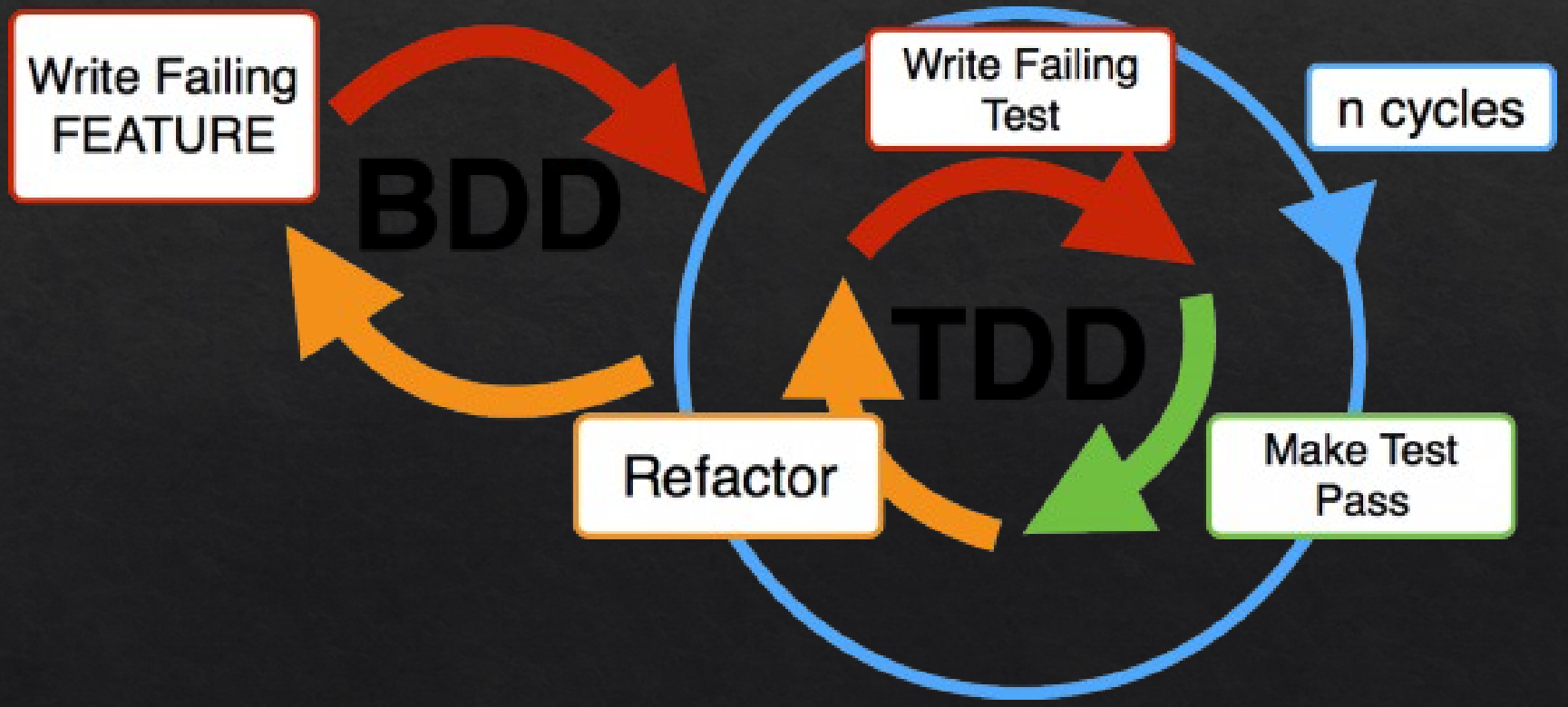

Behavior Driven
Development

A BDD (Behavior Driven Development) egy agilis fejlesztési módszertan, amely a viselkedésvezérelt tervezés (TDD) elveire épül



Célja:

- a szoftverfejlesztés minden szereplője számára érthető és egyértelmű specifikációt hozzon létre a szoftver viselkedésére vonatkozóan

Előnyei:

- jobb kommunikáció a csapattagok között
- a hibák korai felismerése és javítása
- valamint a szoftver minőségének javítása

Hátrányai

- A BDD bevezetése és alkalmazása időigényes lehet.
- A BDD hatékony alkalmazásához a csapattagoknak specifikus képességekkel kell rendelkezniük.
- A BDD automatizált tesztek futtatásához szükséges eszközökre és infrastruktúrára van szükség.

A BDD alapelvei

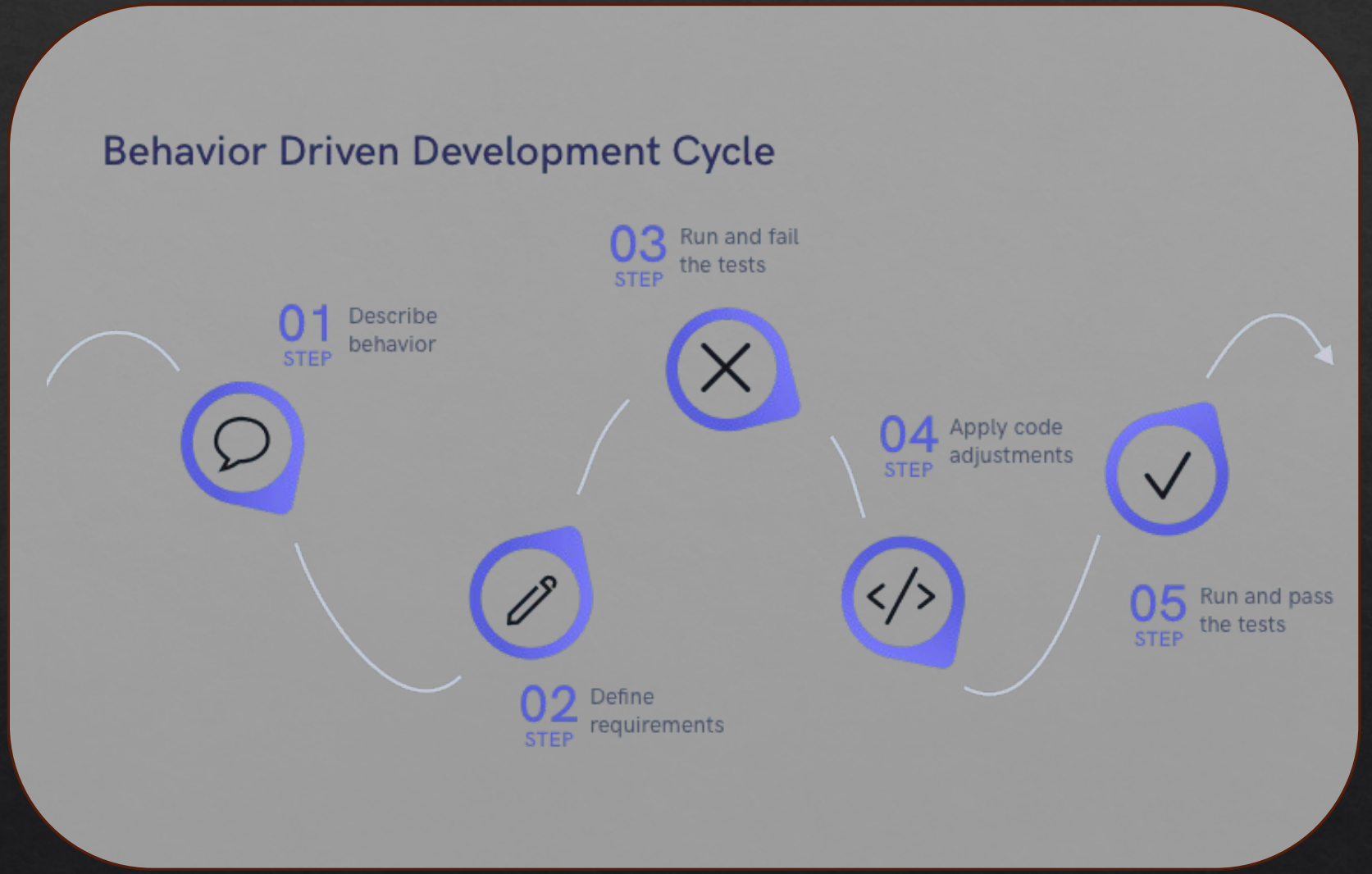
Elfogadás: A specifikációkat a megrendelőknek és a felhasználóknak kell elfogadniuk.

Automatizálás: Minden specifikációt automatizált tesztek formájában kell megírni.

Végrehajtás: A fejlesztő végrehajtja a forgatókönyvet, és ellenőrzi, hogy a szoftver a várt módon viselkedik-e.

A BDD folyamata

1. **Meghatározás:** A kívánt viselkedés leírása természetes nyelven.
2. **Tesztelés:** Automatizált tesztek írása a viselkedések validálására.
3. **Fejlesztés:** A szoftver implementálása a tesztek kielégítése érdekében.
4. **Ismétlés:** A folyamat ciklikus ismétlése a szoftver befejezéséig.



Business

Often named the "business analyst" (BA) or "product owner" (PO), the business role provides *what* problem must be solved. They provide requirements for the solution. Typically, the business role is non-technical.

Developer

The developer role provides how the solution to the problem will be implemented. They build the software and must be very technical. Examples help them to better understand and build the features using BDD.

Tester

The testing role, sometimes named "quality assurance" (QA), verifies that the delivered software product works correctly. They also try to find defects. The tester role must be somewhat technical. It is a complete shift left of the traditional tester.

Fejlesztői eszközök

Cucumber: Viselkedésleíró nyelv (Gherkin nyelven)

JBehave: Viselkedésleíró nyelv (Java nyelven)

SpecFlow: Viselkedésleíró nyelv (.NET nyelven)

Tesztelés cucumber használatával

- **Funkció:** A felhasználó kosárba tehet egy terméket.
- **Elvárt viselkedés:**
  1. A felhasználó kiválaszt egy terméket a terméklistából.
  2. A felhasználó rákattint a "Kosárba" gombra.
  3. A termék bekerül a kosárba.
  4. A rendszer visszajelzést ad a kosárba helyezés sikerességéről.

```
Feature: Termék kosárba helyezése

Scenario: A felhasználó kosárba tehet egy terméket

Given a user is on the product page
When they select a product
And they click the "Add to Cart" button
Then the product is added to the cart
And the system displays a success message
```

```
77  Before(async () => {
78      browser = await puppeteer.launch({ headless: false});
79      page = await browser.newPage();
80  });
81
82  Given('I am logged in to Outlook', function () {
83      login();
84  });
85
86  When('I send an email to a given email address', function () {
87      sendEmail();
88  });
89
90  Then('I check if the email appears in the Sent folder', function () {
91      checkSentFolder();
92  });
93
94  Then('I delete all emails from the Sent folder', function () {
95      deleteSentEmails();
96  });
```

Feladat

Legyen egy egyszerű kalkulátorunk, ami össze tud adni két számot.

```
Feature: Egyszerű kalkulátor # calculator.feature:1

  Scenario: Két szám összeadása          # calculator.feature:3
    Given Calculator létrehozása         # steps/steps.py:15
    When össze tudunk adni 3-at és 5-öt # steps/steps.py:20
    Then Az eredmény 8 lesz              # steps/steps.py:25

1 feature passed, 0 failed, 0 skipped
1 scenario passed, 0 failed, 0 skipped
3 steps passed, 0 failed, 0 skipped, 0 undefined
Took 0m0.002s
```