Lies and forgotten facts about Test Automation

Percy Pari Salas

Who can do test automation?

My opinion is contrary to how most test automation tools are sold
“not everybody can do test automation”

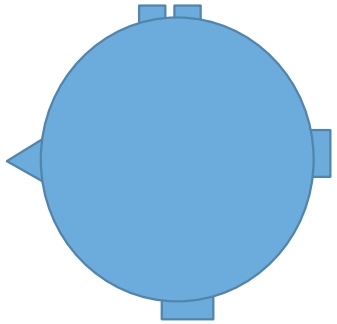
However, even when it seems contradictory I also think that
“test automation is not only for a few chosen ones… most people can work on it”

1. Anybody can write automated tests

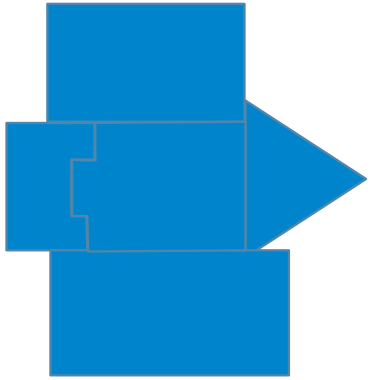
"… Our approach enables *non-technical* users to create automated business test cases in *plain English* so that *no programming of any kind* is needed …"

Why we need Technical Testers

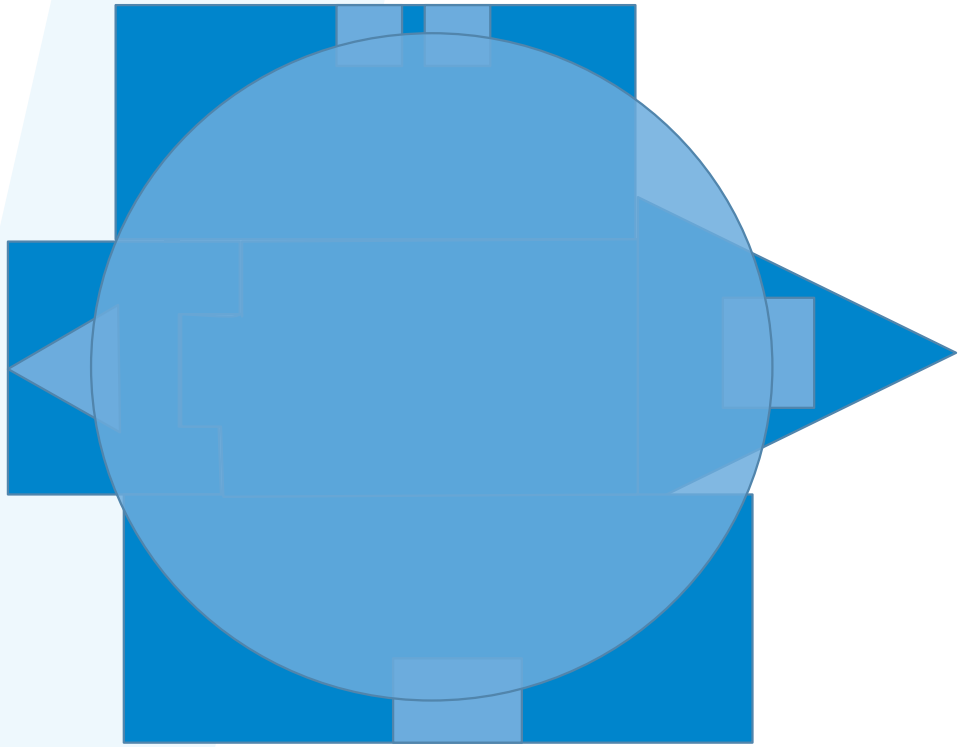
Requirements



Technical components
Development



Testing scope



No programming of any kind

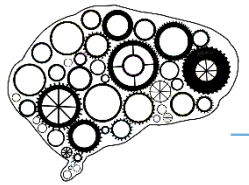
Given a 5 by 4 game
When I toggle the cell at (3, 2)
Then the grid should look like
.....
.....
..X..
.....
When I toggle the cell at (3, 1)
Then the grid should look like
.....
.....
..X..
..X..
When I toggle the cell at (3, 2)
Then the grid should look like
.....
.....
.....
..X..

```
private Game game;
private StringRenderer renderer;

@Given("a $width by $height game")
public void theGameIsRunning(int width, int height) {
    game = new Game(width, height);
    renderer = new StringRenderer();
    game.setObserver(renderer);
}

@When("I toggle the cell at ($column, $row)")
public void iToggleTheCellAt(int column, int row) {
    game.toggleCellAt(column, row);
}

@Then("the grid should look like $grid")
public void theGridShouldLookLike(String grid) {
    assertThat(renderer.asString(), equalTo(grid));
}
```

2. Writing automated tests code is too technical

- Learning a programming language is like learning a second language
- Generally, automated test scripts are a matter of simple programming
- A test case is defined as Input/Output pairs
  - Predefined Input/Output pairs is simple
  - Open Input/Output pairs require a process to calculate the Output (but ultimately the process to calculate the Output is the SUT itself)
- It can only be as difficult as writing the SUT itself
- Software has an intrinsic complexity. We can hide (transfer) it but not eliminate it. Complex to use would be easy to develop, easy to use will be complex to develop.

The path of abstraction



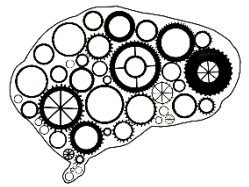
Business view

Require ments

Test cases

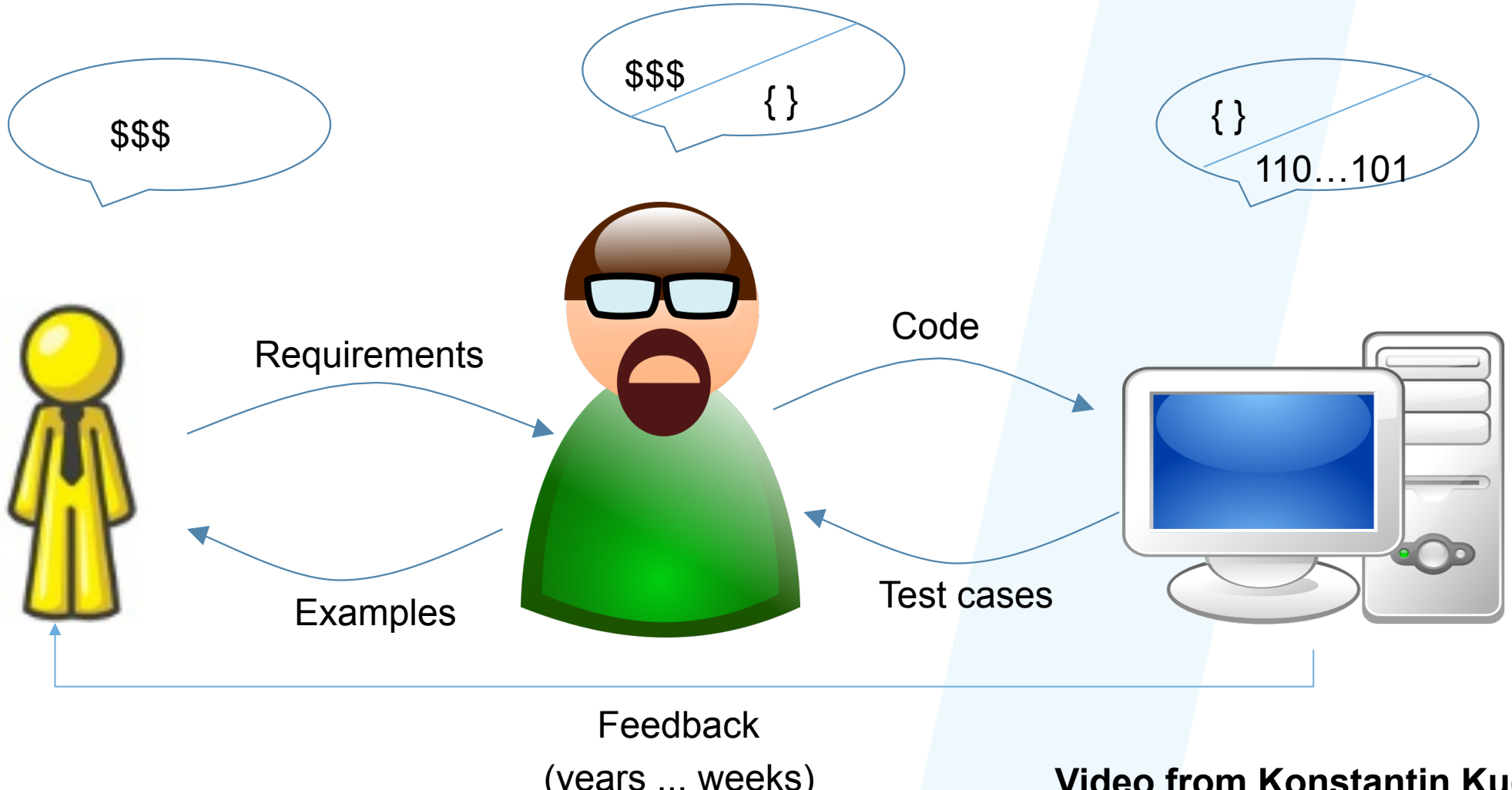
Programs (code)

Automated Test cases



Software

The cost of translation

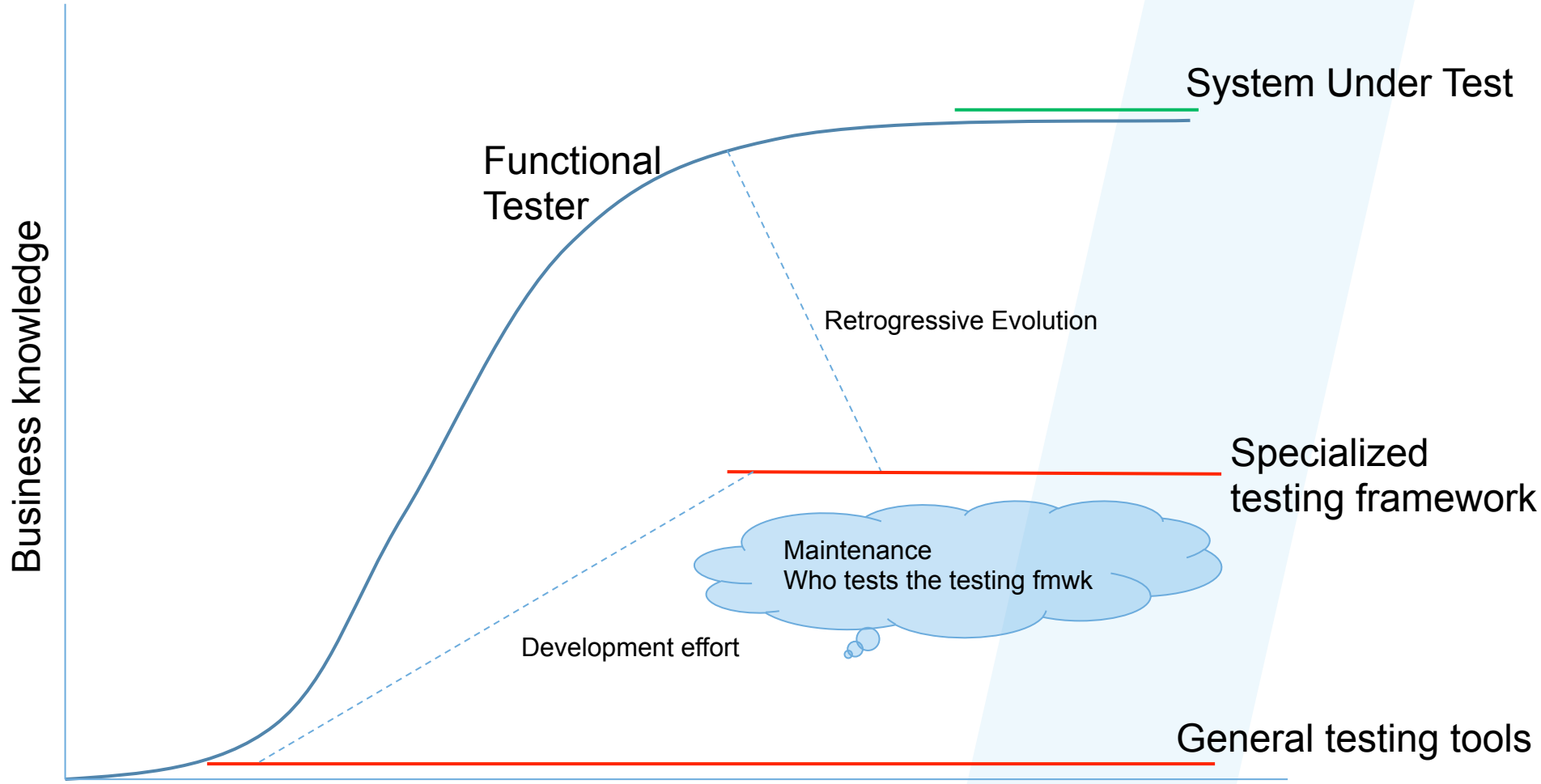


Video from Konstantin Kudryashov
https://www.youtube.com/watch?time_continue=2&v=njcHzGYv7nI

3. Test automation increases test coverage

- That is not automatically true
- Automated tests only check what they have been programmed to check by the person who wrote the script. The scripts are as good as the person who wrote them.
- Automated tests are good at catching regression issues
- The number of regression issues, in most cases, tends to be far less than issues in the new functionality that's being developed
- Test automation enables the testers (and other QA people) to perform tasks that otherwise they will not have the time or even the resources (e.g. data) to complete
- "The number of test cases (e.g. 500, 257, 39345) tells nothing to anyone about 'how much testing' you are doing." – James Bach

Building knowledge in testing framework



4. Over-marketed GUI testing

- GUI is constantly changing and having automated checks failing due to the UI changes and not changes in the functionality can give a false impression of the state of application.

- GUI tests are also much slower in speed of execution than Unit or API tests
  - Feedback to the team is slow. It might take few hours before a defect is spotted and reported back to the developers.
  - The root cause analysis takes longer because it is not easily apparent where the bug is.

- Increasing number of microservices. Software were the client is a machine/device, that interfaces with a device, not a human.

- Making automated tests GUI-proof is where their development can become a bit more difficult than the SUT itself

- Can we do testing without testing the GUI?
  - Exploit the separation of concerns in modern software development

5. Maintenance

- Maintenance of automated test scripts is more difficult than manual scripts

Motto : Be Abstract, Be Happy

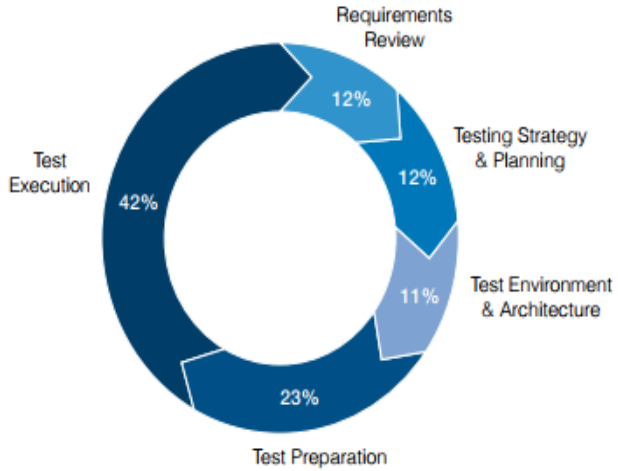
Notification screen: Ensure all relevant notifications display in list, ensure target field values are correct, select all the relevant notifications, check for issues (if any), resolve issues and proceed to order screen	All relevant open notifications on main equipment and sub-equipment (selection criteria: main equipment), <or> all relevant notifications on uninstalled sub-equipment (selection criteria: uninstalled sub equipment) will be listed and available to check and process On check, issues if any on the notifications will be highlighted in a log

- Can we resort to manual testing if our functional experts (SMEs) are not available?
- Can SMEs report defects meaningfully without testing training?

6. Test automation will drastically reduce costs

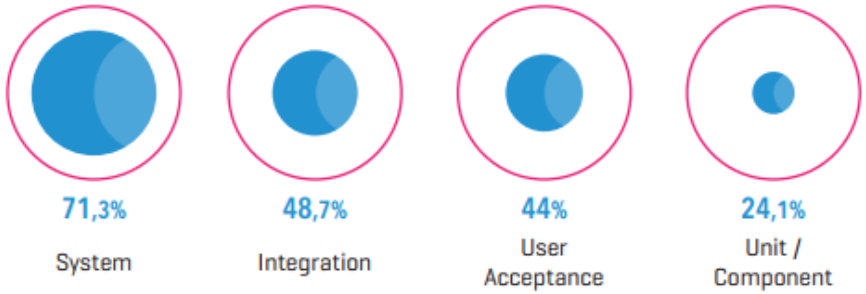
Planit Testing Index 2016

Allocation across testing phases



To which test level (s) is most of your budget allocated?

(Multiple answers were allowed.)



System testing is by far the activity absorbing most of the testing budget. This is in line with the EuroStar 2014 Survey.

(http://testhuddle.com/practices-attitudes-in-software-testing-study)

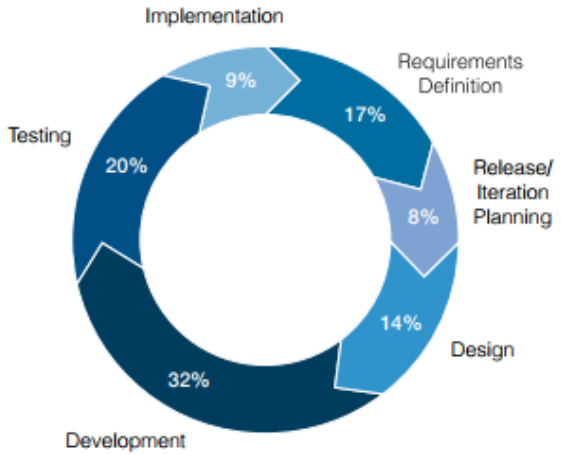
cost of automating tests

- If test automation reduces costs as far as usually promised, then to maximise cost reductions we should concentrate automation on the System level - that's a contradiction

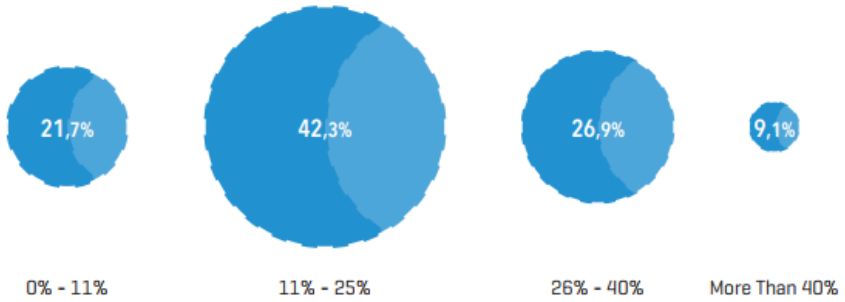
Budget proportions

Planit Testing Index 2016

Project budget allocation



What percent of a typical IT/ R&D project budget is allocated to software testing?



The large majority of respondents indicate budgets between 11% and 40%.
This is in line with World Quality Report 2015-16 that indicates an average expenditure of 26% for 2014 and 35% for 2015.

It makes sense that the budget allocated to Testing is the same as or close to the one allocated to Development

ROI and efficiency

- Difficult to see benefits over the first projects… it has to span over several projects

- There are other interrelated development activities than just scripting test cases
    - a framework needs to be developed that can support bespoke operations which are useful and meaningful for the business
    - the development of the framework is a project on its own and requires technical testers or developers and takes time to build

- The ROI is returned in the long run when we need to execute the same tests in regular intervals.
    - Returns are also noticed when we have to “execute under time pressures” – parallelized execution, after hours, maintaining execution pace

- Test automation enables the testers (and other QA people) to perform tasks that otherwise they will not have the time or even the resources (e.g. data) to complete
    - The “tomato sauce” analogy

7. Any organisation can implement Test Automation

- Budgets bucketed to projects
- Test automation cannot “die” with the projects
- Automated test scripts should become part of the SUT itself
  - This enables adequate maintenance
  - “Increases costs” of BAU activities
  - Introduces constraints in future projects

BAU – The never-ending project

- Testing budget and resources need to be allocated for management and maintenance of tests

- Tests “living near” the SUT code
    - Difficult to synchronise manual test scripts and their corresponding application code
    - The same applies to automated scripts if they are not readily available to developers
    - Training in automation tools can be necessary

Summary

Not everybody can do test automation but most IT people (**with the proper training**) will be able to play a role on it

Building a test automation framework is a development project itself and requires time and effort. Even when using a general framework/tool, we need to transfer/build the domain knowledge on it

Test automation requires an important initial investment and the ROI is gradual. The more you use it the more you get back (but it also “goes off”)

Apply test automation in scenarios like

- Regression testing
- Complex processes
- Big number of tests to be executed in a predefined short time-frame

Test automation should be part of any development activity

Thanks

Any questions?

pparisalas@planittesting.com