<Continual Service Improvement & Principles In Software Testing>

Myron Angelo Dizon
Web Quality Assurance
Freelancer.com Inc. Philippines

The Life In The Service Management Industry
@ Engineering Conference Hall
March 1, 2019

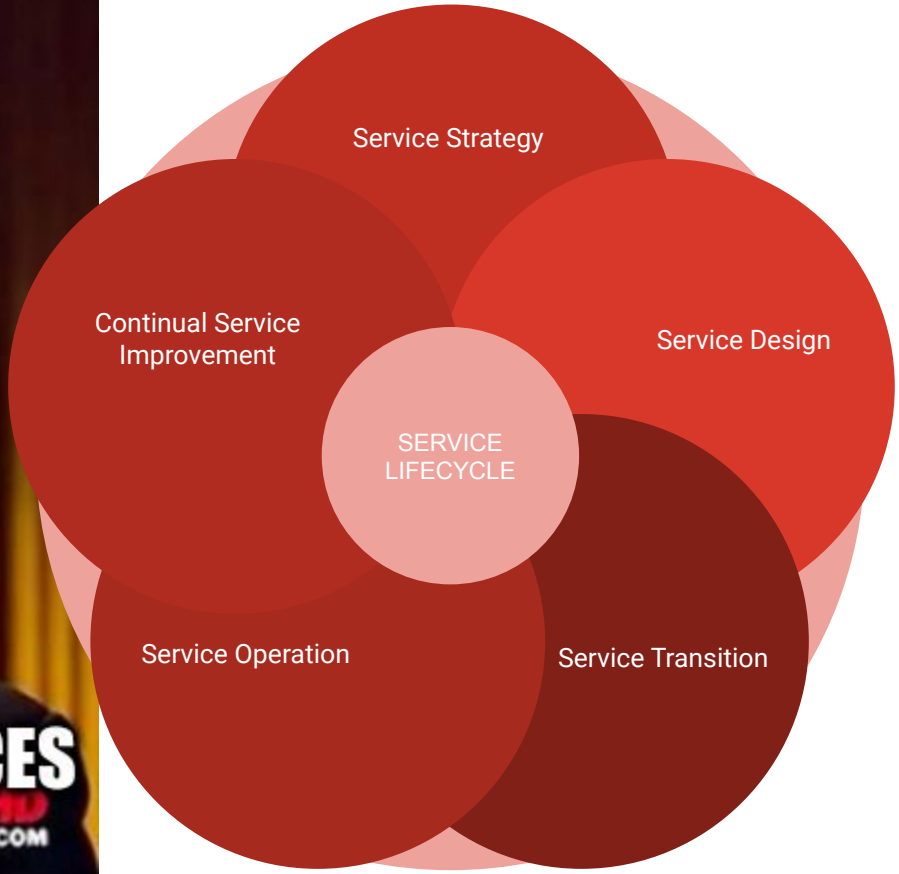
SETTING EXPECTATIONS

by the end of this talk you should've walked away with:

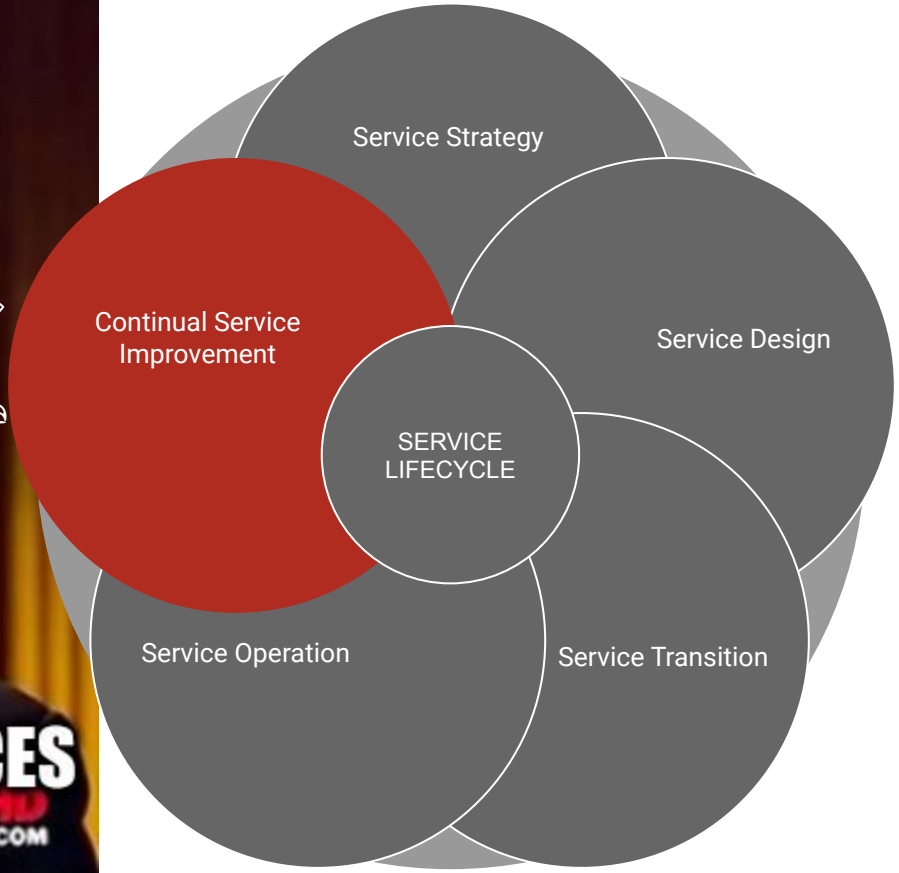
- CSI framework as it how it integrates to business processes... and everyday life
- an understanding of testing principles to produce high quality software
- some swag... and stickers!

dude pare chong bro, what is ITIL?


IT INFRASTRUCTURE LIBRARY (ITIL)
A FRAMEWORK OF BEST PRACTICES FOR DELIVERING IT SERVICES
imgflip.com


Service Strategy
Continual Service Improvement
Service Design
SERVICE LIFECYCLE
Service Operation
Service Transition


IT INFRASTRUCTURE LIBRARY (ITIL)
we're gonna focus here
A FRAMEWORK OF BEST PRACTICES FOR DELIVERING IT SERVICES
imgflip.com


Service Strategy
Continual Service Improvement
Service Design
SERVICE LIFECYCLE
Service Operation
Service Transition

wait lang bro, mej nalilito ako sa terms

Continuous Improvement

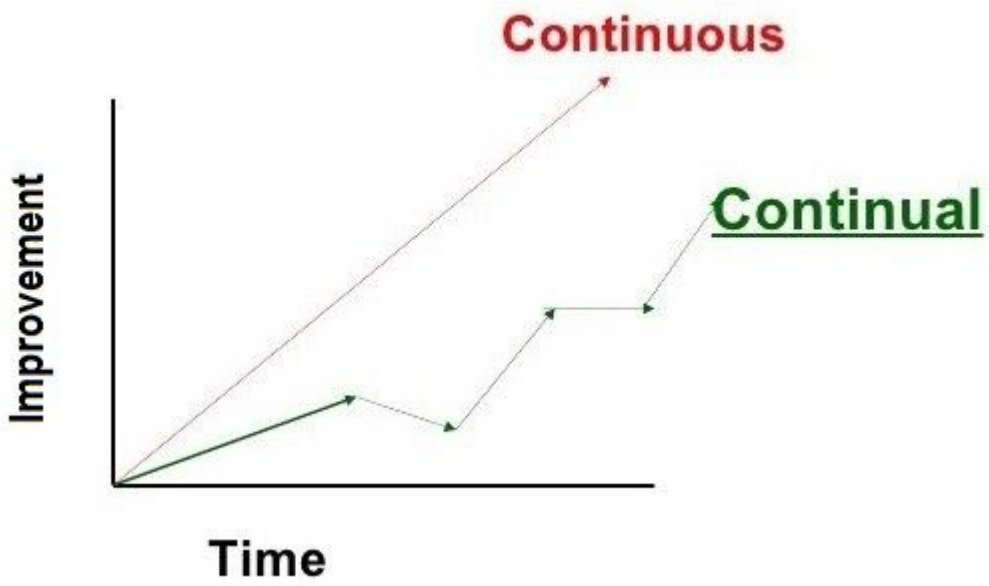
Continuous improvement is simply a chain link to continual improvement, which focuses more on, processes with short/small improvements within a system.

Continual Improvement

Continual improvement is a concept initially to make changes and improvements in the existing systems to produce better outcomes.

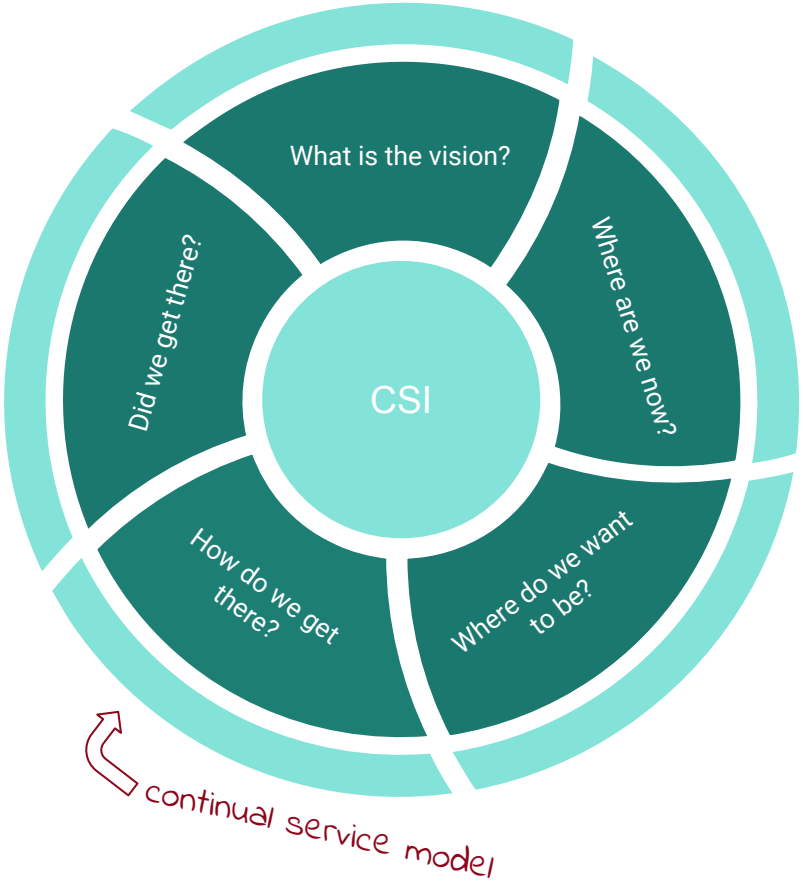
Continuous Improvement

Continual Improvement



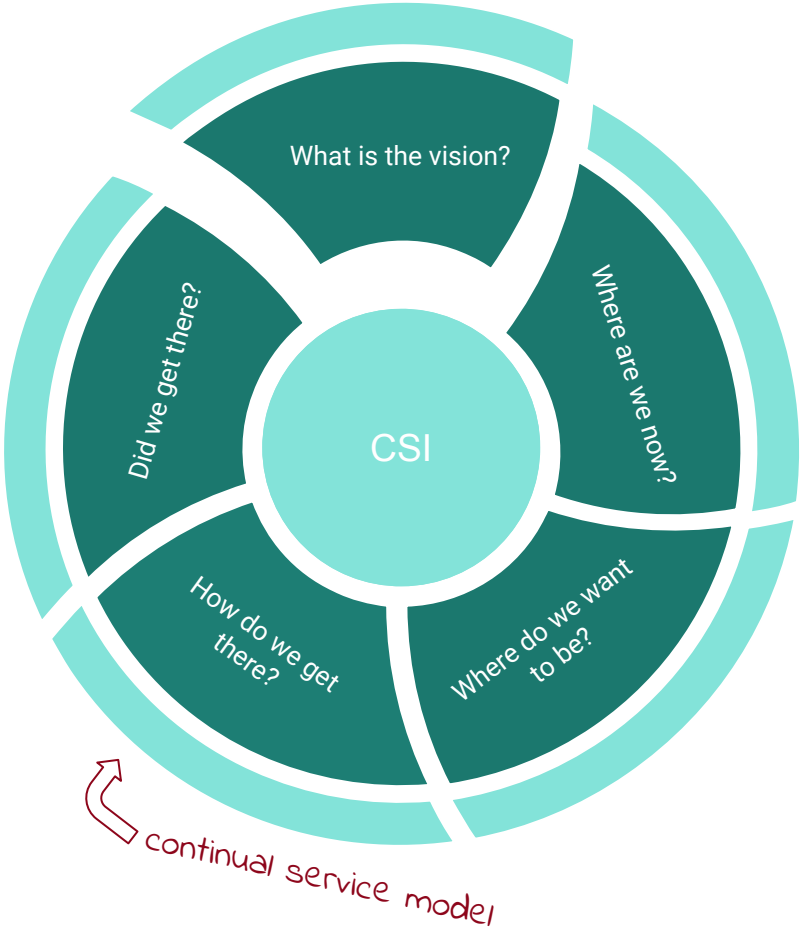
to get things going,

to get things going,
what is the CSI approach?



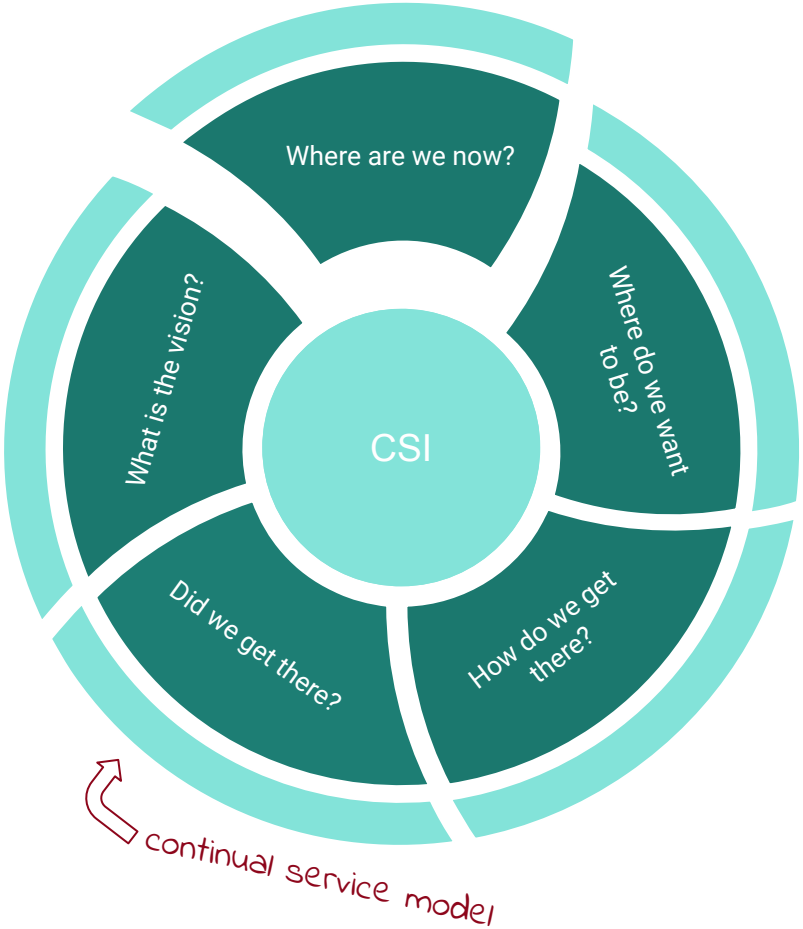
ITIL CSI APPROACH

The purpose of this cycle is to continually align and re-align IT services to the changing needs of the business. This cycle looks for ways to improve process effectiveness, efficiency, and cost effectiveness.



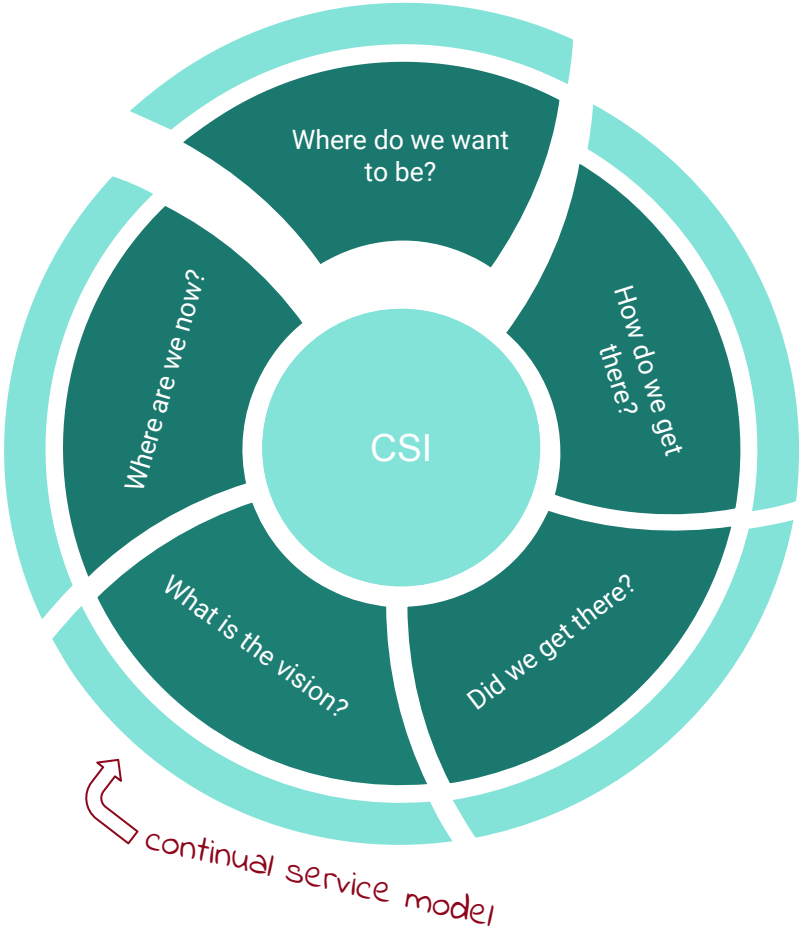
1. What is the vision?

In this stage, we figure out what do envision ourselves to be. Assuming we have built a spectacular product or service, how do we innovate from there? Should I rework what I currently have right now, subtract something or add something to my product?



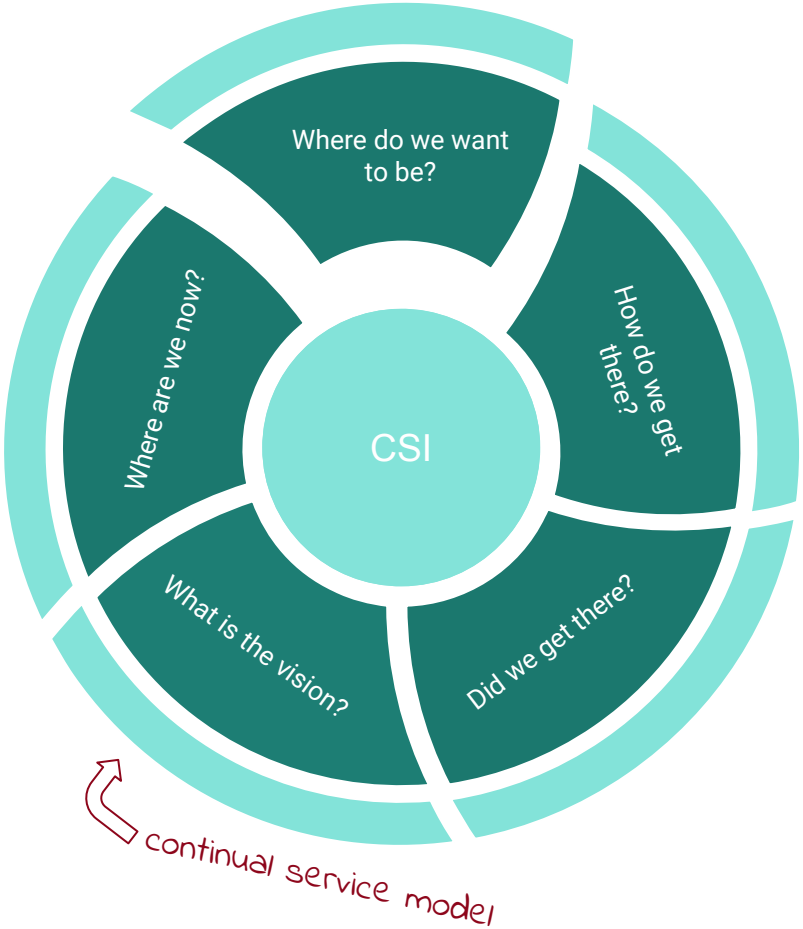
2. Where are we now?

In this stage, we determine what is the current state of our product or service. How are our people, our processes, our organization? From here, we'll get to know our baselines.



3. Where do we want to be?

In this stage, do we wanna be the very best? Like no one ever was? To be the number one is my real test, to continuously improve is my cause.



3. Where do we want to be?

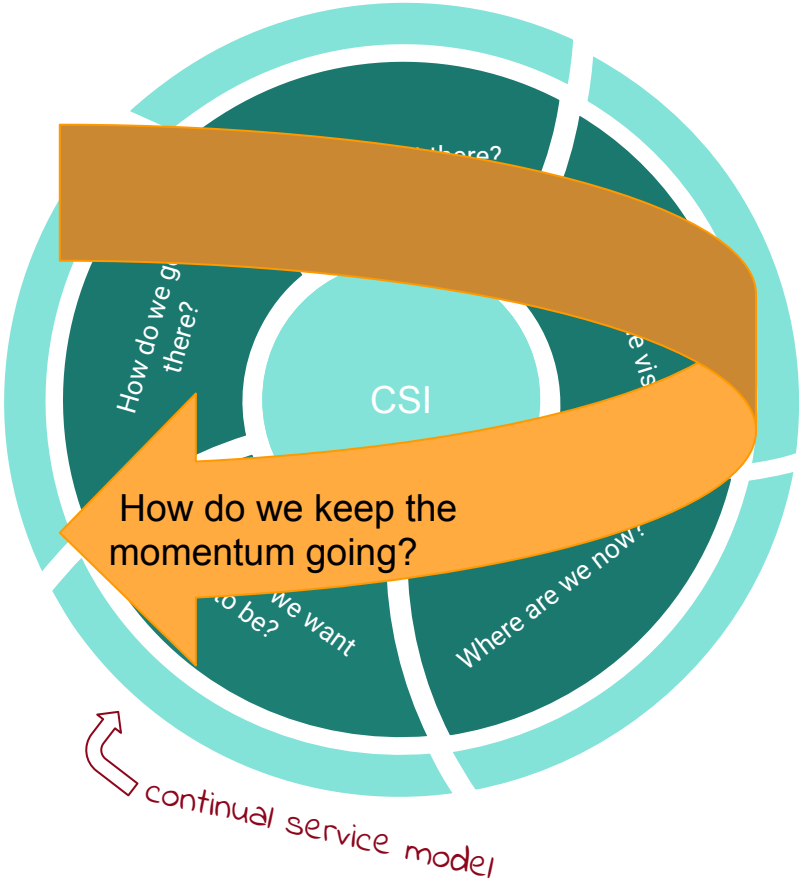
In this stage, do we wanna be the very best? Like no one ever was? To be the number one is my real test, to continuously improve is my cause.

4. How do we get there?

In this stage, we determine the improvement steps we need to get there. What can we do in the short term? Medium term? Long term? What metrics should we create and measure? What's gonna be our goals?

5. Did we get there?

Based from our baselines conducted in Stage 2, did we arrive to a point of where we wanted to be? Did we plan well? Did we build the right things? Did we build things right? What can we learn from our mistakes?



6. How do we keep the momentum going?

Let's capitalize on our strengths and wins. Do we need to readjust our strategy to meet our goals? Do we need to revise our plans to keep them current? How do we keep the momentum going?

but myron, daz cool and all

why do we need to continually improve our product or service?


Innovate or perish. There is no in between.
Solutions solve problems. Competition kills. Cash is oxygen. Revenue makes people happy.

so we get CSI already, how do we CSI our product?

through testing our product, that's how!

bugs

defects

improvements

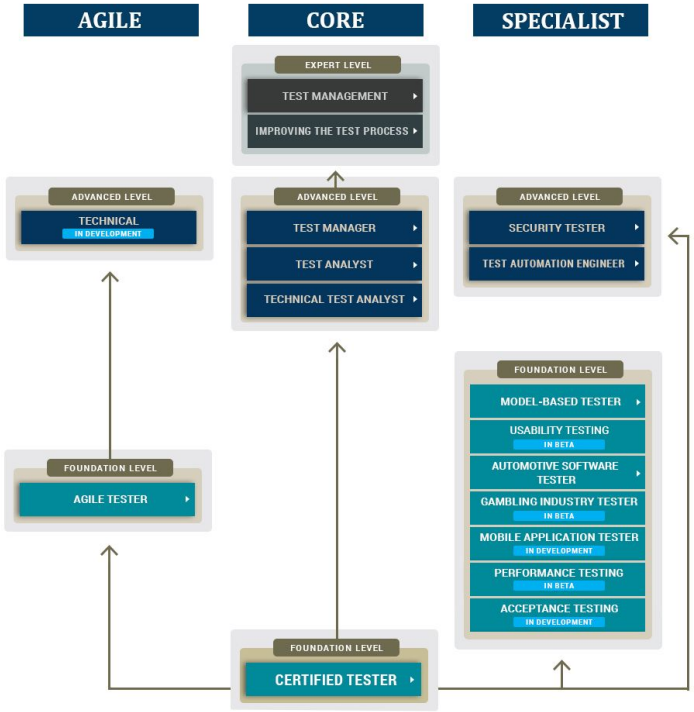
so, what do we get from testing our product?

quality

reliability

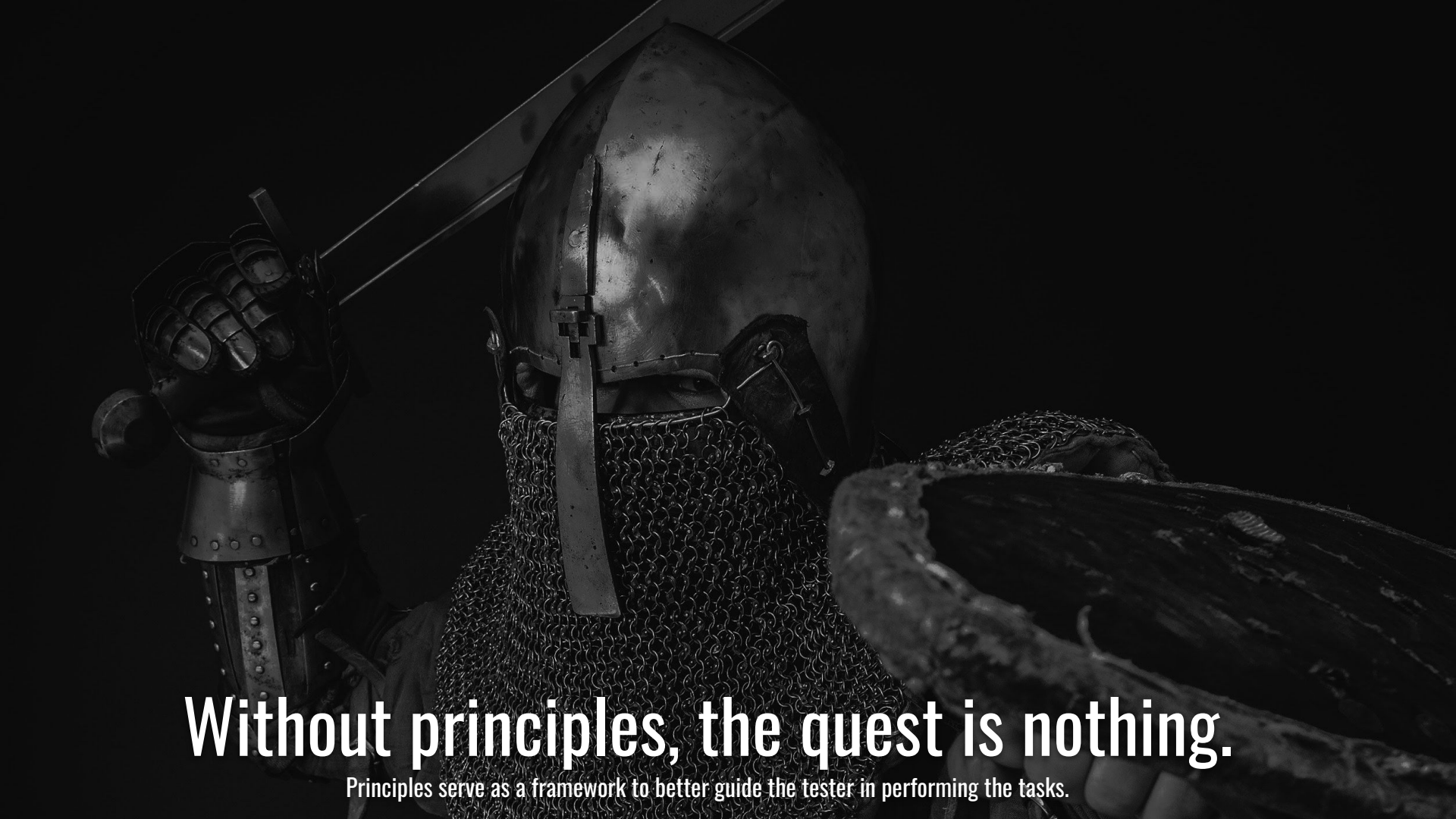
confidence

ISTQB (International Software Testing Qualifications Board) is an organization that aims to define and maintain a Body of Knowledge which allows testers to be certified based on best practices, connecting the international software testing community, and encouraging research.



so, why bother with principles anyway?

so that we could be guided in the quest
for producing high quality software



Without principles, the quest is nothing.

Principles serve as a framework to better guide the tester in performing the tasks.

what are the seven principles? esketit! ->

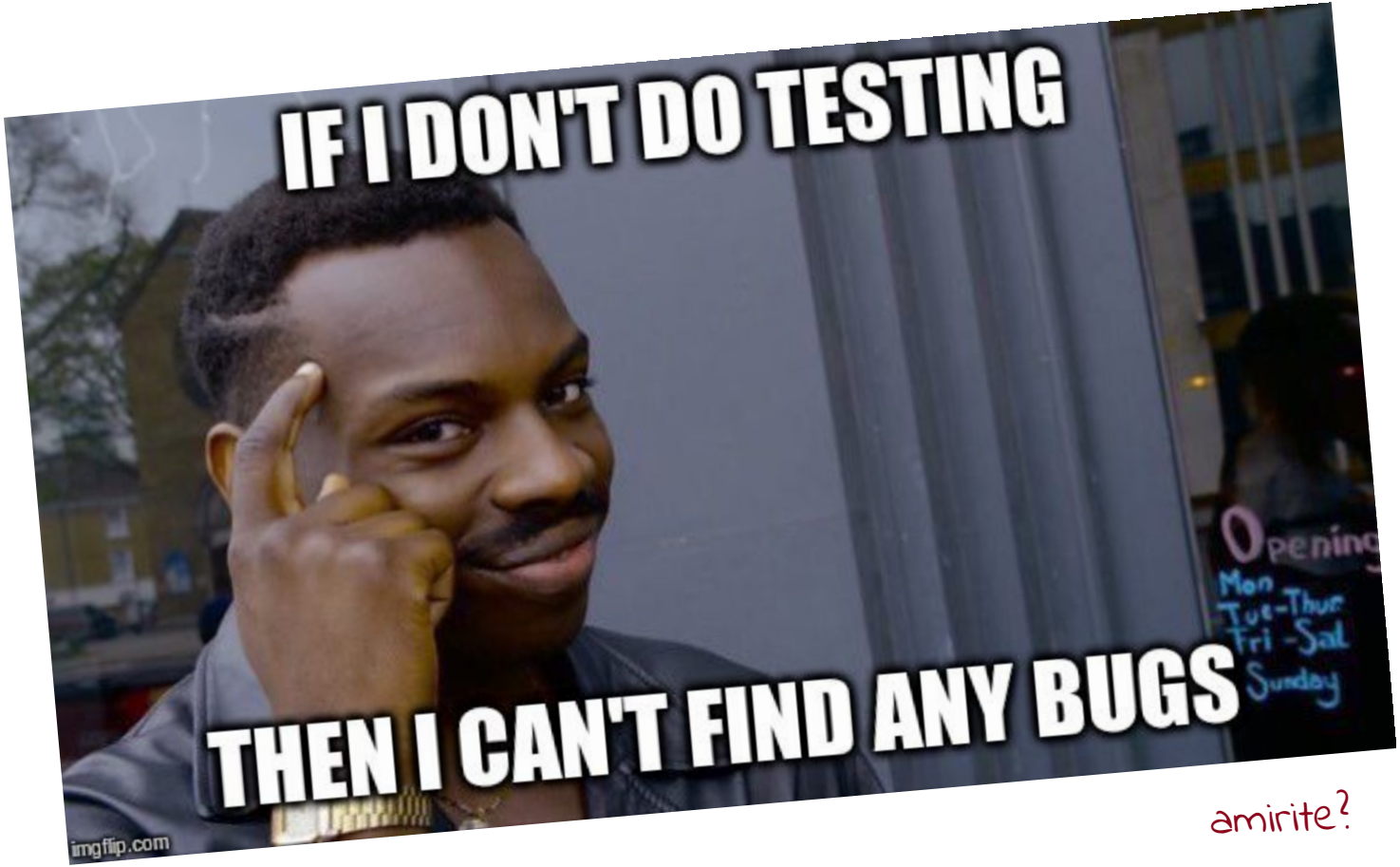
testing shows the presence of bugs

principle #1

Testing reduces the probability of undiscovered defects remaining in the software but, even if no defects are found, testing is not a proof of correctness.

testing shows the presence of defects

principle #1


IF I DON'T DO TESTING
THEN I CAN'T FIND ANY BUGS
Opening
Mon
Tue-Thur
Fri -Sat
Sunday
imgflip.com

amirite?

outages

lost data

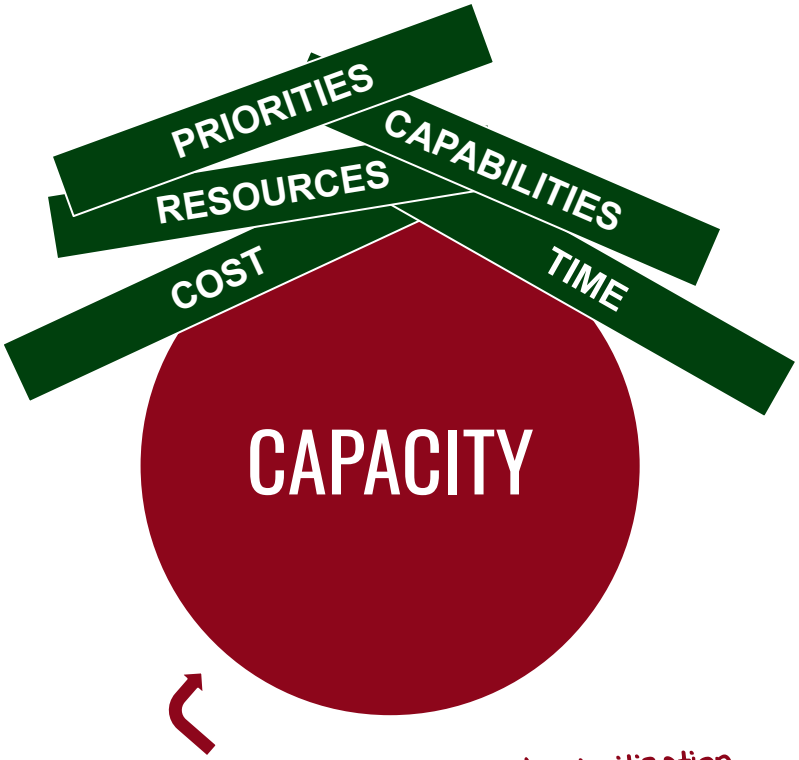
the cost of not testing

higher costs

bad reputation

exhaustive testing is impossible

principle #2



risk analysis, test techniques, and prioritization should be used to focus test efforts

exhaustive testing is impossible

principle #2

testing is context dependent

principle #3

testing is context dependent

principle #3

the absence of errors is a fallacy

principle #4

a bug-free system
can be still unusable :)

There is no such thing as a bug-free system or perfection.

the absence of errors is a fallacy

principle #4

prevention is better than the cure

early testing saves time and money

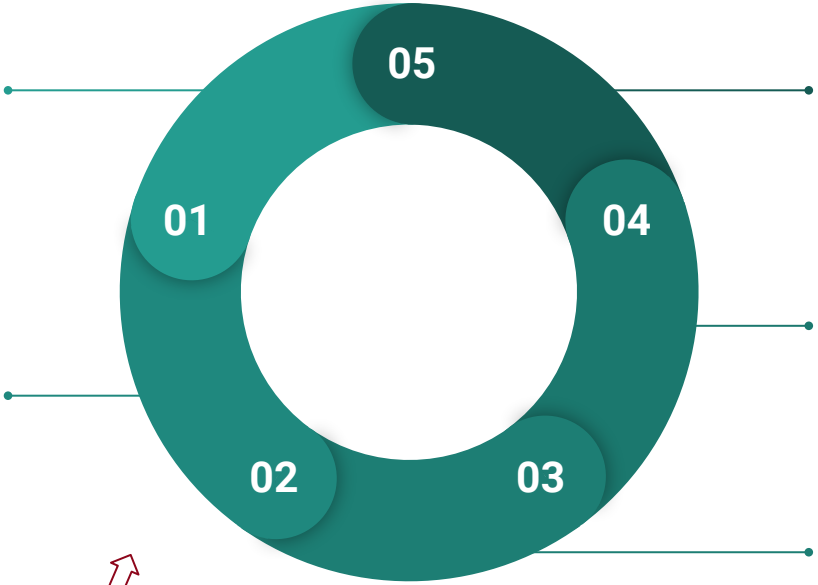
principle #5

Requirements Definition ($ 0)

In this stage, we define what the software outputs should be and which roles will use the system. Fixing mistakes here is easiest.

Design ($ 100)

In this stage, the software is being fleshed out and its look and feel is being drawn up.



a simplified version of the software development cycle, this time with arbitrary costs

Release ($ 100 000)

In this stage, fixing mistakes not caught by development team requires effort and cost to be corrected.

Testing ($ 10 000)

In this stage, errors and improvements are found. Bugs are discovered and triaged accordingly. Here we test the validity of output and verify the correctness of requirements.

Development ($ 1000)

In this stage, the software is slowly being realized and effort is exerted to create the software. Fixing mistakes at this stage requires effort and is prioritized accordingly.

EARLY TESTING SAVES MONEY

In 1962 NASA launched the Mariner-1 spacecraft to Venus. Mariner-1, which was launched from Cape Canaveral, veered off course because of a software bug. According to one of the reports, this accident was caused by the omission of a hyphen in the coded computer instructions or in some formula. As a result, Mariner-1 got wrong guidance signals. This error cost to NASA more than 18 million dollars.

defects cluster together

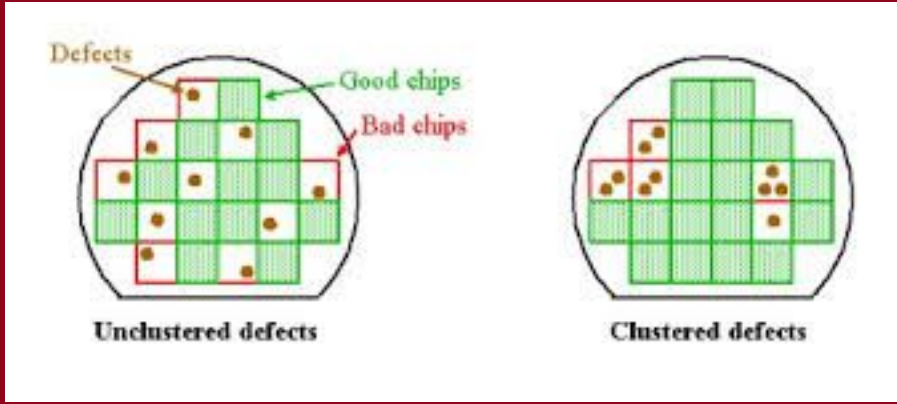
principle #6

a small number of modules usually contains most of the defects

80/20

Percent of defects

Modules in the software



pesticide paradox

principle #7

the same techniques may never uncover new bugs

Doing the same techniques when testing software is somehow insane.* To overcome this:

- We must update test cases on our ever-evolving product
- Set test cases that are relevant so as to not to increase our testing time
- Look for other ways to break software

*same techniques are totally okay for automated testing

pesticide paradox

principle #7

any questions?

Myron Angelo Dizon
Web Quality Assurance
Freelancer.com Inc. Philippines

</Continual Service Improvement & Principles In Software Testing>

Myron Angelo Dizon
Web Quality Assurance
Freelancer.com Inc. Philippines

The Life In The Service Management Industry
@ Engineering Conference Hall
March 1, 2019