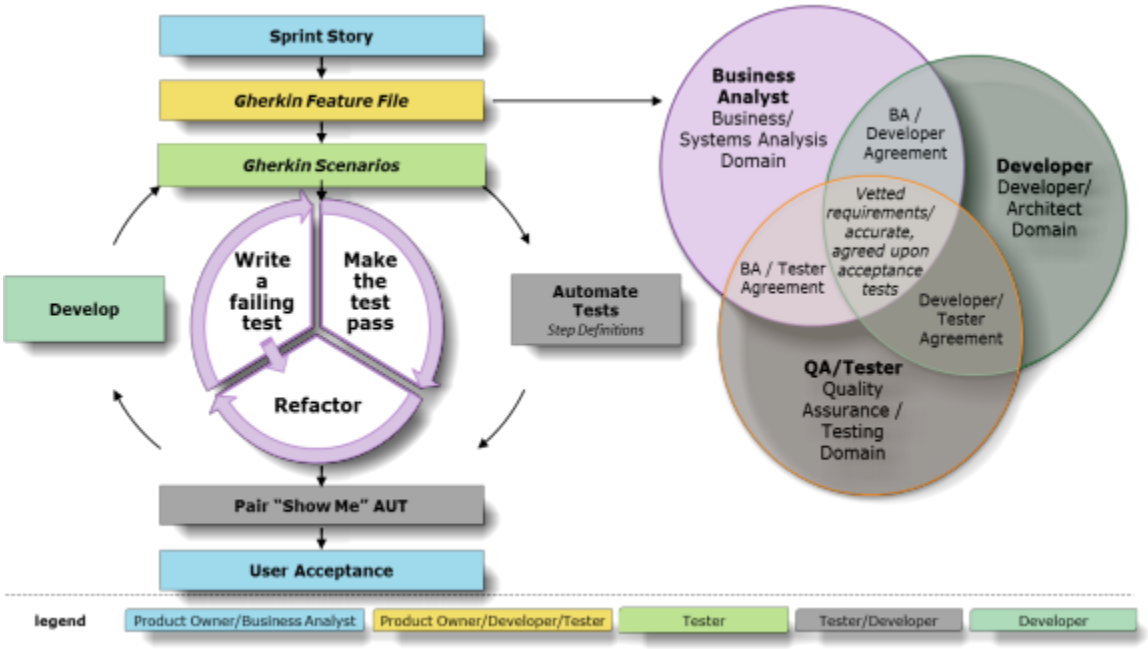
# Test Approach for ATDD

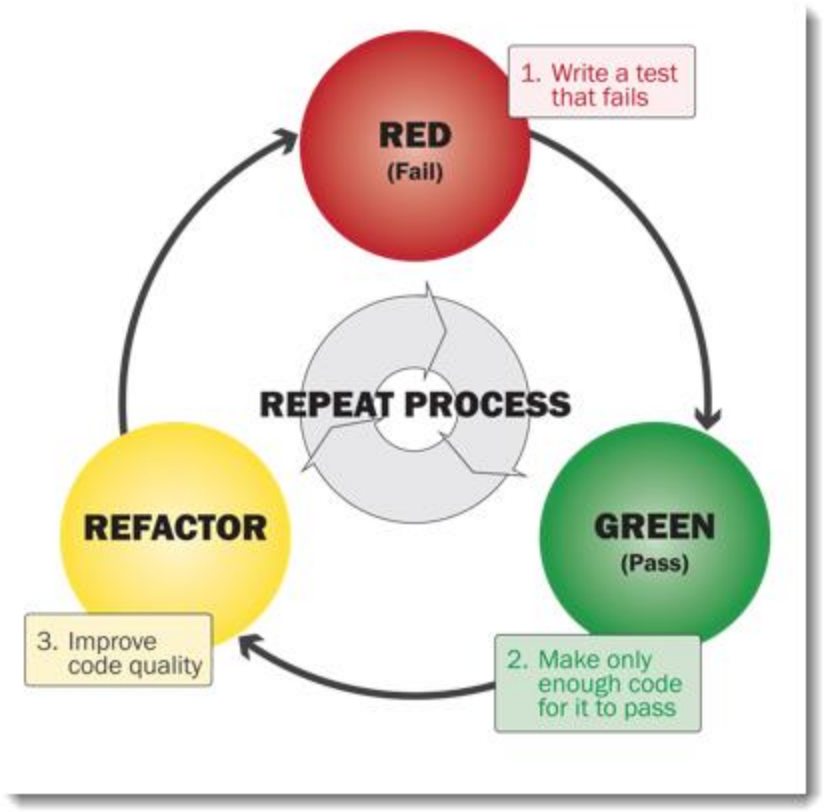
***ATDD usually involves establishing the criteria first,*** most often from a user perspective, and, ***acceptance tests are developed and run to see the results*** of failure with the right code based on examples.

Customer need technical help. Developer and Tester to provide technical support. Pair wise authoring. Developers need business knowledge. Customer can provide business rules. Pair wise implementation



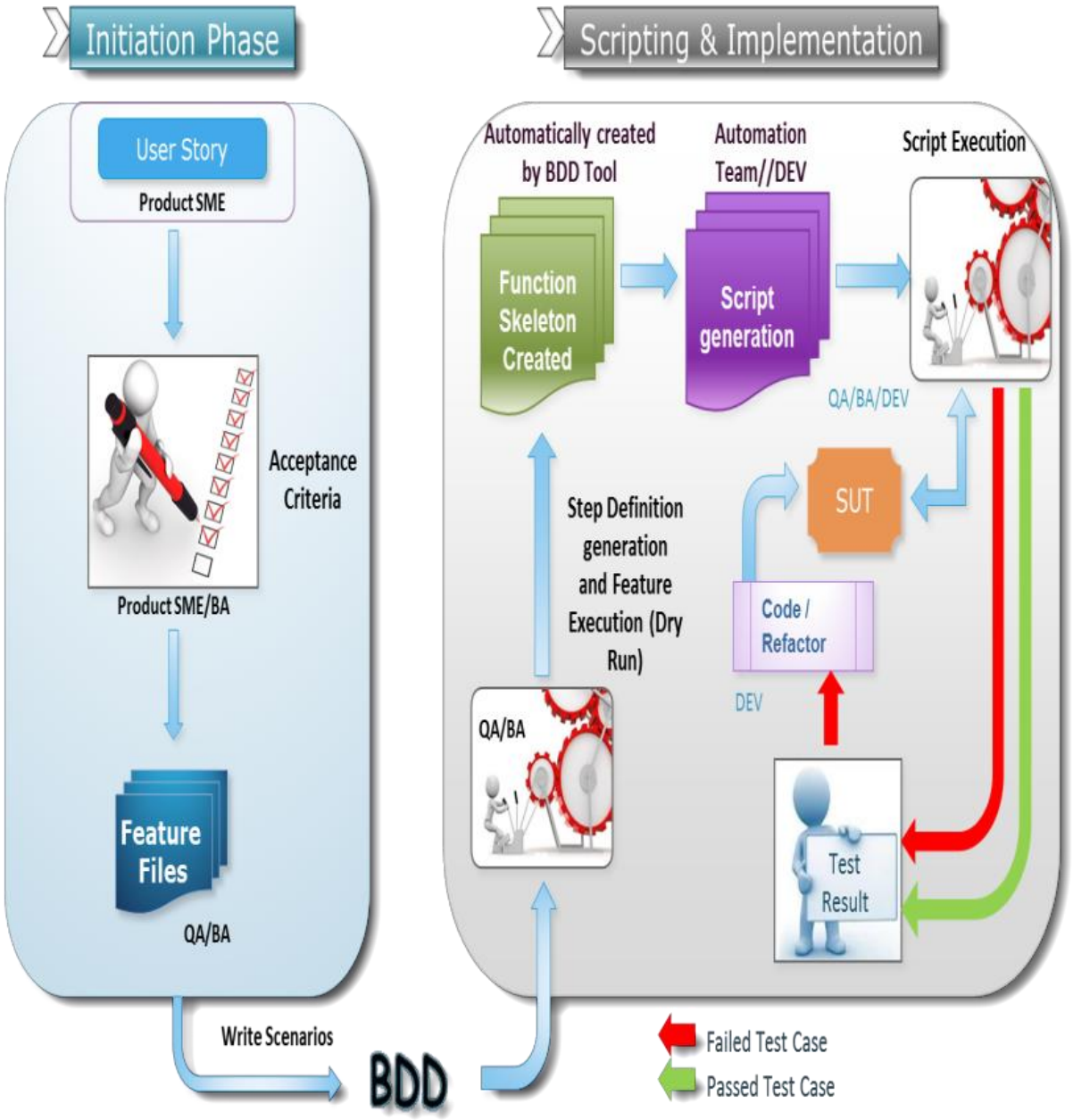

# Test Approach for TDD

***TDD is repetition of a very short development cycle.*** Code is written specifically to pass a given test case. When the written code successfully passes the test (‘green’), the passing code is refactored. Known as ‘red-green-refactor,’ this process is the mantra of TDD.




# Test Approach for BDD

ATDD combines the general techniques and principles of TDD with ideas from ***domain-driven design ATDD is practice of writing tests first, but focuses on tests which describe behavior***, rather than tests which test a unit of implementation.




***ATDD Vs TDD Vs BDD***

***Don't be biased on tools. BDD tools can be used in ATDD and vice versa also. Please focus and understand the practices /process in this blog.***

This table will give the ideation on when to use and whom to use these approaches -TDD, ATDD and BDD

| ***Approaches / Comparison Parameters*** | ***ATDD*** | ***TDD*** | ***BDD*** |
|---|---|---|---|

| | | | |
|---|---|---|---|
| ***Users Involved and Scope*** | Communication mechanism between Business user, Developer, Tester to ensure requirements are well documented | Developer approach between developer and tester to create well written unit of code (module, class, function) | Combination of ATDD and TDD. |
| ***Focus*** | Focus on capturing requirements in acceptance test and use to drive the development. Technique to bring the customer in design phase. | TDD is model and paradigm. | Focus on behavioral aspect of system for customer and developer but still practice of writing tests.<br>Bring the customer in testing phase and focus on behavior incrementally to certify. |
| ***Agile Steps*** | Step 1: Discuss,<br>Step 2: Develop,<br>Step 3: Deliver | Step 1 : Test,<br>Step 2 : Code ,<br>Step 3 : Refactor | Build the functionality incrementally guided by expected behavior. |
| | It should be repetitive. | It should be repetitive. | It is the extension of TDD and writing test to fail the feature / behavior |
| ***Input documentation*** | Acceptance Criteria + Examples (data and scenarios) = Acceptance Test | Requirement documentation will be base for development and testing. | Specification document in native language (Plain English) with given, when , then Acceptance criteria is also specified. |
| ***Automation Required*** | Doesn't require automation but needed for regression purposes. | Yes. Must have | Yes. Must have |
| ***Story / Feature : Test Mapping*** | Each story should have acceptance test | Each functionality should have implementation of test | Each story should have behavior test. |

| | | | |
|---|---|---|---|
| ***Acceptance criteria and Tests / Target level*** | Avoid implementation details.<br>1. Specific<br>2. Measurable<br>3. Achievable<br>4. Relevant<br>5. Time bound | Targeted at Implementation specific | Both of ATDD and TDD |
| ***Tools in the market*** | · Robot Framework,<br>· FitNesse,<br>· FIT | · Junit, TestNG, NUnit frameworks,<br>· Selenium tool ( any open source tools ) | · MSpec, Specflow – used to define the behavior.<br>· Cucumber with Selenium / Serenity |
| ***End users*** | ATDD Tests should be readable and focused for customers | TDD tests are technical and should be understood by developers/ testers | BDD tests should be understandable for both customers and IT team |