informatech
Specialist IT Testing

ANZTB Canberra SIGiST: Practical Strategies for Being a Better Tester

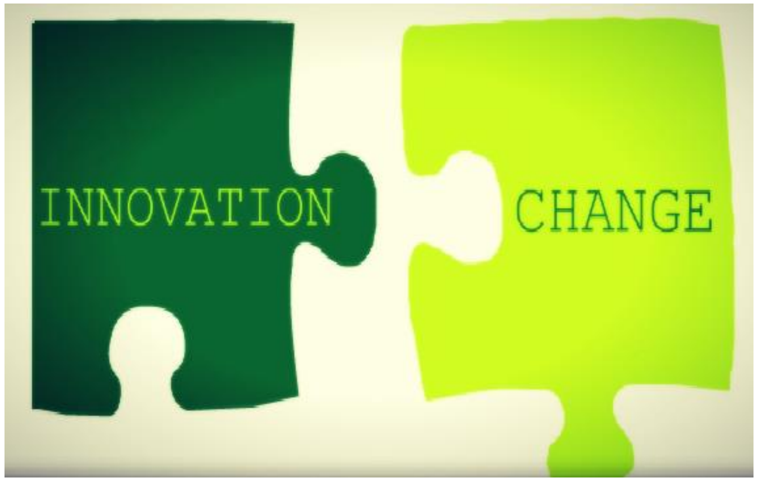
Presented by Nathan Bligh

Nathan Bligh – The Tester

- ✓ Professional Tester with 10 years dedicated testing experience
- ✓ Has worked on some of the largest and most complex projects in the Southern Hemisphere
- ✓ 1 of only 22 people in Australia and New Zealand Certified at the ISTQB Full Advanced Level Test Professional
- ✓ Director of a company providing premium testing services "Informatech – Specialist IT Testing"
- ✓ Contact Details
  - ✓ Nathan.bligh@Informatech.com.au
  - ✓ au.linkedin.com/in/nathanbligh/en

1. Ask Yourself "Why am I Doing it This Way" More Often – The Problem

- ✓ It's easy to continue doing the same things we have always done
- ✓ It can be hard to identify new ways of doing things
- ✓ How do you know if a different way of doing something will be any better than the existing way
- ✓ If you do identify a new way of doing something how do you justify it to management



1. Ask Yourself "Why am I Doing it This Way" More Often – The Solution

The Dos

- ✓ Think critically about what you are doing and why you are doing it
- ✓ Challenge long held ideals of "this is how it has always been done"
- ✓ Ask yourself if what you are doing is going to provide the best outcome for the business
- ✓ Provide options for different ways of doing things

The Don'ts

- × Don't confuse critical thinking with negativity
- × Don't leave it until the last minute to ask questions (though late is still better than never)
- × Don't look to change everything – some things may be working perfectly fine

1. Ask Yourself “Why am I Doing it This Way” More Often – Example

How many times have you written a suite of tests, run them all in “Cycle 1” of the Test Execution Phase, found few or no Defects only to be asked by the Test Manager to run the exact same tests again in “Cycle 2” and “Cycle 3”?

Ask yourself “**why am I doing it this way**” - what is the benefit? If you found no Defects the first time then what is the likelihood of finding Defects the second, third, or subsequent times (remember the “Pesticide Paradox”)?

Is it an effective use of your time and skills? Could your time and skills be better spent performing exploratory testing, error guessing, or analysing existing Defects to identify areas that merit further testing?

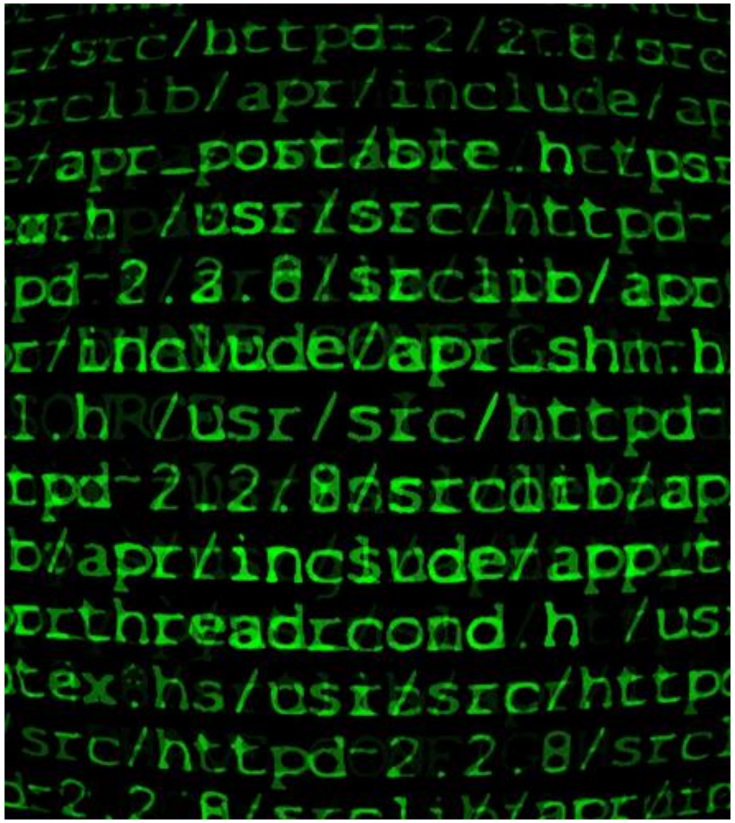
These are all valid questions that should be asked.

1. Ask Yourself "Why am I Doing it This Way" More Often – The Benefits

- ✓ More effective use of test resources and minimising "wasted effort"
- ✓ Potential for greater test coverage by refining processes and procedures
- ✓ Improved job satisfaction for testers by allowing them to stimulate their minds and be creative
- ✓ Potential for cost and time savings through more efficient activities
- ✓ Avoiding negative conformist behaviours

2. Learn a Programming Language – The Problem

- ✓ Need to be able to speak a "common language" with Developers
- ✓ Demonstrating technical ability and proficiency
- ✓ Understanding the technology so that you can apply appropriate test techniques
- ✓ A program or piece of software is more than just a "black box"



2. Learn a Programming Language – The Solution

The Dos

- ✓ Pick a language that is relevant to your work
- ✓ Make use of free resources such as w3schools
- ✓ Make sure you learn the theory of how and why things work
- ✓ Perform exercises to reinforce what you have learned
- ✓ Use your newly learned skills to improve your testing

The Don'ts

- × Don't skip the basics – they are the key to ensuring you understand the language
- × Don't give up when things get difficult
- × Don't be afraid to ask your peers for help if you get stuck
- × Don't get cocky just because you can cut code - it takes years to become proficient

2. Learn a Programming Language – Practical Example

```
#Loop through each object in the Expected Details array, pull out the
server name, gather machine objects and write them to the actual details array

foreach ($objServer in $ExpectedDetailsArray)
{
   $GlobalServerName = $objServer.Server.toUpper()

   $NewServerActual = "" | Select "Server", "OSName", "OSVersion", "IPv4",
                  "IPv6", "CPU", "RAMCapacity", "NIC", "NTP", "WSUS"
   $NewServerActual."Server" = $GlobalServerName
   $NewServerActual."OSName" = getOS("Name")
   $NewServerActual."OSVersion" = getOS("Version")
   $NewServerActual."IPv4" = getIPv4Status
   $NewServerActual."IPv6" = getIPv6Status
   $NewServerActual."CPU" = getCPUSpecs
   $NewServerActual."RAMCapacity" = getRAMCapacity
   $NewServerActual."NIC" = getNetworkAdapters
   $NewServerActual."NTP" = getNTPServer
   $NewServerActual."WSUS" = getWSUSSource

   $ActualDetailsArray += $NewServerActual}
```

2. Learn a Programming Language – The Benefits

- ✓ Improved skill set and ability to perform decision testing, statement testing, and other technical testing
- ✓ Increased respect from other members of the project team, particularly developers
- ✓ Enables you to automate repetitive tasks to save time and money and achieve more accurate results
- ✓ Gives you a great new skill to put on your CV

3. Get Educated – The Problem

- ✓ Lack of respect of the testing industry by other disciplines
- ✓ Perception that "anyone can be a tester"
- ✓ Challenges in selling testing as an area of technical expertise
- ✓ Struggle to achieve consistent understanding of terminology, techniques, strategies, and approaches

3. Get Educated – The Solution

The Dos

- ✓ Buy a textbook focused on testing (and actually read it)
- ✓ Undertake some form of recognised testing certification
- ✓ Share your knowledge and experiences with others
- ✓ Analyse and learn from your mistakes
- ✓ Get out of your comfort zone and learn something new

The Don'ts

- × Don't think you can be an expert tester without continual education
- × Don't go on a course just for the sake of it. If you don't learn anything then you haven't achieved anything
- × Don't attend courses and then not sit the exams. Assessment of your knowledge and understanding is an important part of the learning process

3. Get Educated – Interactive Exercise

Everybody please stand up.

3. Get Educated – The Benefits

- ✓ Formally recognised qualifications and certifications show that you are willing to go above and beyond
- ✓ Learning new techniques and strategies helps to improve the way you test
- ✓ It makes you more desirable from an employment perspective
- ✓ It gives you the confidence to provide informed opinions and arguments
- ✓ **It gives you credibility as a professional tester**

4. Make Better Use of Standards – The Problem

- ✓ Lack of respect of the testing industry by other disciplines
- ✓ Difficulty in demonstrating proficiency in the testing domain
- ✓ Struggle to achieve consistent understanding of terminology, techniques, strategies, and approaches
- ✓ We don't know what we don't know

4. Make Better Use of Standards – The Solution

The Dos

- ✓ Become Familiar with various Standards:
  - ✓ ISO29119 is the new Standard for Software Testing
  - ✓ ISO 25000 lists a set of quality attributes and sub-attributes
- ✓ Use Standards as a tool – apply them where they fit and adapt them if needed

The Don'ts

- × Don't ignore competing standards or in-house processes
- × Don't be afraid to "mix and match" different standards if you believe it will give you a better result
- × Don't let standards dampen your creativity

4. Make Better Use of Standards – Example

You have recently been given the opportunity to manage the testing for a new project. You have assisted with Test Management previously but this is your first time being responsible for the entire testing effort.

The Project Manager has asked you to develop a high level test planning document and within this define a set of quality attributes that you will verify during the testing lifecycle. Where do you start?

- IEEE829/29119-3 provides a comprehensive definition of test documentation, document contents, templates, and examples
- ISO 9126/25000 provides a list of quality attributes, sub-attributes, and logical descriptions for each of these that can form a starting point

It won’t define the required implementation for your project but it will provide you with information to guide your testing effort.

4. Make Better Use of Standards – The Benefits

- ✓ You don’t have to reinvent the wheel by attempting to create your own model
- ✓ You have the backing of documented standards developed by professionals in their field of study
- ✓ You have pre-defined expectations about what testing documentation to develop and what those documents should contain
- ✓ There is extensive literature that can help to guide you

5. Learn About Non-Functional Testing – The Problem

- ✓ It tends to be more difficult than functional testing
- ✓ It often requires a specialist set of skills to perform
- ✓ It can be harder to sell the need for non-functional testing and obtain appropriate funding and resources
- ✓ The known-unknown – we know non-functional aspects exist so if we don't test them then what are we missing?

5. Learn About Non-Functional Testing – The Solution

The Dos

- ✓ Review the ISTQB Advanced Technical Test Analyst syllabus for different types of tests
- ✓ Become familiar with the ISO9126/ISO25000 Quality Characteristics Standard
- ✓ Analyse how you can apply non-functional testing to your situation
- ✓ Remember that something is better than nothing

The Don'ts

- × Don't limit yourself only to Performance Testing – there is so much more out there
- × Don't let yourself become overwhelmed by the technical nature of non-functional testing
- × Don't give up when it gets hard (and it will)

5. Learn About Non-Functional Testing – Examples of Quality Characteristics

Here are just a few of the quality characteristics (and sub-characteristics) that you could verify using non-functional testing (taken from ISO 9126).

Reliability: Maturity, Fault Tolerance, Recoverability
Usability: Understandability, Learnability, Operability
Efficiency: Time Behaviour, Resource Utilisation
Maintainability: Analysability, Changeability, Stability, Testability
Portability: Adaptability, Installability, Co-Existence, Replaceability

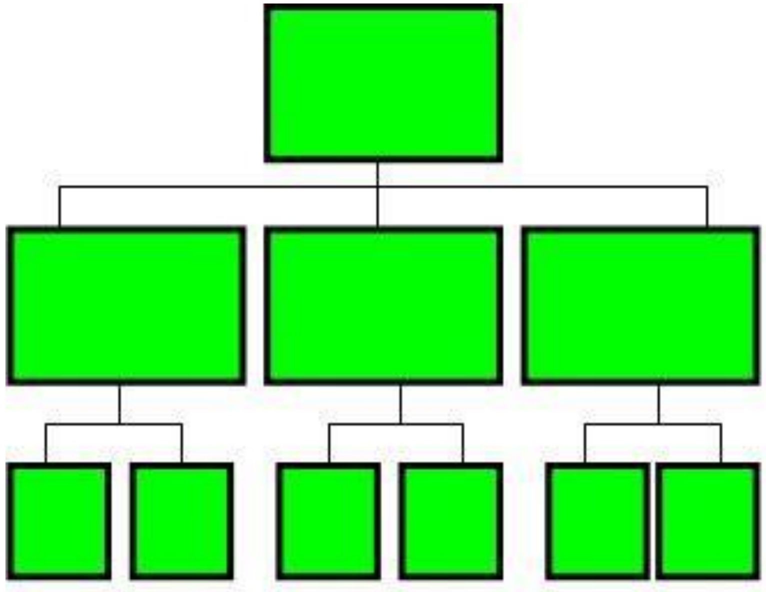
By no means do these represent all different characteristics that could be verified from a non-functional perspective. But if you managed to cover all of these off you would be doing pretty well.

5. Learn About Non-Functional Testing – The Benefits

- ✓ It's interesting!!
- ✓ It provides you with an advantage over other testers who focus only on functional testing
- ✓ It helps to verify areas that may not be addressed by functional testing, identify new defects, and provide additional confidence of a thoroughly tested product
- ✓ It gives you a much better idea of the overall quality and function of the system under test

6. Make Better Use of Test Levels and Phases – The Problem

- ✓ Systems are becoming larger and more complex making them more difficult to test
- ✓ Users are demanding higher levels of quality and are less tolerant of errors in software requiring more testing
- ✓ Because of these things Testing is becoming harder to manage
- ✓ How do you know what should be included in a single Level/Phase or split across multiple Levels/Phases



6. Make Better Use of Test Levels and Phases – The Solution

The Dos

- ✓ Study methodologies like the V Model and understand how different activities can result in different Test Levels/Phases
- ✓ Review the ISTQB Advanced Test Manager Syllabus for examples of different Test Levels/Phases
- ✓ Group logically related items and consider making them a Test Level/Phase (e.g. Security)

The Don'ts

- × Don't be afraid to challenge the theory and create your own Test Level/Phase if you can justify the benefits and it logically fits within your project
- × Don't introduce multiple Levels/Phases if there is not a genuine need for it

6. Make Better Use of Test Levels and Phases – Real Life Example

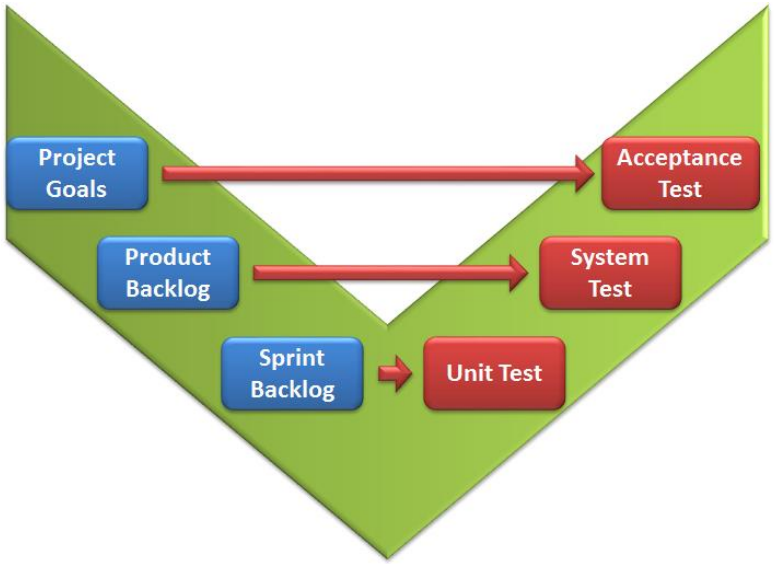
In recent years I have been including a Test Level/Phase that I call "Operational Readiness Testing". Its goal is to ensure that the hardware and services that support the software are continually available to end users by verifying things like: redundant network adapters, server clustering, disaster recovery procedures, failover and failback of services, data centre start up procedures, and other similar activities.

All of these activities are transparent to the system users but are of vital importance to ensuring maximum up-time. And you'd be surprised how many Defects we've been finding in these areas.

How do you think your organisation could benefit from taking a similar approach?

6. Make Better Use of Test Levels and Phases – The Benefits

- ✓ Scope is broken down in to more manageable bodies of work
- ✓ Logically related entities are grouped together providing opportunities for efficiencies to be realised
- ✓ Easier to allocate the correct people, with appropriate skills, to the applicable Level/Phase
- ✓ More appropriate Test Entry and Test Exit Criteria can be defined for each Test Phase



7. Learn How To Say "Yes, But" Instead Of "No" – The Problem

- ✓ As testers we tend to be inherently pessimistic. In fact, we have often been described as the "professional pessimist".
- ✓ We need to be careful not to carry this over to other activities and need to look for ways to achieve difficult tasks rather than saying it can't be done.
- ✓ We are often perceived as a roadblock to projects

7. Learn How To Say "Yes, But" Instead Of "No" – The Solution

The Dos

- ✓ Make a conscious decision to place the success of the project over the success of testing
- ✓ Remember that Testing is there to support quality outcomes, not hold the project to ransom
- ✓ Look for mutually beneficial solutions
- ✓ Look for alternative options such as prioritisation or incremental releases

The Don'ts

- × Be careful not to become a "yes man/woman"
- × Don't be afraid to raise issues just because you are trying to be positive
- × Don't succumb to pressure to achieve deadlines if the quality of the product simply isn't good enough to release it

7. Learn How To Say "Yes, But" Instead Of "No" – Example

You have determined that you need 4 weeks to test a small system. Development runs overtime and the Project Manager asks you if testing can be finished in 2 weeks to achieve the end date and avoid slippage.

Rather than saying "**No**, it can't be done" because you feel frustrated that testing has been 'short changed' again, try something like this: "**Yes, but** we may not be able to run all of the tests in that amount of time. If there are tests and defects outstanding after the two weeks are up we will need to assess the residual risk and determine if it is acceptable before proceeding".

7. Learn How To Say "Yes, But" Instead Of "No" – The Benefits

- ✓ You avoid the "too bad, make it happen" scenario
- ✓ You demonstrate you are willing to try and achieve project goals
- ✓ It more accurately articulates what the limitations are than a flat out no
- ✓ It offers a win-win solution rather than a win-lose or a lose-lose
- ✓ **It stops testing being seen as a roadblock and helps to improve the perception of the testing discipline**

Summary

- ✓ Don't limit yourself to the concepts and strategies proposed in this presentation
- ✓ Think for yourself, challenge the norm, and look for better ways to do things
- ✓ Take steps to improve your skills and make yourself a better tester
- ✓ Look for ways to enhance the testing discipline and how people perceive it
- ✓ Be proud of your chosen Profession