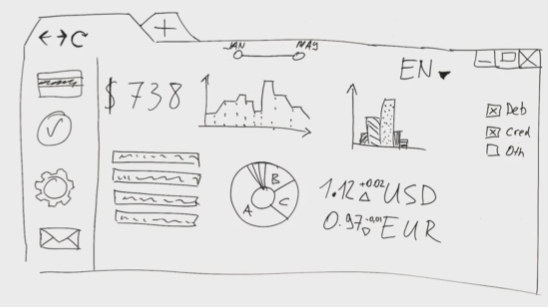
**Test Strategy**

**В первую очередь определяем:**

1. Что нам необходимо тестировать?
2. Какие временные рамки на тестирование?



**Тестовая стратегия отвечает на 3 вопроса:**

1. Какие тестовые типы будут в рамках тестирования?
2. Как мы будем выполнять эти типы тестирования?
3. Что будет выполняться в рамках определенных фаз?

**Как понять какие типы тестирования нам важны?**

Рассмотрим приложение интернет-банкинга.



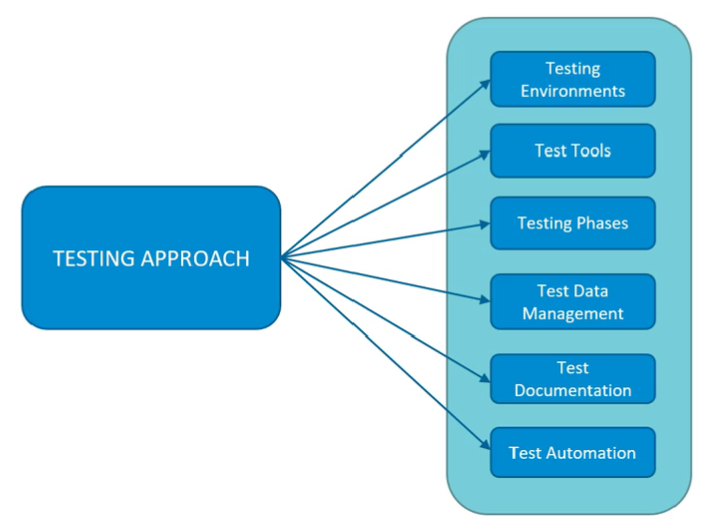
1. Функциональность, которую предоставляет нам это приложение. **Functional testing**
2. Какие данные мы будем видеть**? Integration testing**
3. Как приложение будет работать под требуемыми устройствами? **Compatibility testing**
4. Как будет работать приложение во время стандартной функциональности телефона? (звонок) Как будет работать под различными типами соединениями? **Mobile testing**.
5. На сколько наше приложение будет защищено**? Penetration testing**
6. На сколько быстро будет работать приложение? **Performance testing**
7. Комфорт использования приложения людей с ограниченными возможности. **Accessibility testing**
8. Смена языка, времени и т.д. **Localization testing**
9. Удобство пользования приложением. **Usability testing**

С этим списком идем к заказчику и он говорит, что **ДОЛЖНЫ** выполнять **МЫ**.

**Как же нам организовать более эффективно и что важно при выполнении различных типов тестирования?**

1. *Комбинированные типы* (Regression & Localization & Mobile & Compatibility & Permissions)
2. Различные *типы тестирования* требуют *разных* людей и бюджет (Performance, security, usability)
3. Есть *типы тестирования*, требующие доп *знаний* (Accessibility, Localization, Mobile testing)
4. Какие типы тестирования когда будут выполняться? ()

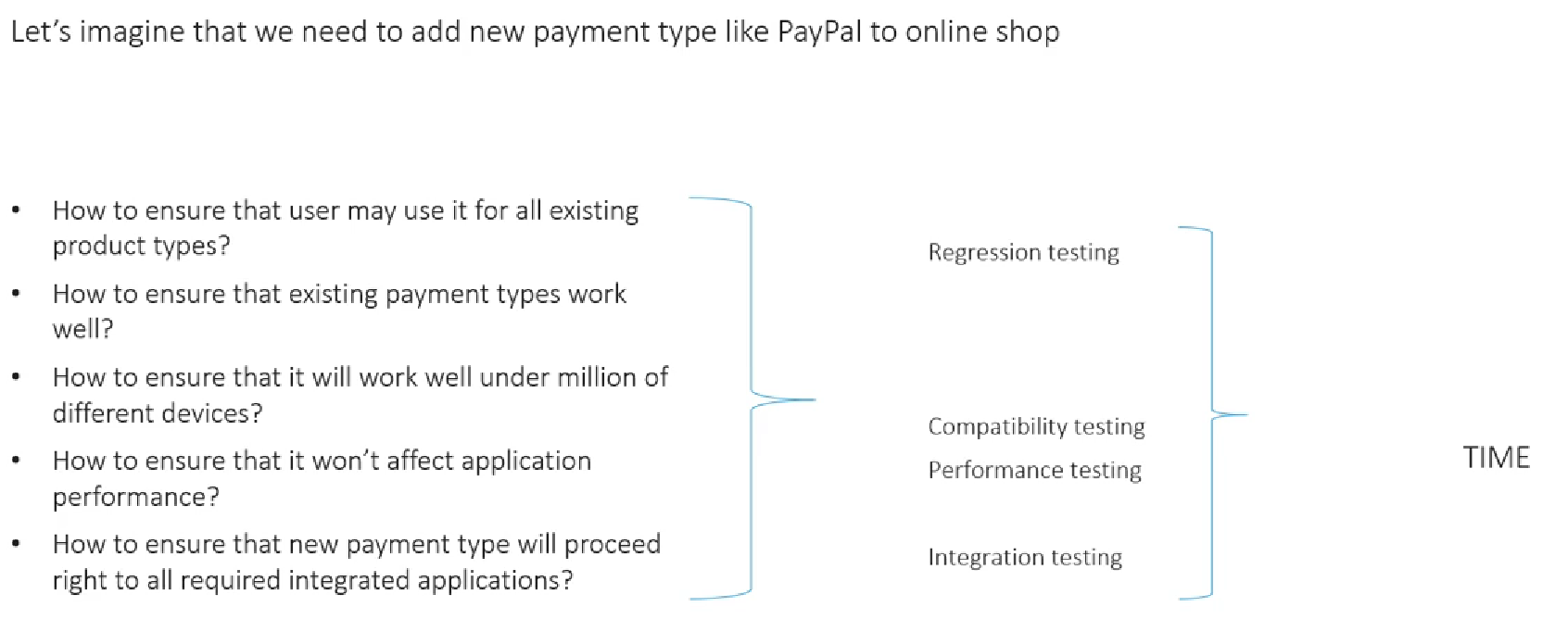
**Также важно выяснить для типов тестирования свои особенные наборы и подходы:**

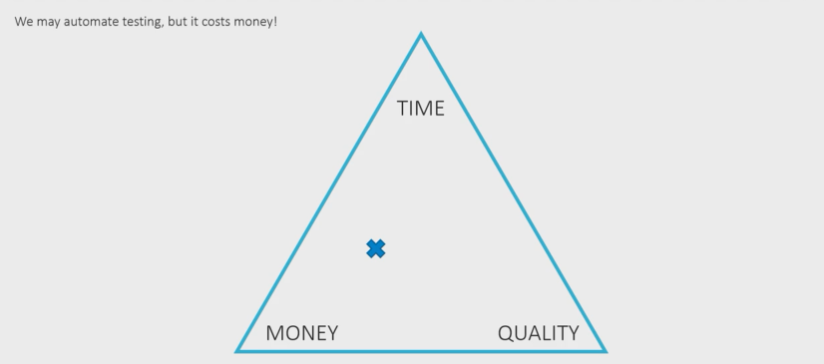


1. **Testing Environments**. Среда, в которой тестирование, какие требования.
2. **Test Tools.** Какие инструменты необходимы для тестирования (эскизы, заглушки)
3. **Testing Phases.** Кто и когда что выполняет?
4. **Test Data Management**.
5. **Test Documentation**. Какой тип документации подходит больше.
6. **Test Automation.** Что и как тестировать, где действительно автоматизация будет приносить пользу.

**Agile & Scrum-Fall**

**Как время влияет в целом на разработку тестовой стратегии и на подход к разработке приложения?**





**Исходя из того, что необходимо найти баланс между 3 факторами, появилось 2 течения:**

**Agile**

После каждого спринта можем предоставлять новую функциональность пользователям. Взаимодействие между девелоперами и тестировщиками



**Главное различие:**

**Traditional Testing**

Тестировщик в первую очередь должен найти расхождение между тем, что он видит в разработанном продукте и тем, что требует.

**Agile Testing**

Здесь тестировщик должен предупредить, о возможном расхождении результатов и требований.

**Как это сделать?**

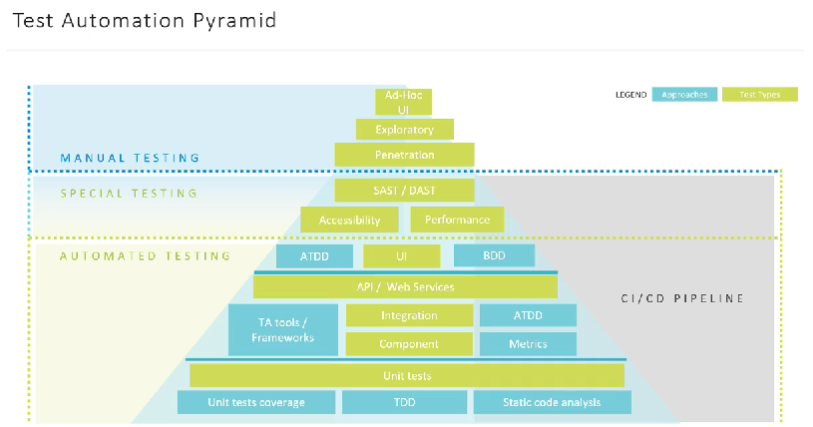
1. Начать тестирование этих требований до того, как приступят к разработке этих требований.
2. Активное взаимодействие с разработчиками:

* как можно раньше предоставлять инфу о том, что мы собираемся тестировать,
* договориться кто и на каком уровне будет создавать тесты,
* активно делаем код-ревью,
* фокус на разработке, разработчики должны получать инфу о том, как разработка может повлиять на функционал.

1. Разработка автоматиз тестов
2. Инженерные практики.
3. Стандарты разработки
4. Автом статист код анализа
5. Юнит тестирование
6. TDD
7. Четкие правила что должно быть выполено

**Testing in Agile Automation**

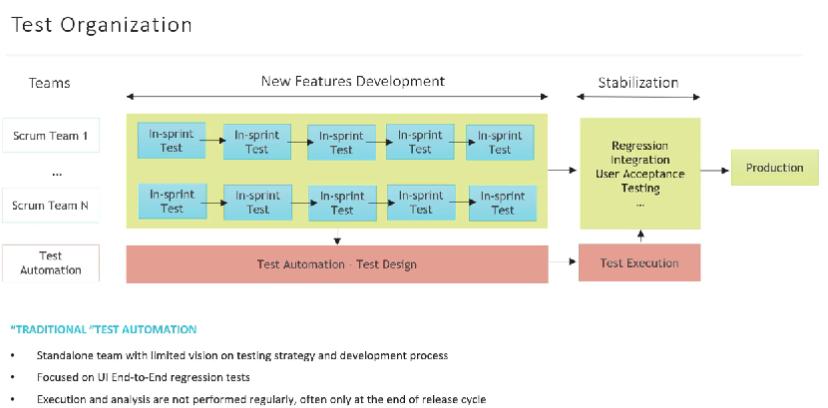
**Как же организовать автоматизацию в рамках эджайла:**

****

**Автоматизация в спринте:**

1. Взаимодействие с разработчиками, договориться кто и на каком уровне будет создавать тесты.
2. Какие будут id у контролов.
3. Уже начался спринт. Оценка сценариев. Как будет выполнено
4. Стараемся расписать шаги, которые можем уже сделать
5. Писать в моки те комп, которые недоступны.
6. Сфокусироваться на подготовке тестовых данных
7. Дебаг готовой фитчи с новыми тестами
8. Тесты готовы, включаем их в сиайсиди и вкл в коммиты, чтобы они приносили пользу

**Scrum-fall, организация тестирования**



**Какие основные проблемы?**

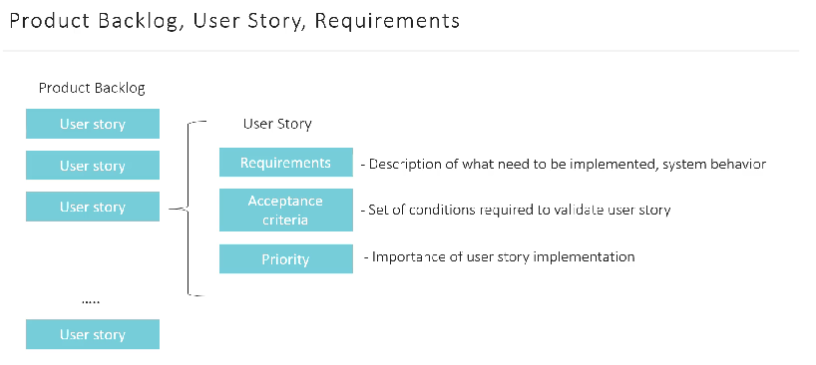
1. Нет четкого понимания для команды автоматизаторов что важно, поэтому она иногда не находит некоторое количество дефектов.
2. ….
3. Тесты становятся нестабильными
4. Тесты очень долго и много времени на поддержку.

**Что можем сделать?**

1. Больше взаимодействовать с командой тестирования, задавать вопросы, смотреть тестирование.
2. Можем делать какие-то комбинации.
3. Включение теста-автоматизации в CI/CD.
4. Параллельное выполнение тестов.
5. Выполнение тестов 1 раз в день.

**Requirements Testing. Introduction**

**Требование, какие бывают вопросы и как с ними работать?**



**User Story** может состоять из требований. Требования – необязательный атрибут юзер стори.

**Acceptance criteria –** набор условий, сценариев, которые показывают как эта функциональность может использовать приложение

**Priority -** команда будет понимать с чего начинать.

Типичные ошибки в требованиях:

* Неоднозначность.
* Страница логина должна быть юзер-френдли
* Неполные