The feature-under-test: Access Point control of Client Transmit Power (TPC)

A WiFi client (e.g. a PC) is connected to the network via an Access Point (AP).

The WiFi standard defines the maximum power that a client can use to transmit messages to the AP. However, when the client is physically close to the AP there is no reason do to that, and the client can save battery power by using lower transmission power.

The feature:
The AP can send a control message to the WiFi client, instructing it what transmission power to use.

The client needs to adjust the transmit power according to the message from the AP.

Control mechanisms in the driver and the GUI allow the user to enable/disable this feature.

The transmit power is presented to the user via the GUI.

> **Note:** When this was written, we had a Test Engineering (TE) group that worked closely with the development and a validation team, based in another site, that tested the product as a black box with little contact with the development team.

Test Strategy Example

Testing follows the organization's generic test approach:

- Requirements-based testing using black-box testing techniques.
- Testing is started on the first code drop, by the TE group. At Alpha timeframe, the Validation team joins the effort, and by Beta the ownership of regression testing is handed off from TE to Validation. Regression mode continues till the release of the project, by the Validation team. TE support development in bug-fixing and testing engineering code drops before merging code changes to the main code branch.
- The priority of tests is set as part of the test design activity and is updated only if certain aspects of the TPC prove to be more (or less) buggy than expected.
- Test cases are specified, documented and managed in our Test Case Management System (TCMS).

WiFi transmission power is regulated by the FCC. Therefore, our test case list contains all the relevant regulatory test cases in the DV-Tx432-1969 standard.

Commented [SM1]: Most organizations have a "way we do things". In theory, when following ISO 29119 standard, the generic testing processes (those used for all the projects) are documented in the "Test Policy" document and the "Organizational Test Plan". Many companies – for sure small startups – don't have this well documented; but there is always a de-facto process that governs the general approach.

Commented [SM2]: Since this feature is tested by a certification body, there is a reference here to the certification tests and an explanation how we will ensure we pass them.

Don't look for this standard… it's just a number we invented.

In general, Test Engineering's approach is to test a feature in great detail, but to use limited selection of external variables (e.g. different APs, Operating systems). This means that all tests are designed and executed as early as possible in the project, ensuring the feature is working at least on some setups. These tests may fail later when the setup details change.

Some tests require a complicated setup that either exists already or is more in line with the activities of other groups (e.g. the hardware team). As a result, some identified test cases will not be executed by TE but left for the appropriate team to cover. Some sub-features go through basic testing only and the bulk of testing will be left for Validation. This partition is outlined in tables sections 4.2 and 4.3 of the Test Plan, where the responsibilities of each team are listed in detail. Specific test case lists are in section 5, including the assignment of tests to different teams.

After the first test-cycle by the and bug-fix cycle, the level of testing is reduced. TE continue to test the feature using a sub-set of the tests that are marked as "basic tests". TE run additional tests only when changes in the code are expected to affect the feature. In such cases, the appropriate test cases are be chosen.

While the feature deals with transmit power, this test plan does not call for intensive accurate measurements of the transmit power. Power measurement tests using an RF (Radio Frequency) meter are done as part of the Client Power Control (CPC) feature tests and it is redundant to repeat all of them here. We perform a minimal level of testing using accurate measurements, to verify that the GUI Tx power indicator is indeed correct. We then use the user-interface indicator to verify what is the transmit level that was set in the hardware.

Commented [SM3]: Clearly explain what you see as your responsibility. This will ensure everyone in the project is aligned with you and has the same expectations.

Commented [SM4]: When the work is divided across a few teams and locations, there is a risk of gaps – where each team thinks the other covers them. It is very important to be clear and document who does what. Some overlap is good; excessive overlap in activities is a waste of resources.

Commented [SM5]: It is important to mention if you intend not to test parts of the feature. This gives the reviewers a chance to comment about the wisdom of doing so, and also mean that no one will be surprised later to find out we do not test something.

Commented [SM6]: Don't look for these… we just want to show how the high-level is explained in the strategy, but eventually you need to go to lower level and clearly specify what features and what tests are covered by whom. Usually this is done in tables or test-case management systems.

Commented [SM7]: Give an general idea about the cadence of testing and the content of test cycles.

Commented [SM8]: Up till now this is more or less generic and fits many features and products. The next sections are more feature-specific.

Commented [m9]: This is a critical decision: It means that the TPC tests rely on a GUI indicator and assumes that the indicator is right since another test verifies that. This one decision makes the tests much easier to perform, as it leverages on the work that must be done anyway for another feature. The setup becomes trivial, as opposed to a setup that contains Radio Frequency power measurement equipment.

If you recognize such a strong link between your feature and some other feature, consider what can be leveraged to save work when testing either feature. If you find such opportunity, outline it in the Strategy section.

There may be cases where test duplication is obvious, and you decided to keep it anyway. In such cases, explaining your decision will save a lot of criticism of your plan.

Tests of this feature are not be part of the Build Acceptance Test (BAT), as a failure of the feature does not block testing of any other feature.

TE execute tests only on Windows 10 Home OS. The validation team expands testing to all the operation systems specified in the Product Requirements Document (PRD). Tests are executed using commercial laptop PCs clients. No specific vendor is mandated in the test plan.

Some requirements are not testable, since it is difficult or impossible to generate the test case. These are covered by code review.

The tests for this feature are not automated by Test Engineering, since there is already automation for the regulatory tests (The Pincho system). We are planned to get a Pincho system in house within the next 3 month, so there is no point automating ourselves. When the system arrives, this test plan will be updated accordingly.

Commented [m10]: Another important item to think about: Is this feature blocking others? If yes, consider adding some very basic tests to the BAT, to ensure it works.

Commented [m11]: Outline the OS and platforms that will be used

Commented [m12]: Outline how you deal with items that cannot be easily tested.

Commented [m13]: Note how automation strategy is part of the overall strategy

Commented [m14]: Note that this means that the test plan is something that can change as circumstances change. It is the responsibility of the feature owner to keep the test plan updated.

NOTE to our students:
This document is an example for the tone and some of the topics that can/may appear in a test strategy. It is NOT a template. It is NOT God-given truth. It does not mean that if you don't have all the details mentioned in this example, you will lose points, nor that if you DO have all the details here, you have a good strategy.

Don't invent things that don't exist (e.g. additional imaginary test teams). We defined a situation for you: Test Tesseract assuming a specific set of requirements, with a team of 4-5 people, using up to 300 images and less than 15 minutes of PC time). This is what you have.

Consider what Tesseract does; the limitation imposed by the exercise; common-sense; risk (what if *this* part or *that* aspect don't work? What's the loss? What's the chance of missing it if you don't test much?... etc.). etc.

Oh: And when you use "common sense" to decide on things, remember that what's common-sense to you, may not be so for us… so EXPLAIN why you decided to do what you did.