**ATDD. Разработка ПО через отладочные тесты**

Разработка через приемочное тестирование ( ATDD ) — это методология [разработки](https://en.wikipedia.org/wiki/Software_development) , основанная на общении между бизнес-клиентами, разработчиками и тестировщиками. ATDD включает в себя многие из тех же практик, что и [спецификация на примерах](https://en.wikipedia.org/wiki/Specification_by_example) (SBE), [разработка на основе поведения](https://en.wikipedia.org/wiki/Behavior-driven_development) (BDD), разработка на основе примеров (EDD), и на основе поддержки разработка, также называемая разработкой через тестирование (SDD). Все эти процессы помогают разработчикам и тестировщикам понять потребности клиентов до внедрения и позволяют клиентам общаться на языке своего домена.

Приемочные испытания проводятся с точки зрения пользователя — внешнего взгляда на систему. Они исследуют внешне видимые эффекты, такие как указание правильного вывода системы при заданном входе. Приемочные тесты могут проверить, как изменяется состояние чего-либо, например, заказа, который переходит из «оплаченного» в «отправленный». Они также могут проверять взаимодействие с интерфейсами других систем, такими как общие базы данных или веб-службы. Как правило, они не зависят от реализации, хотя их автоматизация может и не быть.

**Создание**

Приемочные тесты создаются при анализе требований и до кодирования. Они могут разрабатываться совместно заказчиком требований (владельцем продукта, бизнес-аналитиком, представителем заказчика и т. д.), разработчиком и тестировщиком. Разработчики реализуют систему с помощью приемочных тестов. Неудачные тесты обеспечивают быструю обратную связь о том, что требования не выполняются. Тесты указаны в терминах бизнес-домена. Затем термины образуют вездесущий язык, который используется клиентами, разработчиками и тестировщиками. Тесты и требования взаимосвязаны. Требование, не прошедшее проверку, может быть реализовано неправильно. Тест, который не относится к требованию, является ненужным тестом. Приемочное испытание, разработанное после начала реализации, представляет собой новое требование.

**Стратегия тестирования**

Приемочные испытания являются частью общей стратегии тестирования. Это клиентские тесты, которые демонстрируют бизнес-намерение системы. Тесты компонентов — это технические приемочные тесты, разработанные архитектором, которые определяют поведение больших модулей. Модульные тесты создаются разработчиком для управления кодом, который легко поддерживать. Они часто получаются из приемочных и модульных тестов. Кросс-функциональное тестирование включает тестирование удобства использования, исследовательское тестирование и тестирование свойств (масштабирование и безопасность).

**Критерии приемки и тесты**

Критерии приемки — это описание того, что будет проверяться тестом. Учитывая такое требование, как «Как пользователь, я хочу взять книгу из библиотеки», критерием принятия может быть «убедиться, что книга помечена как извлеченная». Приемочный тест для этого требования дает подробную информацию, чтобы тест можно было каждый раз запускать с одним и тем же эффектом.

**Формат текста**

Приемочные испытания обычно имеют следующую форму:

**Дано (установка)**

Заданное состояние системы

**Когда (триггер)**

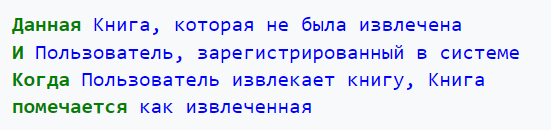
Действие или событие происходит

**Затем (проверка)**

Состояние системы изменилось или был произведен вывод

Кроме того, можно добавить операторы, начинающиеся с **И**, в любом из разделов ниже (Дано, Когда, Тогда).

**Пример**



**Преимущества:**

* главное преимущество данного подхода в том, что он запускает диалог между бизнесом и командой и дает взгляд со стороны пользователя;
* следующее преимущество в том, что вы не просто начинаете писать код, реализовывая абстрактные технические требования от аналитика, а вы пишите код и сразу тестируете себя.