**Research**

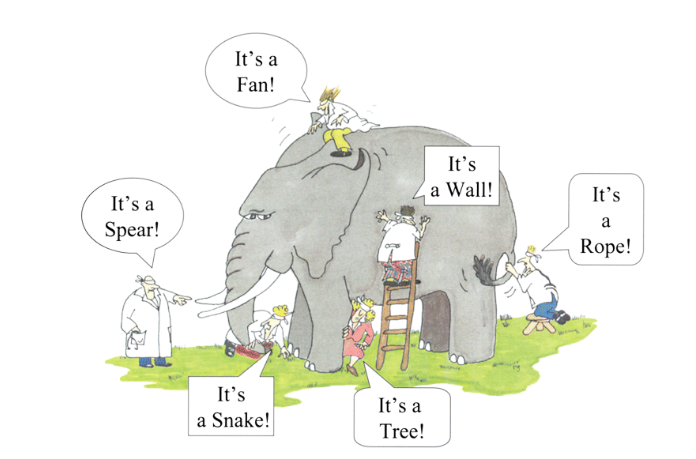
**When and why BDD was defined**

Behavior-driven development was developed by Dan North as a response to the issues encountered teaching test-driven development:

* Where to start in the process
* What to test and what not to test
* How much to test in one go
* What to call the tests
* How to understand why a test fails

BDD arose from the need to explain the function of the product, since the person who develops the idea is not the same person who has the idea. This is why agile software teams have learned to work in small increments, using software that is built progressively as the feedback tells stakeholders "Is this what you mean?"

Example:



**The most important aspects of BDD**

The main feature of BDD is that it focuses on acceptance tests. It made it easier for anyone on the test and read computer with this feature that provides business users for the testing process, helping the teams to explore and understand the requirements. The most important aspect is that it is written in plain text, using these keywords (given, when, then)

**Why is useful and when could be applied.**

It is useful because it helps explain and understand the requirements of the client, also helps as documentation of the code, being more understandable what the client wants and what the developer should do.

It is applicable to test the product, taking into account the history of the user, scenarios are created to test the functioning of certain sosftware functionality.

Applying BDD helps the client and the developer understand the requirements and thus be more readable and understandable to both parties.

**Which type of teams could implement BDD**

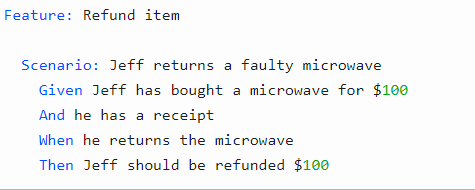
I think you should use both the development team and the Automation team, since

We write the application code based on the tests. This gives a first testing environment for the development and the code of the generated application turns out to be free of errors.

With each iteration we write tests and, consequently, with each iteration we obtain an automated regression package. This turns out to be very useful, since with each iteration we can be sure that the previous characteristics are working.

These tests serve as documentation of application behavior and reference for future versions.

**Structure of a feature – Give an example**



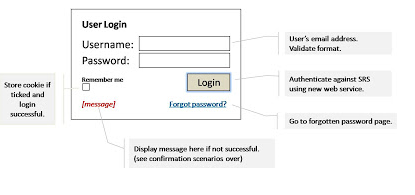
**Structure of a user story – Give an example**

User stories are short, simple descriptions of a feature told from the perspective of the person who desires the new capability, usually a user or customer of the system. They typically follow a simple template:

As a <type of user>, I want <some goal> so that <some reason>.

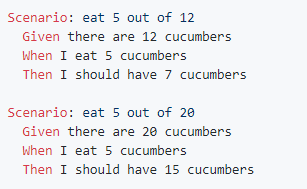
Examples:

* As a power user, I can specify files or folders to backup on file size, date created and data modified.
* As a user, I can indicate folders not to backup so that my backup drive is not filled up with things I do not need saved.

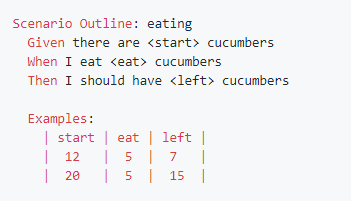


**Structure of a scenario – Give an example**

1 Example:



2 Example :



**Differences between BDD and BDT**

In software engineering, behavior-driven development (abbreviated BDD) is a software development process based on test-driven development (TDD). Behavior-driven development combines the general techniques and principles of TDD with ideas from domain-driven design and object-oriented analysis and design to provide software developers and business analysts with shared tools and a shared process to collaborate on software development.

