✅ **Should have** – отслеживание доставки.  
✅ **Could have** – рекомендации на основе истории заказов.  
✅ **Won’t have** – встроенный чат с курьером.

**2. Метод 100 долларов**

Каждому участнику команды **выдается 100 долларов**, которые он может распределить между требованиями.

* Требования с **наибольшим количеством инвестиций** получают **высший приоритет**.
* Позволяет учитывать мнение всех стейкхолдеров.

📌 **Пример:**  
✅ Оплата картой – 40$  
✅ Поиск по блюдам – 30$  
✅ Программа лояльности – 20$  
✅ Чат с поддержкой – 10$

**3. Метод Kano (Кано)**

Определяет **удовлетворение клиента** от реализации требований:

* **Базовые (Basic Needs)** – функции, которые **обязательны** (если их нет, продукт бесполезен).
* **Ожидаемые (Performance Needs)** – чем лучше они работают, тем выше удовлетворенность.
* **Вау-функции (Delighters)** – неожиданные, но приятные (могут стать конкурентным преимуществом).

📌 **Пример:**

* **Базовые** – оформление заказа, оплата.
* **Ожидаемые** – отслеживание заказа, отзывы.
* **Вау-функции** – бонусы за первый заказ, AI-рекомендации.

**4. Метод RICE (Reach, Impact, Confidence, Effort)**

Каждое требование оценивается по 4 параметрам:

* **Reach (Охват)** – сколько пользователей это затронет?
* **Impact (Влияние)** – насколько сильно это улучшит продукт?
* **Confidence (Уверенность)** – насколько уверены в результате?
* **Effort (Затраты)** – сколько ресурсов потребуется?

**Формула:**  
📌 RICE Score = (Reach × Impact × Confidence) / Effort

Чем **выше RICE-оценка**, тем приоритетнее задача.

📌 **Пример:**  
✅ Новый способ оплаты – **(100×3×0.9) / 10 = 27**  
✅ Улучшение UX кнопки – **(500×1×0.8) / 5 = 80**  
✅ AI-рекомендации – **(200×2×0.7) / 20 = 14**

🔹 Выбираем **улучшение UX**, так как у него **высший приоритет**.

**5. Метод WSJF (Weighted Shortest Job First)**

Используется в **SAFe (Scaled Agile Framework)**.

**Формула:**  
📌 WSJF = (Business Value + Time Criticality + Risk Reduction) / Job Size

* **Business Value (Бизнес-ценность)** – насколько важно для бизнеса?
* **Time Criticality (Критичность во времени)** – как быстро нужно реализовать?
* **Risk Reduction (Снижение рисков)** – насколько уменьшает риски?
* **Job Size (Размер задачи)** – сложность выполнения.

**Чем выше WSJF, тем быстрее нужно сделать фичу**.

📌 **Пример:**  
✅ Добавить push-уведомления: **(9+6+5) / 3 = 6.7**  
✅ Интеграция с PayPal: **(8+5+4) / 5 = 3.4**  
✅ Оптимизация кода: **(4+3+2) / 8 = 1.1**

**Выбираем push-уведомления, так как WSJF выше**.