DELACOR

Save Time, Money and Sanity with Unit Testing

Fabiola De la Cueva
Delacor

NATIONAL INSTRUMENTS Alliance Partner

NATIONAL INSTRUMENTS LabVIEW CERTIFIED ARCHITECT

NATIONAL INSTRUMENTS Certified Professional Instructor

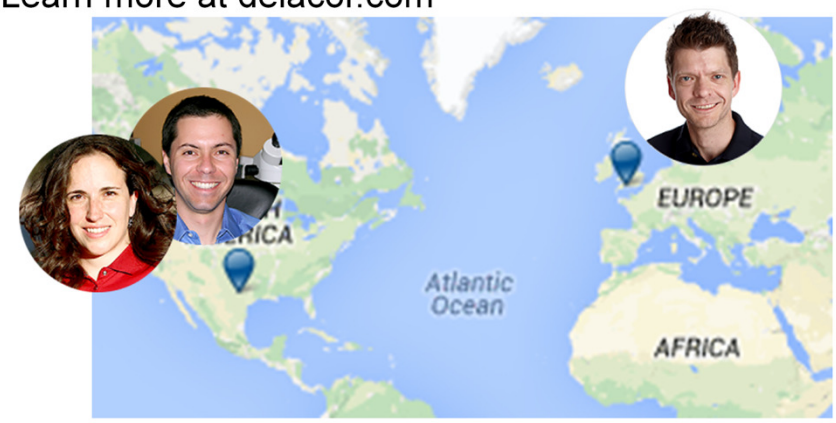
LabVIEW CHAMPION

DELACOR

About Delacor

Our focus is helping customers develop successful applications with LabVIEW and TestStand.

Learn more at delacor.com



DELACOR

Is this presentation for you?

- Requirements change in all your projects.
- You work on medium to large projects.
- Your small, quick one week projects come back for more features.
- Your team has multiple developers with different levels of proficiency.
- You believe in Unit Testing, but never find the justification to do it.

DELACOR

DELACOR

What is Unit Testing?

- Unit = Smallest testable part of an application
- Testing = Making sure the unit behaves as expected

- In LabVIEW:
  - Unit = a VI, a class, an Action Engine, etc
  - Testing = for a given set of value(s) entered for control(s), the outputs are equal to the expected outputs.
  - For a given stimulus get expected response

DELACOR

Definition

- Unit Test
  - An automated piece of code that invokes the unit of work being tested, and then checks some assumptions about a single end result of that unit. A unit test is almost always written using a unit testing framework. It can be written easily and runs quickly. It's trust-worthy, readable, and maintainable. It is consistent in its results as long as production code has not changed.

artofunittesting.com Roy Oshervore

DELACOR

Properties of a good unit test

- Should be automated and repeatable
- Should be easy to implement
- Should be relevant tomorrow
- Anyone should be able to run it at the push of a button
- Should run quickly
- Should be consistent in its results
- Should have full control of the unit under test
- Should be fully isolated
- When it fails, it should be easy to detect what was expected and determine how to pinpoint the problem

artofunittesting.com Roy Oshervore

DELACOR

Integration Tests

- Tests that aren't fast or consistent
- Use real system time, real file system real database, etc
- Test via the final UI
- When test fails, it is difficult to pinpoint where the problem is

artofunittesting.com Roy Oshervore

DELACOR

Integration vs Unit Tests

- Integration Tests
  - Use real dependencies
- Unit Tests
  - Isolate the unit of work from its dependencies so that they are consistent in their results and can easily control and simulate any aspect of the unit's behavior.

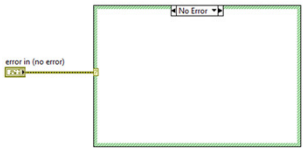
artofunittesting.com Roy Oshervore

DELACOR

Types of Unit Testing

- Black Box:
  - Outputs = Expected outputs?
  - Used for acceptance tests

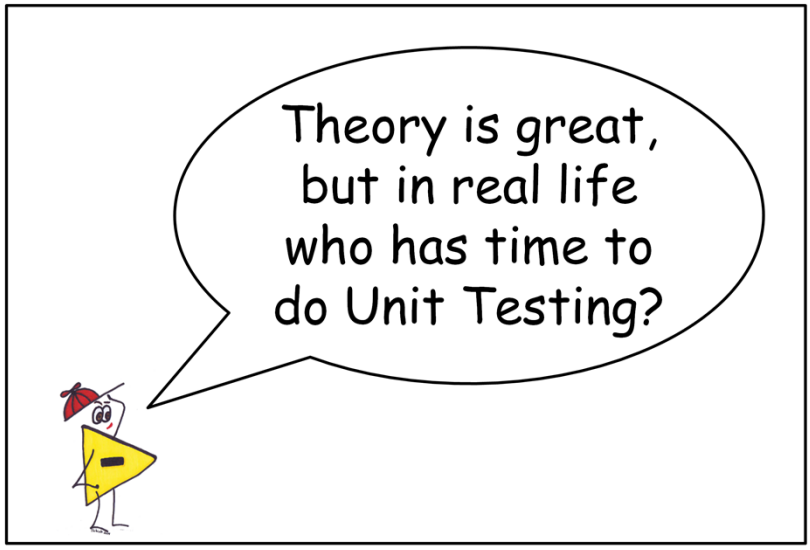
- White Box:
  - Outputs = Expected outputs?
  - Aware of which block diagrams



Question for audience, how many block diagrams do we have in the VI shown. Answer: 3, the UTF will report this as 3 block diagrams: 1 for the error case, 1 for the no error case and one for the VI itself.

Black box vs White box / Clear box / glass box.
White-box testing (also known as **clear box testing**, **glass box testing**, **transparent box testing**, and **structural testing**) is a method of testing software that tests internal structures or workings of an application, as opposed to its functionality (i.e. black-box testing).
More can be found at: http://en.wikipedia.org/wiki/White-box_testing

DELACOR


Theory is great,
but in real life
who has time to
do Unit Testing?

DELACOR

Making the Case

DELACOR

Sample Case

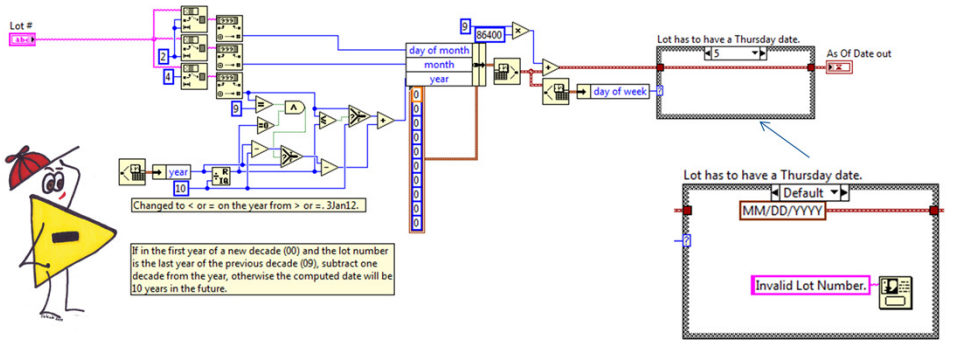
- Requirement:
  - Lot number is of the form mmddyy (month day year).
  - Calculate and return the expiration date for that lot by adding 9 days of decay.
  - All lot numbers have a Thursday date.

I will present a sample code that meets this requirement.

DELACOR

Solution from a CLAD

- Tried to reinvent the wheel and ended up with a flat tire.

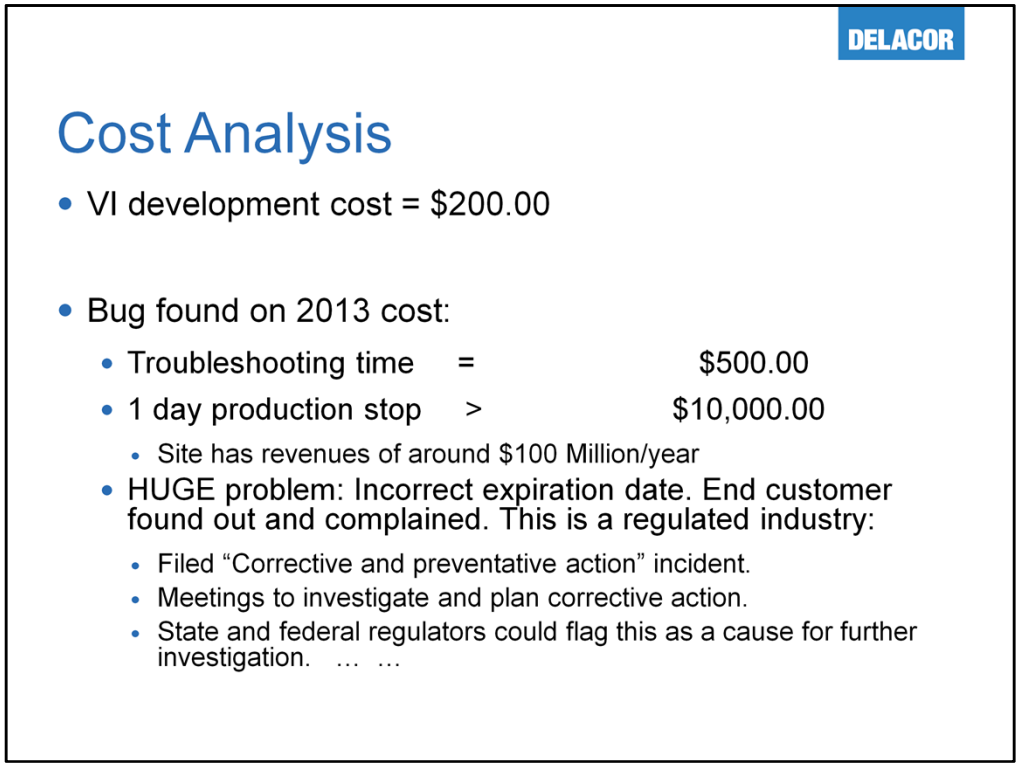


This is to show a complicated way of solving the requirement. This will be used as a case study. The code was created around 2008 and it worked well until 2013 new year. Or customer found that it was not really working until that time, it might have failed before and nobody had noticed.
Yes, it would be better to use scan from string, but stay with me on this one.

DELACOR

Background

- The code was created around 2008.
- It took about 2 hours to write, including testing with a couple of known lot dates.
- Worked well until January of 2013.
- By then original developer was gone.
- It took about 5 hours to find that other problems in another area of the code was due to this VI.
- Had to call LabVIEW consultant to troubleshoot + fix.
- Had to stop production for one day + Client very upset + this regulated environment.



The main takeaway is that investing time during development to test can save you a lot of money in the long run.

From my customer:
"The cost of lost production isn't quite right since customer's markups are so high but you can use $10,000 for this presentation. The HUGE issue is the incorrect expiration date (calibration date) on the label. The end customer complained, we had to execute a "corrective and preventative action" incident which so far has entailed 1 meeting of 5 people, a new pc and about 3 hours of my time, 3 follow-ups to make certain the corrective actions were taken and 3 explanations as to why the actions hadn't happed already. This is a regulated industry and mistakes are taken very seriously. In fact, the state and federal regulators could flag this incident as a cause for further investigation or a black mark of some sort. We sell radioactivity which freaks out a lot of people. Enough black marks and we get shut down. The site has revenues of around $100 million a year. That's where you might get the attention of your listeners."

DELACOR

Compared to

- VI development + Unit Test cost = $500.00

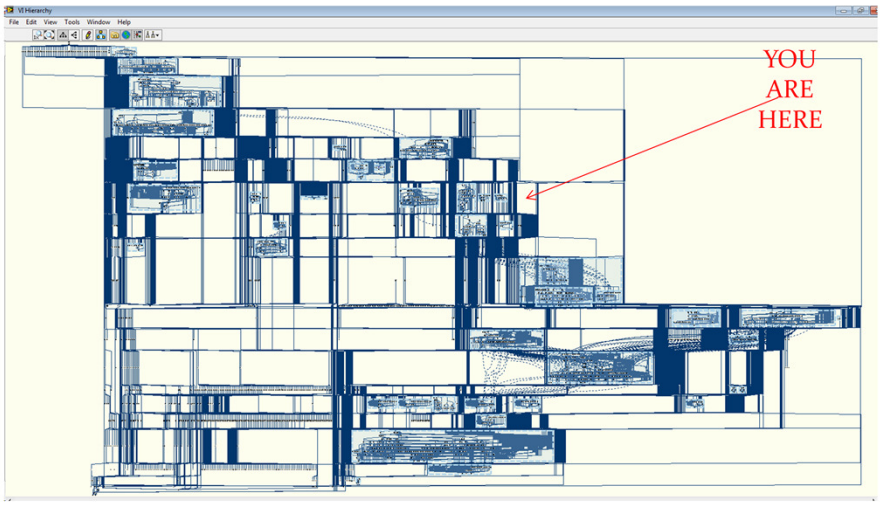
We should do Unit Tests not to get a check mark on a process checklist or use a cool tool.

We should do Unit Tests because they save money and time in the long run.

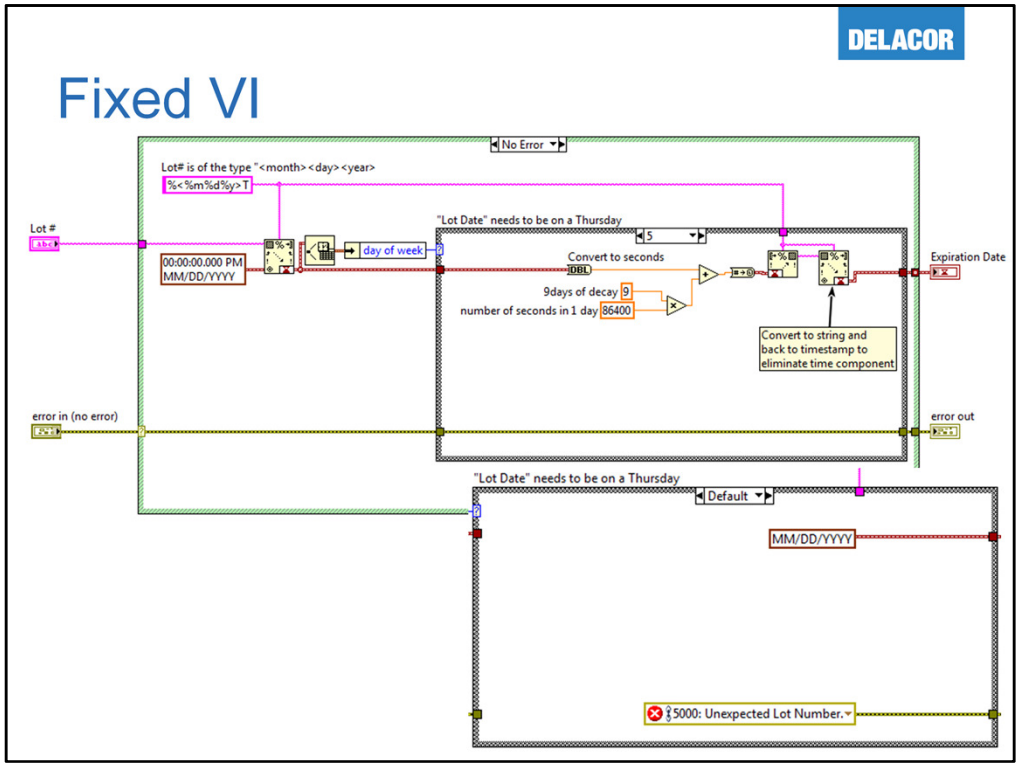
Because it is the right thing to do.

DELACOR

That was ONE VI in the project

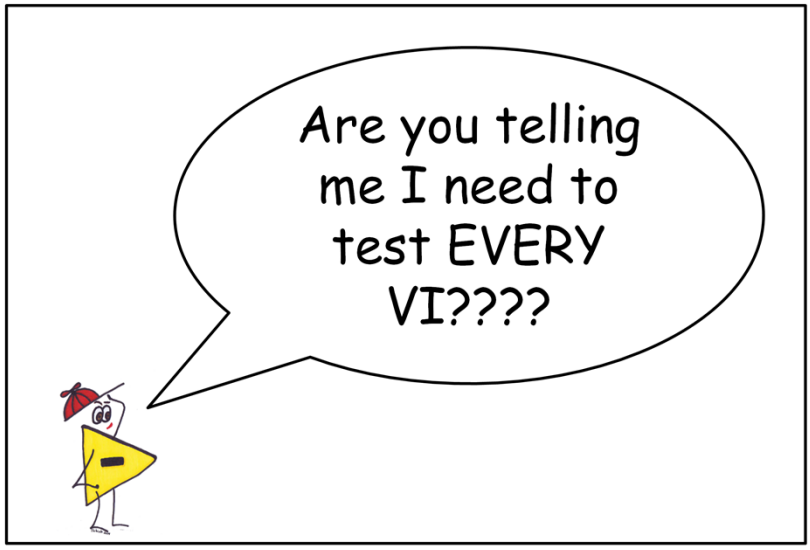


Potential savings?



We removed the dialog box, instead generating an error.
The input output pairs are going to be tested later using unit tests.

DELACOR

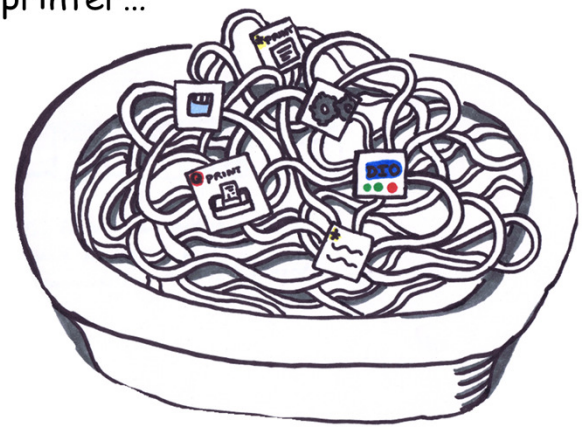


DELACOR

What to Unit Test?

DELACOR

If you have to run your entire application in order to test your printer…

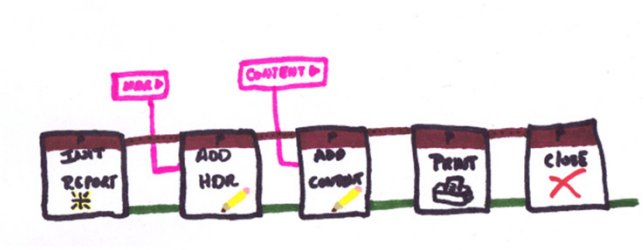


Your development process needs to be adjusted.

The red Q in the Print VI represents a cluster with all the queues used in the entire code. Unfortunately, this is based in a customer's real code.

DELACOR

Consider instead creating test code that:



- Can be used for testing
- Shows how the API is used
- Documents how API is to be used

DELACOR

What to Test?

- Plan ahead to create Unit Tests for
  - Critical code
  - Complex algorithms
- Add Unit Tests as project progresses for

  - As bugs are found that could have been caught via unit tests
  - Regression tests to verify that new method returns same results as old method
- Create Integration Tests for every module API
  - Show/document with code how API is to be used

While developing API, thinking of the Unit tests can help developer verify not only that the API works but also that it is well designed. The API has to be seen from the “customer” perspective.

DELACOR

Steps to develop a simple VI

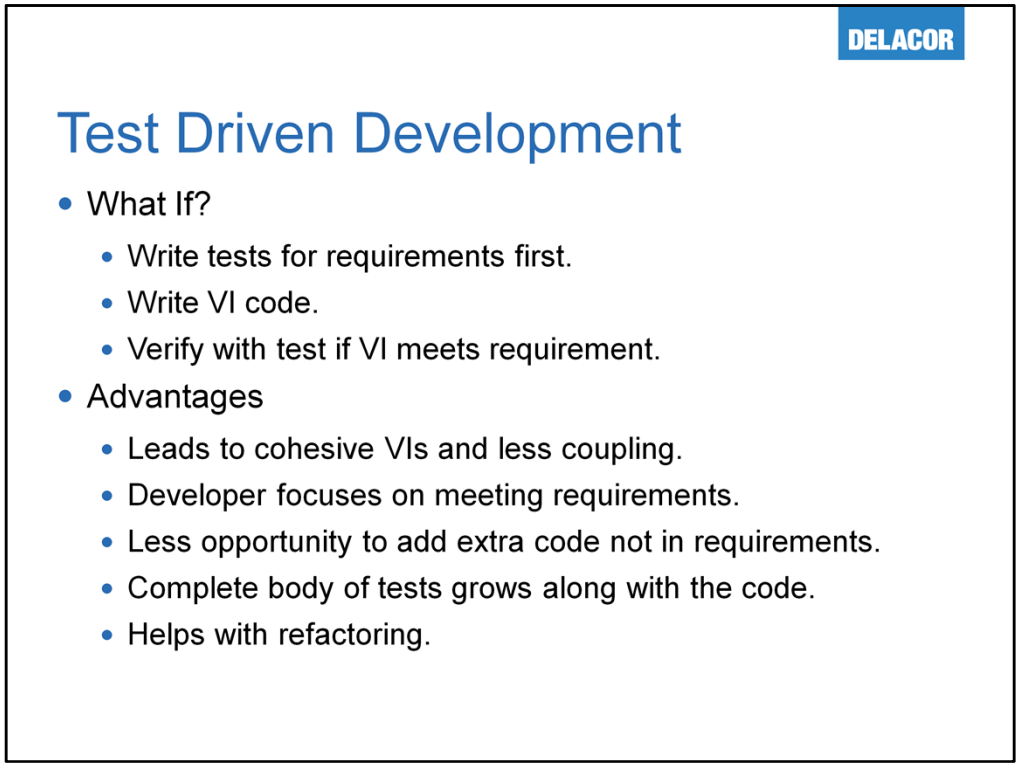
- Set controls and indicators
- Write VI code
- Enter a value on input
- Run VI
- Verify that output is correct
- Move on

DELACOR

Steps to develop a complex VI

- Same as before, maybe add a VI to test multiple inputs and compare results.

- Questions to audience:
  - Do you keep these test VIs?
  - Do you run them ever again?



All code should be written in order to make a failing unit test pass. At the beginning of the iteration, the unit test is designed and of course when ran it fails, because the function does not exist.
Then the actual functionality is programmed and the unit test is ran to verify that the implementation worked correctly.

Agile Software Development Principles, Patterns, and Practices

Robert C. Martin

DELACOR

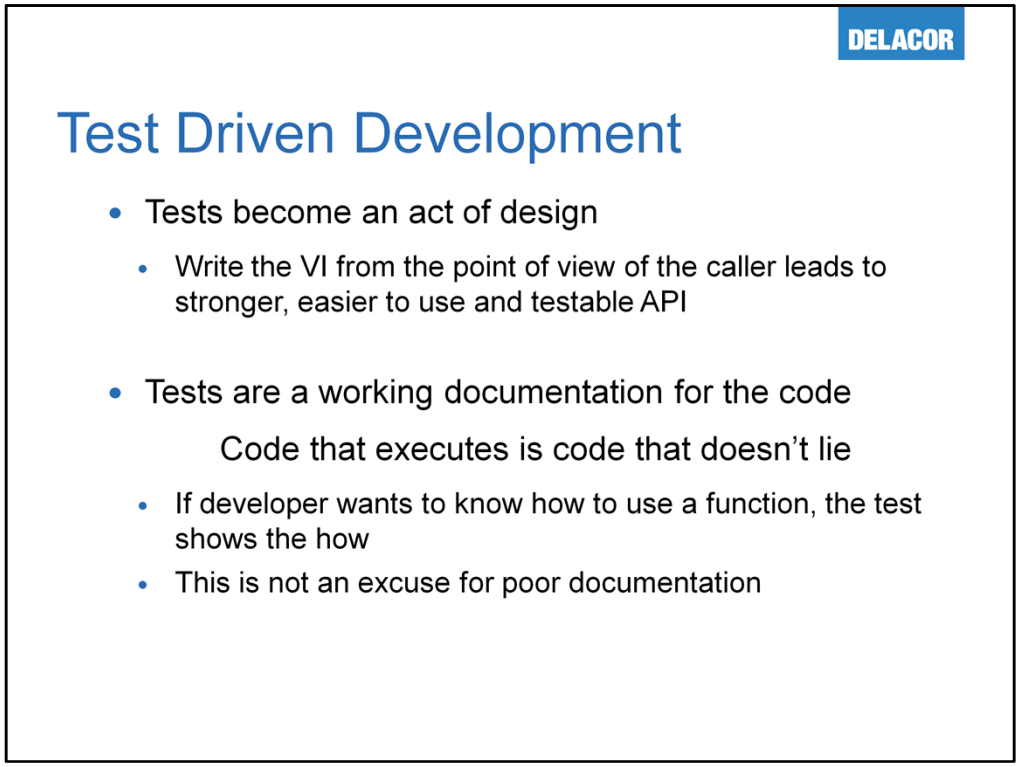
Agile Software Development Principles, Patterns, and Practices by Robert C. Martin

"All production code is written in order to make failing unit tests pass. First we write a unit test that fails because the functionality for which it is testing doesn't exist. Then we write the code that makes that test pass."

All code should be written in order to make a failing unit test pass. At the beginning of the iteration, the unit test is designed and of course when ran it fails, because the function does not exist.
Then the actual functionality is programmed and the unit test is ran to verify that the implementation worked correctly.

Agile Software Development Principles, Patterns, and Practices

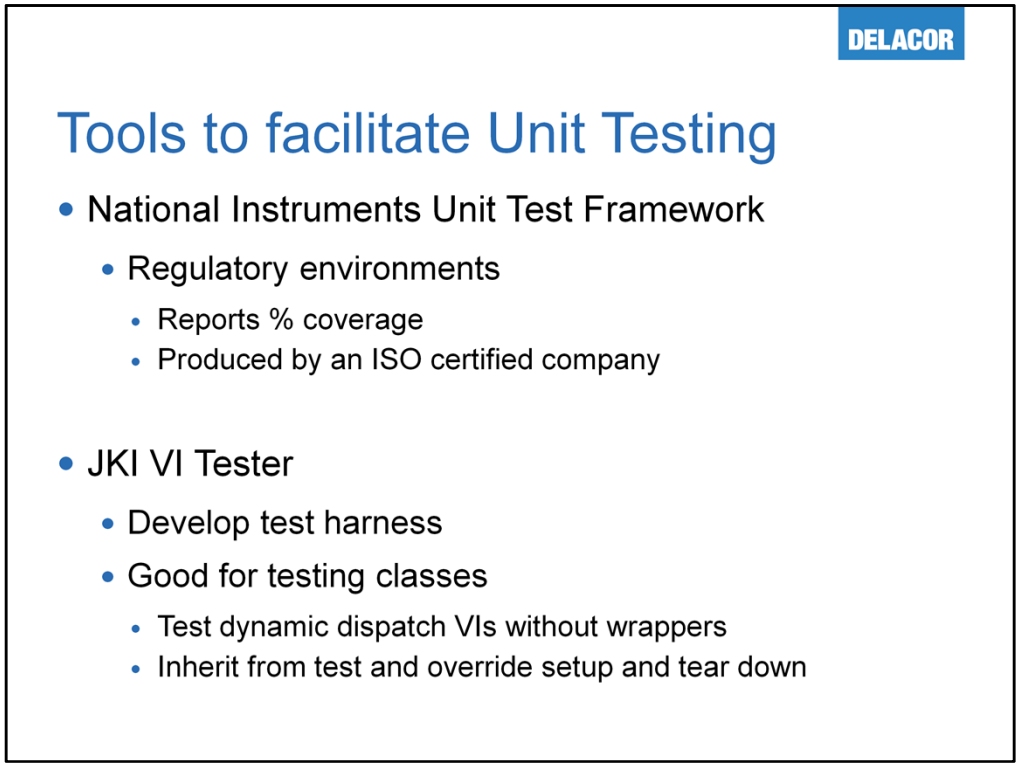
Robert C. Martin



All code should be written in order to make a failing unit test pass. At the beginning of the iteration, the unit test is designed and of course when ran it fails, because the function does not exist.
Then the actual functionality is programmed and the unit test is ran to verify that the implementation worked correctly.

Agile Software Development Principles, Patterns, and Practices

Robert C. Martin

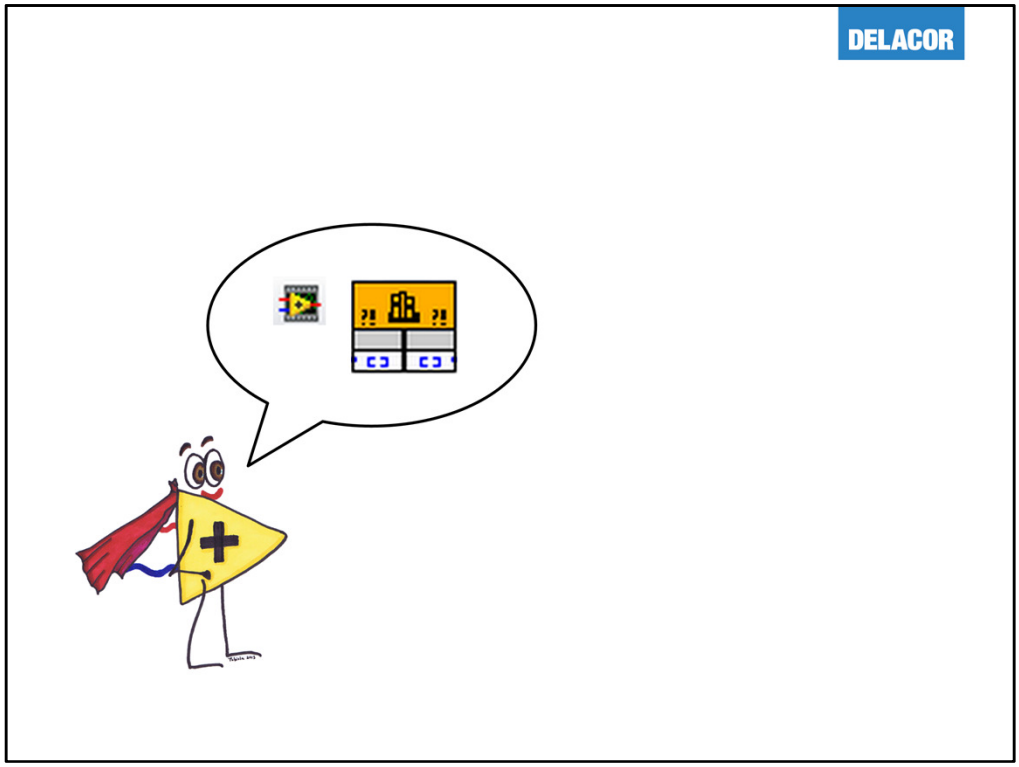


UTF: It's produced by an ISO certified company, so that means the user doesn't have to certify it to use it on projects where regulatory compliance is required.
From Christopher Relf: If you want to use a tool that's not by an ISO certified company:
* You must certify it yourself (or have someone else certify it)
* Whoever certifies it must be ISO 9001 certified
* Another great reason to use LabVIEW and TestStand - they were also created by an ISO 9001 certified company

DELACOR

Agile Software Development Principles, Patterns, and Practices by Robert C. Martin

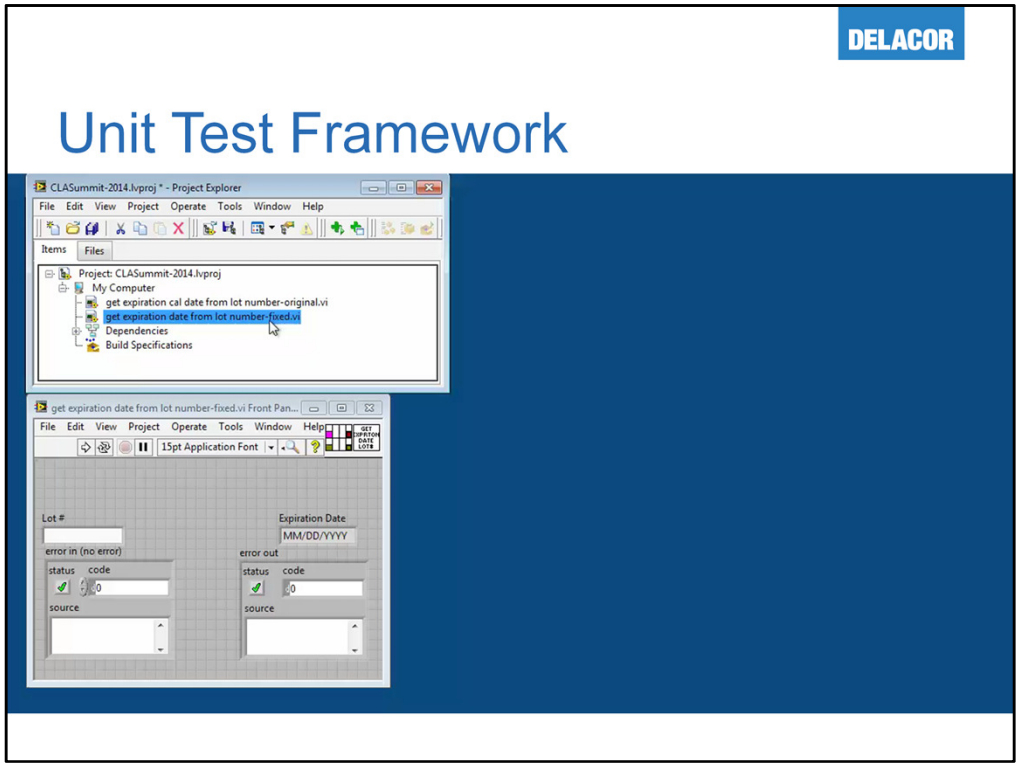
- "The simpler it is to run a suite of tests, the more often those tests will be run."
- "The more the tests are run, the faster any deviation from those tests will be found."
- "The more testable your code is, the more decoupled it is."



Story about a colleague who is bilingual, he is fluent both in C and LabVIEW. He did a dll in C that had to have the same behavior as LabVIEW code I had written. I asked him to use my unit tests to verify that his dll wrapped in LabVIEW dll calls would pass the unit tests. That worked very well, he knew when he was done and we could demonstrate to my customer that both methods returned the same results.

DELACOR

Demonstrations

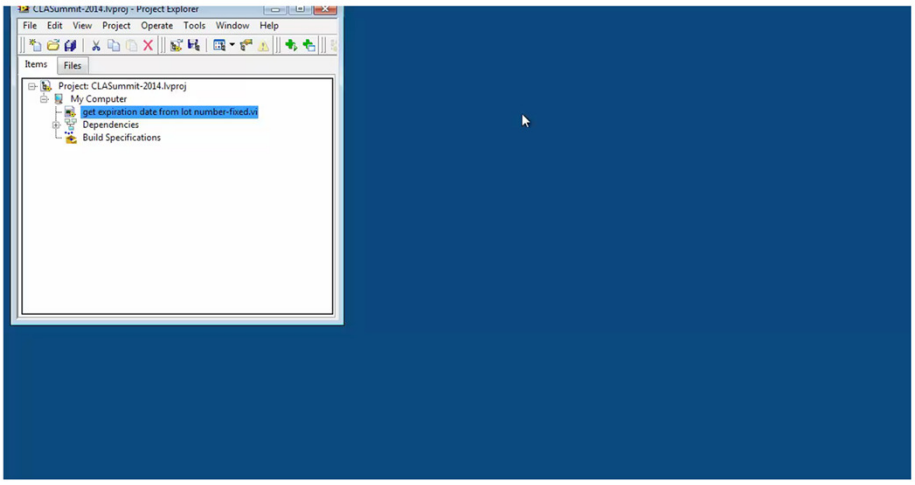


Demo unit test for the Lot VI.

http://www.youtube.com/watch?v=DjlrPjceEdU&feature=youtu.be

DELACOR

JKI VI Tester



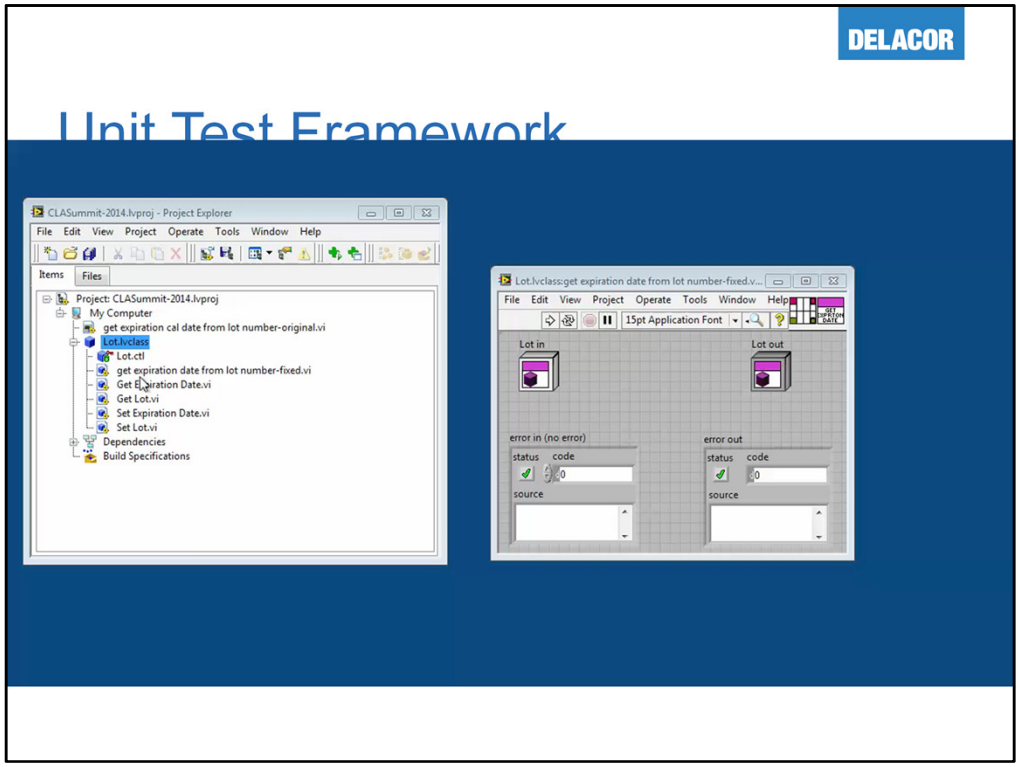
Demo unit test using JKI VI Tester for Lot VI.
http://youtu.be/Ec1GOY_Cgxc

the name TestCase has to be in the Test Case name (VI Tester finds Test classes by inspecting their class inheritance and verifying they inherit from TestCase base class). The key is that the test method names must begin with "test" since there is no way to verify that a test method is intended to be executed vs a test class support VI. The other key - the lvproj file and the lvclass file should be saved prior to opening the VI Tester UI.

DELACOR

What if Lot# is part of a Class?

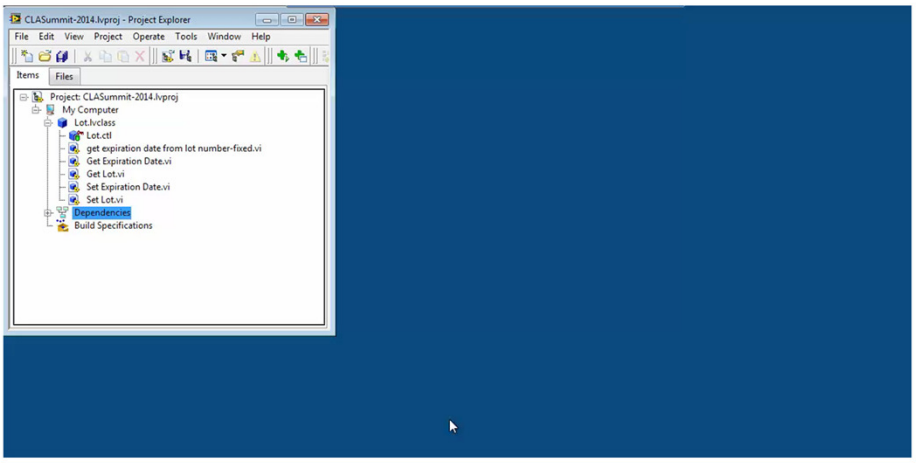
- Setup VI
- VI under test
- Tear Down



Demo unit test using setup and tear down
http://www.youtube.com/watch?v=1bMKFKNiNBw

DELACOR

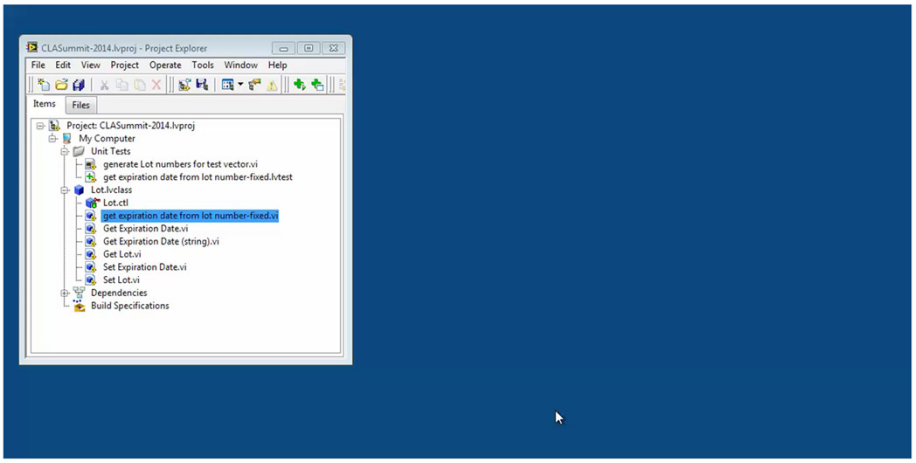
JKI VI Tester



Demo unit test using setup and tear down
http://youtu.be/YYPlYCnSLYo

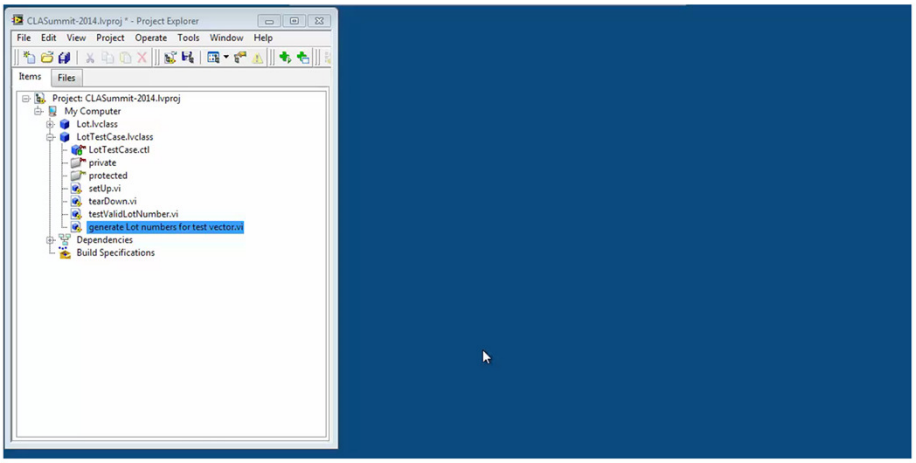
DELACOR

Test a value range



Explain and demonstrate how to create test vectors in Unit Test Framework
http://www.youtube.com/watch?v=i8XHf2MrZdM

DELACOR

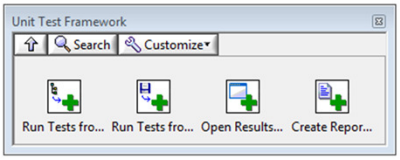


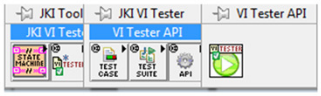
Demo value range in JKI VI Tester
http://www.youtube.com/watch?v=CfVnYtBEqAE

DELACOR

Unit Test Automation

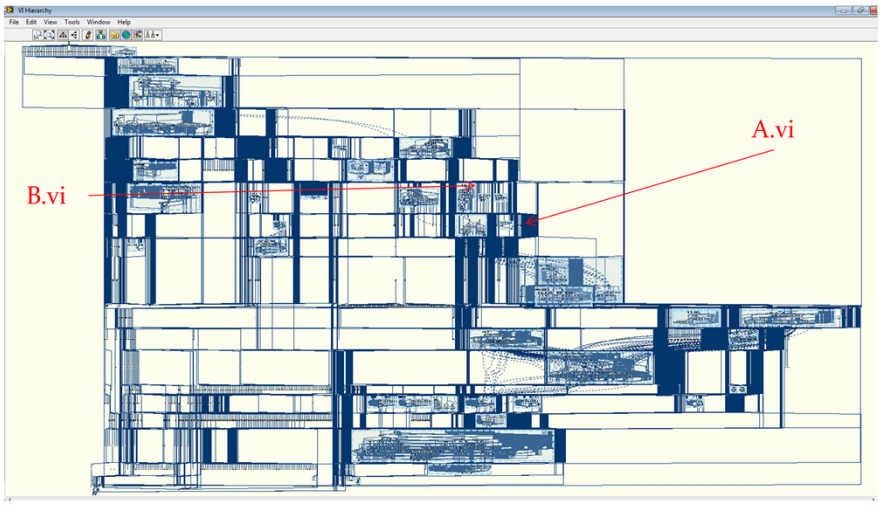
- API available to run unit test





DELACOR

Now if change in B causes A to fail, you will know



Potential savings?

DELACOR

What About Connections?

- Database, file I/O, peripherals, TCP/IP, etc.
- This is considered integration testing.
- Setup and Teardown can be used to manage the connections.

DELACOR

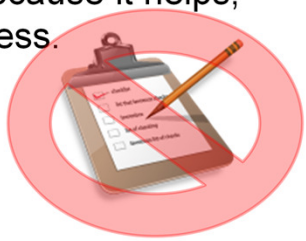
Conclusion

- Choose a Unit Test tool from the beginning and create tests every time you find yourself testing individual values on the front panel of a VI.
- When a bug is found, if it could have been caught by an Unit Test, go back and create a Unit Test.
- If a bug is found that was not found by Unit Test, go back and improve your Unit Test.
- It is better and more efficient to plan and do Unit Tests from the beginning (as opposed to going back at the end).
- Make Unit Test part of your process.

DELACOR

Our Goal

- Working software with minimum number of bugs
- Adapt to changing requirements without breaking existing code
  - Identify what code breaks when changing requirements
- Unit Testing will get us there. Do it because it helps, not just because it is part of the process.



DELACOR

To learn more

- NI.com /groups Unit Testing Group
  - LabVIEW Architects Forum
- Managing Software Engineering in LabVIEW course
- ni.com/largeapps Software Engineering Technical Manual and Exercises for LabVIEW: bit.ly/lv_swe
- jamesmcnally.co.uk/testable-code-labview

DELACOR

Thanks