



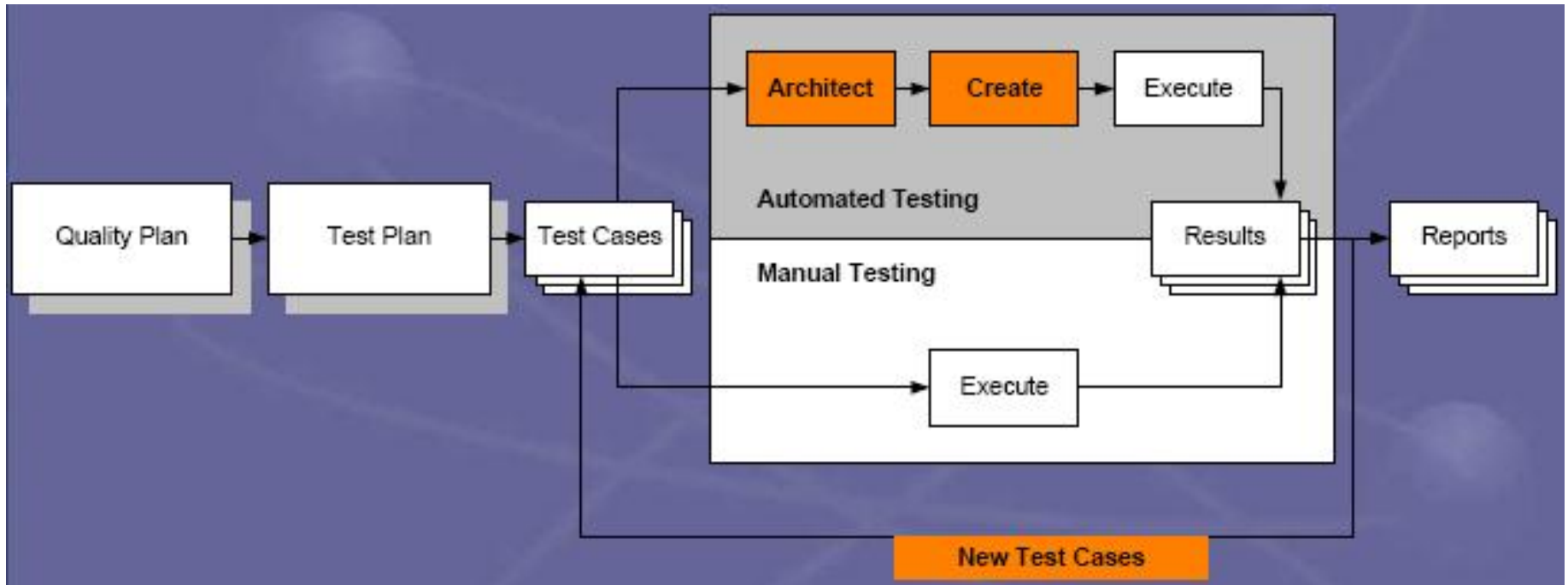
# THE FIRST CONFERENCE ON E-BUSINESS RELIABILITY

## Test Planning for E-Business Applications – Improving ROI

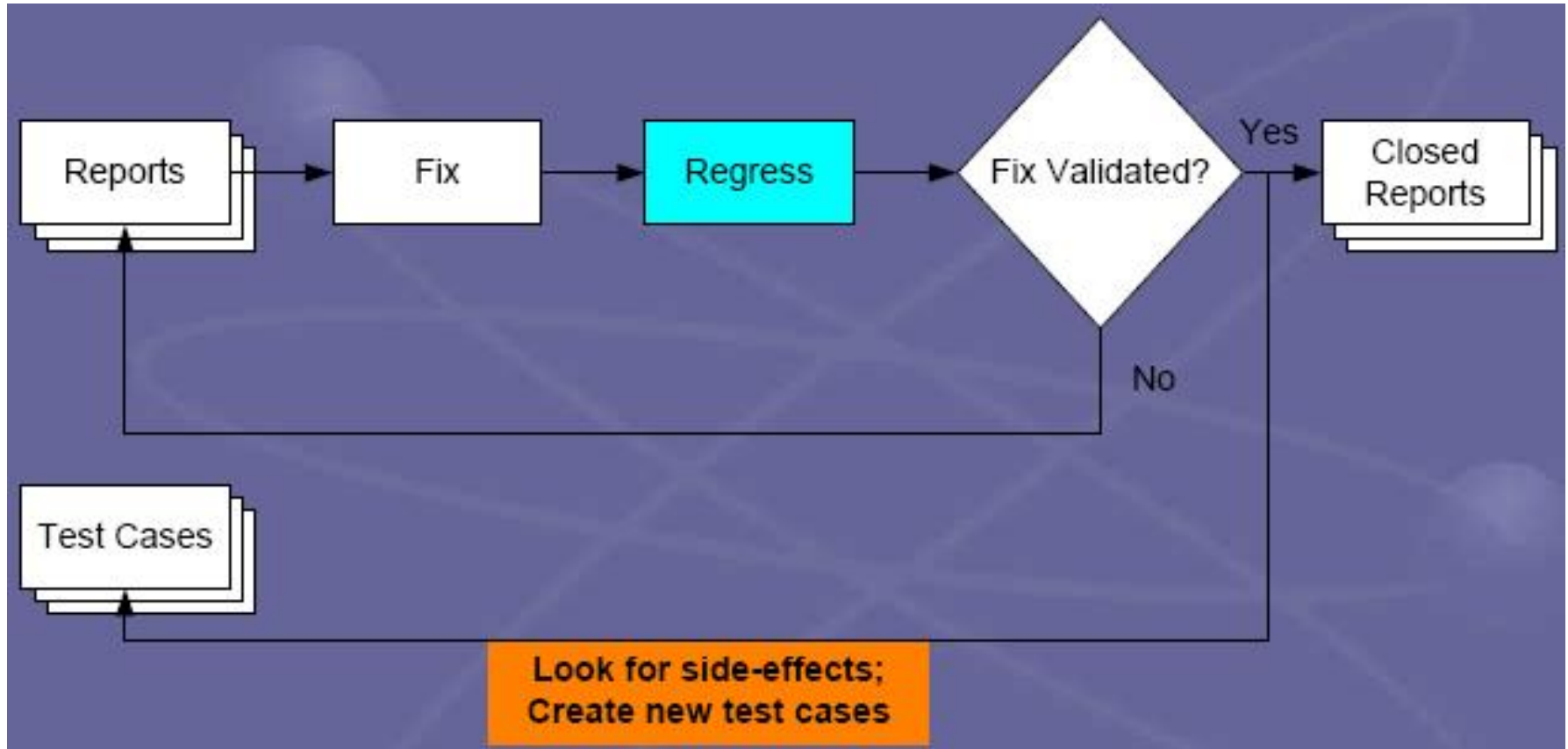
**John W. Green**  
**Automation Expertise**  
<http://www.automationexpertise.com>

**Hung Q. Nguyen**  
**LogiGear Corporation**  
<http://www.logigear.com>

# The Testing Process



# Bug-Fix Regression: Manual Testing is More Effective



- **Is the test project under schedule constraints?**
- **Is the objective of the testing effort to maximize test coverage with limited resources under time constraints?**
- **Is the goal to find as many defects as possible or to save on testing costs while maintaining comparable testing productivity?**

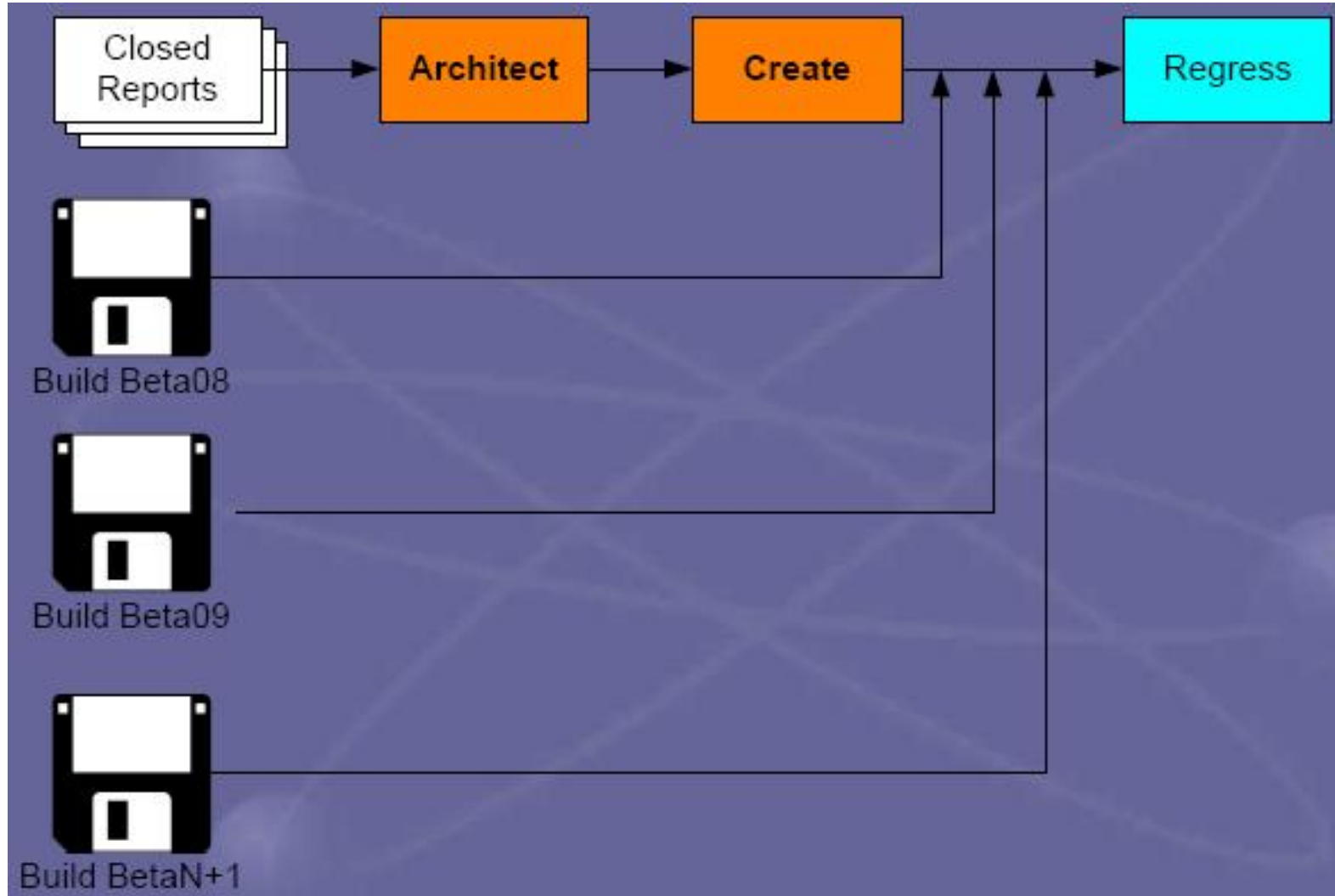
- **Are well-defined steps available for each test case?**
- **Are expected results pre-defined?**
- **Are methods of identifying errors pre-defined?**
- **Are methods of analyzing errors pre-defined?**
- **Will the tests be rerun? If yes, how many times?**
- **Can manual testing be done cost-effectively?**
- **Can the test cases be automated quickly?**
- **Is the UI likely to change?**
- **Is the test data known and available?**

# Manual Vs. Automated Testing: Pros and Cons

	<b>Pros</b>	<b>Cons</b>
<b>Manual Testing</b>	Test case creation is quick and inexpensive.	Might not be consistent in rerunning test cases.
	Better at simulating real world use.	Rerunning large volumes of test cases is expensive.
	Better at analyzing test results and exploring new test cases.	
	Best when test steps and results are not well-defined.	
	Best when extensive analysis is required.	
	Does not require a technically trained staff.	
<b>Automated Testing</b>	Can execute test cases unattended.	Test case creation is expensive.
	Can cost-effectively run large volumes of test cases.	Test case creation is time consuming.
	Can cost-effectively rerun large volumes of test cases repetitively.	Maintaining test cases can be expensive.
		Requires a highly technical staff.
		Rerunning old tests is not likely to find new bugs.

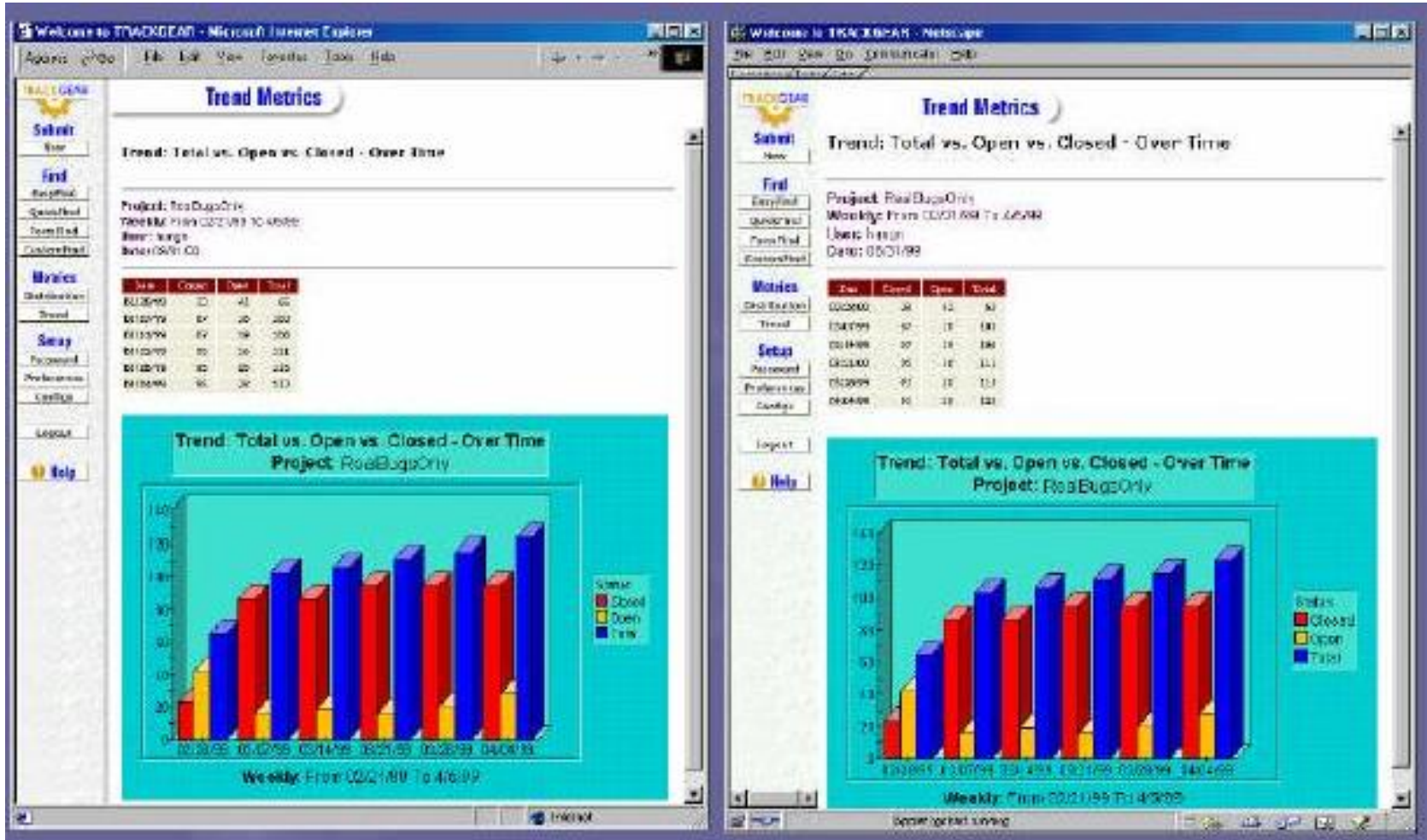
- **Release Acceptance Regression Tests -- See example**
- **Compatibility Tests -- See Example**
- **Configuration Tests -- See Example**
- **Release Acceptance Feature Tests**
- **Functional Acceptance Simple Tests**
- **Stress Tests**
- **Performance Tests**

# Release Acceptance Regression: Consider Automated Testing



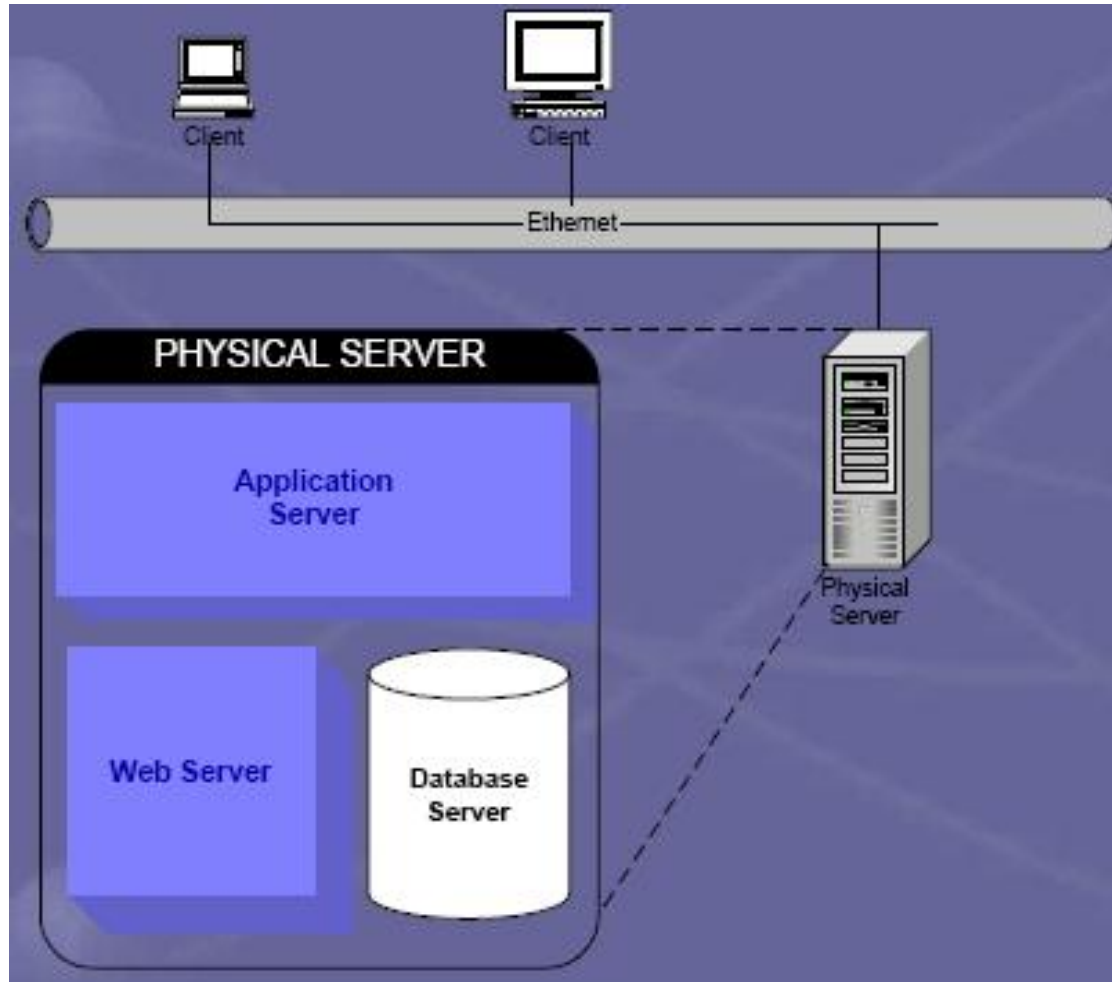
- **Release Acceptance Regression Tests -- See example**
- **Compatibility Tests -- See Example**
- **Configuration Tests -- See Example**
- **Release Acceptance Feature Tests**
- **Functional Acceptance Simple Tests**
- **Stress Tests**
- **Performance Tests**

## Executing the same Client-based test suites on different Browsers--IE/NN

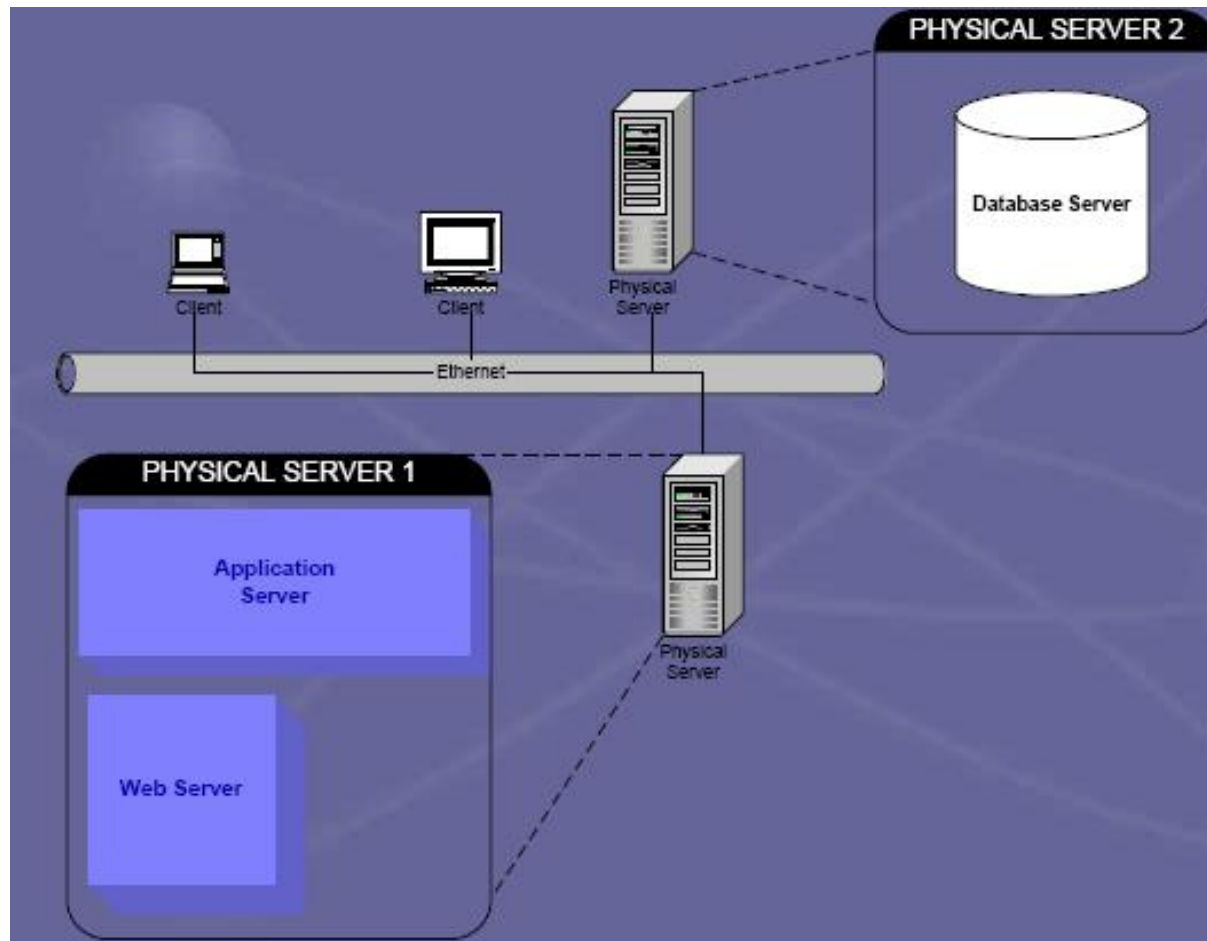


- **Release Acceptance Regression Tests -- See example**
- **Compatibility Tests -- See Example**
- **Configuration Tests -- See Example**
- **Release Acceptance Feature Tests**
- **Functional Acceptance Simple Tests**
- **Stress Tests**
- **Performance Tests**

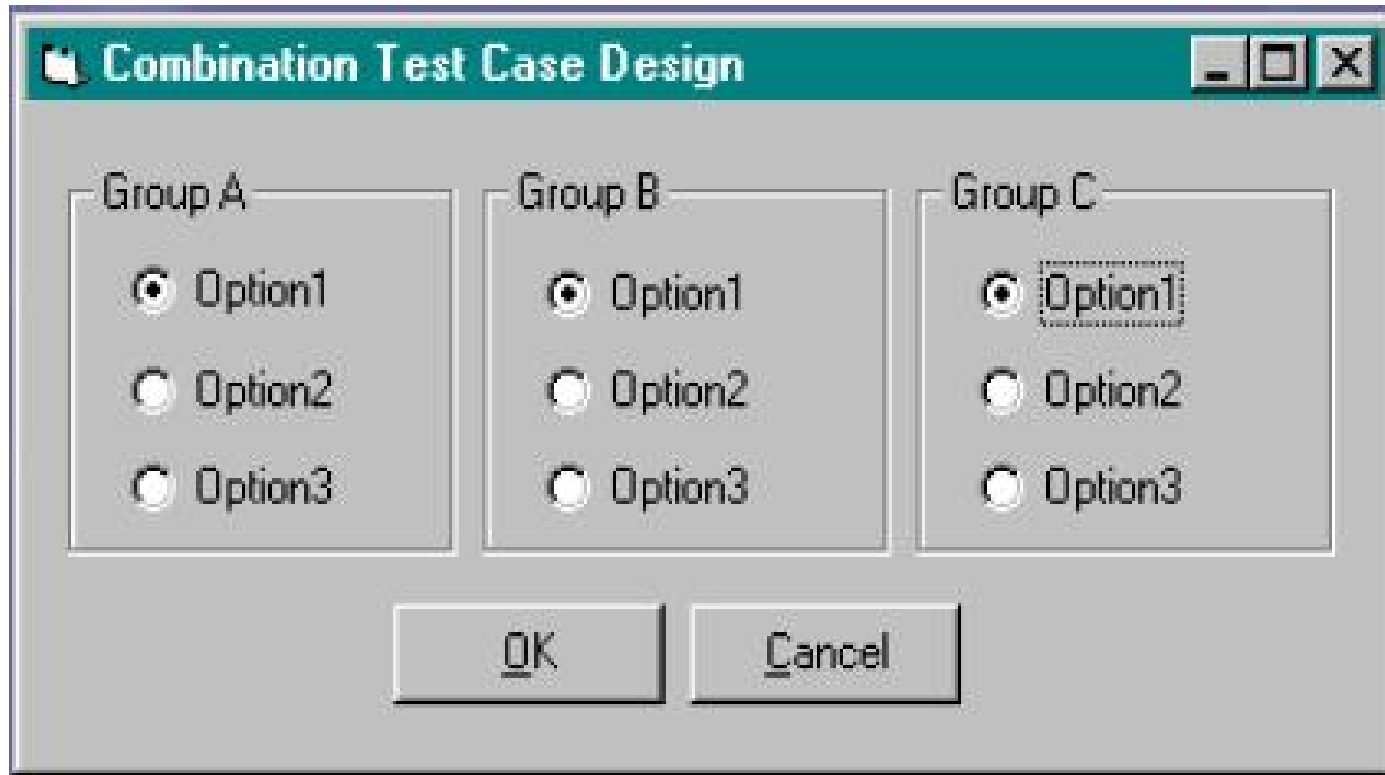
## Executing the same Client-based test suites on different Server-based Configurations



## Executing the same Client-based test suites on different Server-based Configurations



# Data-driven Configuration Tests: Consider Automated Testing



# Data-driven Configuration Tests: Consider Automated Testing

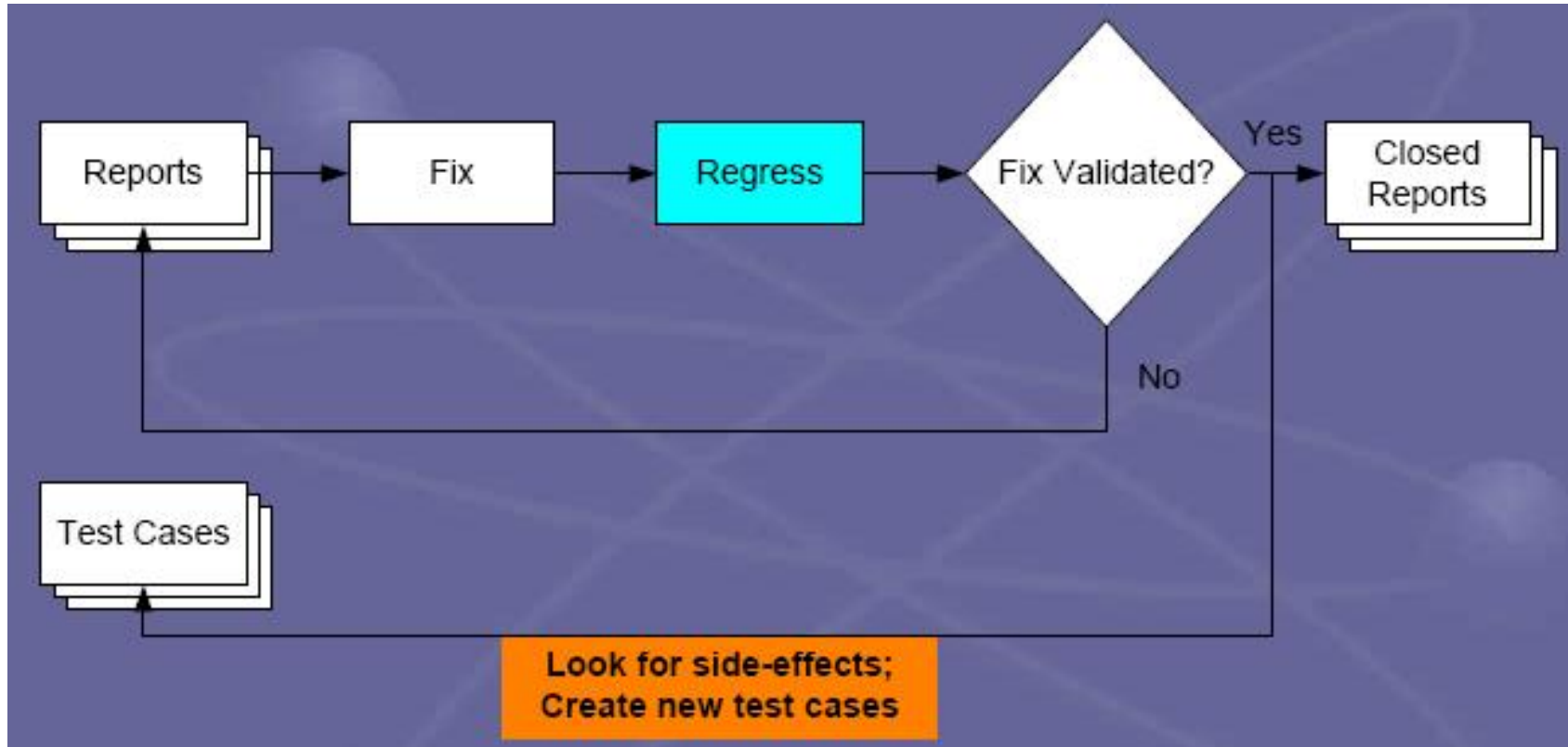
- **for (i=1; i<= iLastDataSet; i++)**
- **Open the dialog box.**
- **Set the control value in Group A, Group B and Group C to TRUE or 1 using the data in DataSet[i] for the control position in each group.**
- **Select OK.**
- **Verify results.**

Data Set	Group A	Group B	Group C
1	0	0	0
2	0	0	1
3	0	0	2
4	0	1	0
5	0	1	1
6	0	1	2
7	0	2	0
8	0	2	1
9	0	2	2
10	1	0	0
11	1	0	1
12	1	0	2
13	1	1	0
14	1	1	1
15	1	1	2
16	1	2	0
17	1	2	1
18	1	2	2
19	2	0	0
20	2	0	1
21	2	0	2
22	2	1	0
23	2	1	1
24	2	1	2
25	2	2	0
26	2	2	1
27	2	2	2

- **Release Acceptance Regression Tests -- See example**
- **Compatibility Tests -- See Example**
- **Configuration Tests -- See Example**
- **Release Acceptance Feature Tests**
- **Functional Acceptance Simple Tests**
- **Stress Tests**
- **Performance Tests**

- **Bug Fix Regression Tests -- See Example**
- **Task Oriented Functional Tests**
- **Exploratory Tests**
- **Real World User-Level Tests**

# Bug-Fix Regression: Manual Testing is More Effective



- **Bug Fix Regression Tests -- See Example**
- **Task Oriented Functional Tests**
- **Exploratory Tests**
- **Real World User-Level Tests**

- **Silk Organizer**
- **Components Characteristics**
- **Test Frame Architecture**
- **Configure Recovery System**
- **Define Common Objects**
- **Test Case Architecture**

- **Powerful front-end for SilkTest**
- **Use Testplan Templates**
- **Pass arguments to reusable tests**
- **Pass global variables**
- **Change test frame in mid plan – optionset command**
- **Track Automation & Pass/Fail progress**
- **Select tests quickly and semi-powerfully**
- **Write .pln files from external application**
- **Use relative paths to test scripts from plans**

- **Planned - architected**
- **Maintainable - able to withstand changes in GUI**
- **Useable - understandable by team**
- **Flexible - useful in multiple situations**
- **Modular - easily modified**
- **Documented - VERY IMPORTANT**
- **Code Review - Find problems, learn**

- **Communicate with Users**
  - + **Understand who will use the frame**
- **Configure Recovery System**
  - + **No automation without recovery**
- **Define common objects**
  - + **Approach the frame as a developer**
- **Create Testcases**
  - + **Architect testcases for maintainability and reuse**

- **BaseState () - easily customize DefaultBaseState**
- **TestCaseEnter () - control behavior before testcase**
- **TestCaseExit () - control behavior after testcase**
- **DefaultBaseState () - return product to known state**
- **Compiler Constants - allows flexibility in test frame**
- **Read values from file or registry - make frame more robust**
- **GetTestCaseState () - make decisions based on whether testcase is starting or ending**

- **Define windows/tags - use good names**
- **Define functions - useable from several locations**
- **Define classes/methods/properties - can be added to Library Browser source**
- **Define records - powerful**
- **Define enumerated types - huge time-saver**
- **Define global variables - useful for multiple scripts**
- **Handle custom objects - can be a show-stopper**
- **Define application states - easily drive to known starting states**

- **Define Reusable Testcases**
- **Define Test Data**
- **Use Agents for remote configuration testing**
- **Use Scenario Testing (spawn/rendezvous) for multiple client testing**
- **Use randomness with documentation to find defects**
- **Create powerful Application States**
- **Define functions/methods as needed**

- **Good test planning leads to successful automation.**
- **Decide what to automate and then automate those specific tests.**
- **Don't forget to find the bugs.**
- **Examples of Test plan Templates and Reusable Testcases can be found at [www.automationexpertise.com](http://www.automationexpertise.com)**

**John Green is the founder of Automation Expertise, an automated testing consulting firm. He has over 15 years of experience in software testing, including seven years of automation experience. He was formerly a Sr. Consultant with Segue Software and provided consulting and training services to Segue's clients across the U.S. and Europe. He has served on the Research Committee of the Quality Assurance Institute and has served on the Board of Directors of the Kansas City QA Association. He has been a frequent speaker on automation for many organizations including the Bay Area Segue User's Group, Adobe, Segue's Users Conference and Hyperion Software. John currently works with several companies providing expertise in software test automation.**

# About Hung Q. Nguyen

Hung Q. Nguyen, B.Sc.Q.A., is Founder, President, and CEO of *LogiGear Corporation*, a Silicon Valley company whose mission is to help software development organizations deliver the highest quality products possible while juggling limited resources and schedule constraints by providing testing expertise and resources. Its partners benefit from its seasoned testing staff and facilities, practical training programs, and test support products. Hung has held management positions in engineering, quality-assurance, testing, product development, and information technology. He has also made significant contributions as a tester and programmer. Hung is coauthor of the best-selling book *Testing Computer Software* (ITP/Wiley). He is an ASQ-Certified Quality Engineer and a senior member and San Francisco Section Certification Chairman of the American Society for Quality.



# THE FIRST CONFERENCE ON E-BUSINESS RELIABILITY

## Test Planning for E-Business Applications – Improving ROI

**THE END**