# LabVIEW Developer Days

## Build Code. Form Communities. Gain Confidence.

NATIONAL INSTRUMENTS™

# Software Engineering for LabVIEW



NATIONAL INSTRUMENTS

High-Volume Production Test

Structural Health Monitoring

Medical Devices

Space Exploration

Large Physics Applications

Avionics Applications


# Large System Development Powered by LabVIEW



NATIONAL INSTRUMENTS

# Lawrence Livermore National Labs

## Developed automated maintenance process for world's most energetic laser array at the National Ignition Facility using NI LabVIEW and PXI

- LabVIEW increased productivity by 3X over Java and C++
- Developed complex application consisting of over 1,000 VIs
- Applied software engineering practices to ensure quality

*An overhead view of one of the main laser chambers*

> *"The value in using the graphical dataflow language is the speed in which a team can deliver a robust solution while still using proper software engineering practices.*
>
> *- Glenn Larkin, LLNL*



NATIONAL INSTRUMENTS

# Business Impact Case Study:
# Philips Home Healthcare Solutions

Accelerated Product Development | Full Regulatory Quality Compliance | Productive Test Platform Team

By implementing software engineering best practices and adopting the NI platform, the team increased its productivity by **347 percent** and reduced the company's cost of quality, saving it **$4.5 million USD** annually.

## Philips HHS Accelerates Product Development and Decreases Test Costs by **81** Percent Using the NI Platform



PHILIPS

# Goal is to Detect Defects Early and Manage the Risks Associated with Change

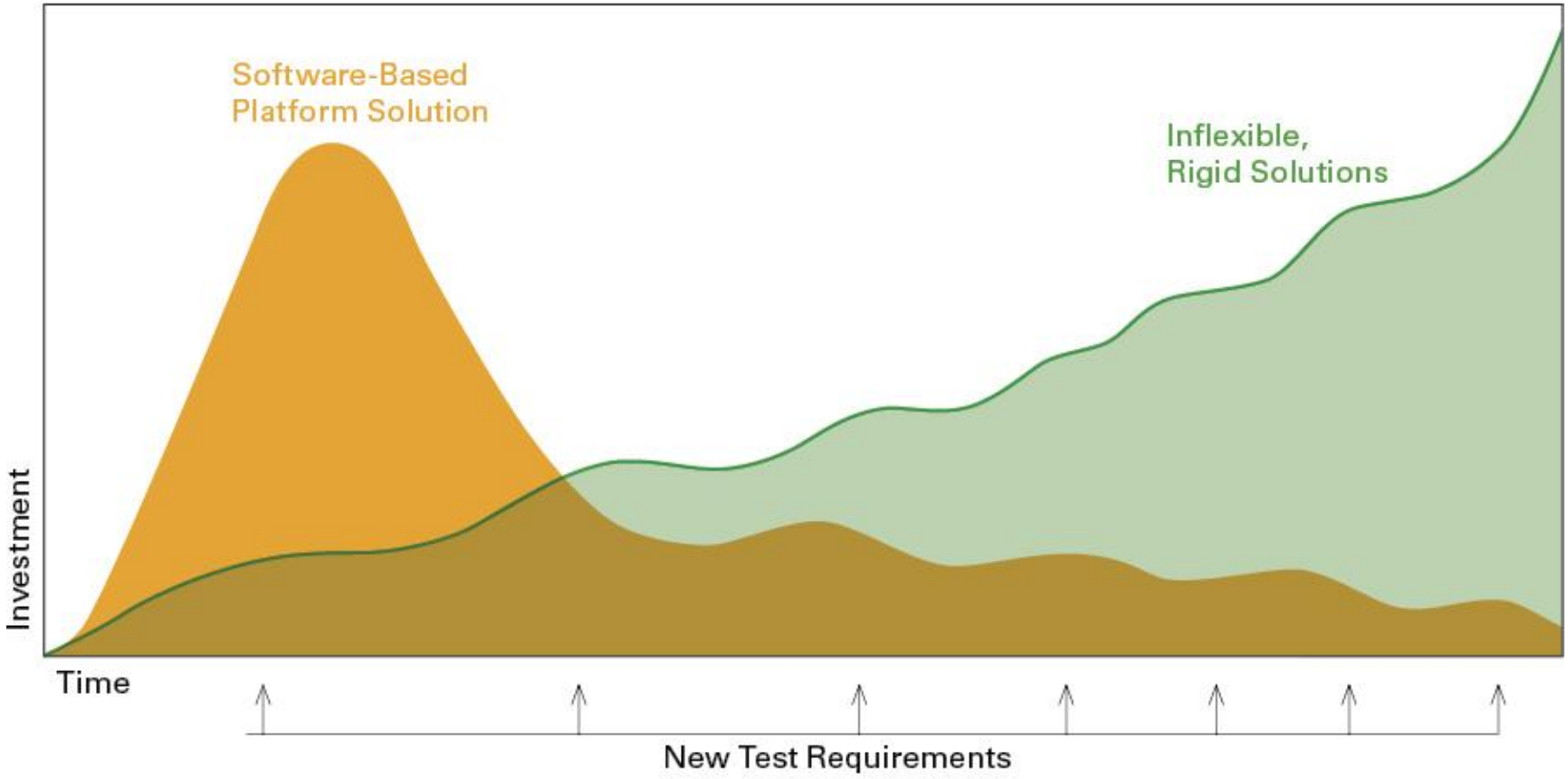
| Development Phase | Cost Ratio |
|---|---|
| Requirements | 1 |
| Design | 3-6x |
| Implementation | 10x |
| Development Testing | 15-40x |
| Acceptance Testing | 30-70x |
| Post Release | 40-1000x |

Based on an analysis of 63 software development projects at companies including IBM, GTE and TRW



NATIONAL INSTRUMENTS

# Investing In People and Processes






NATIONAL INSTRUMENTS

# Examples of Software Engineering Debt

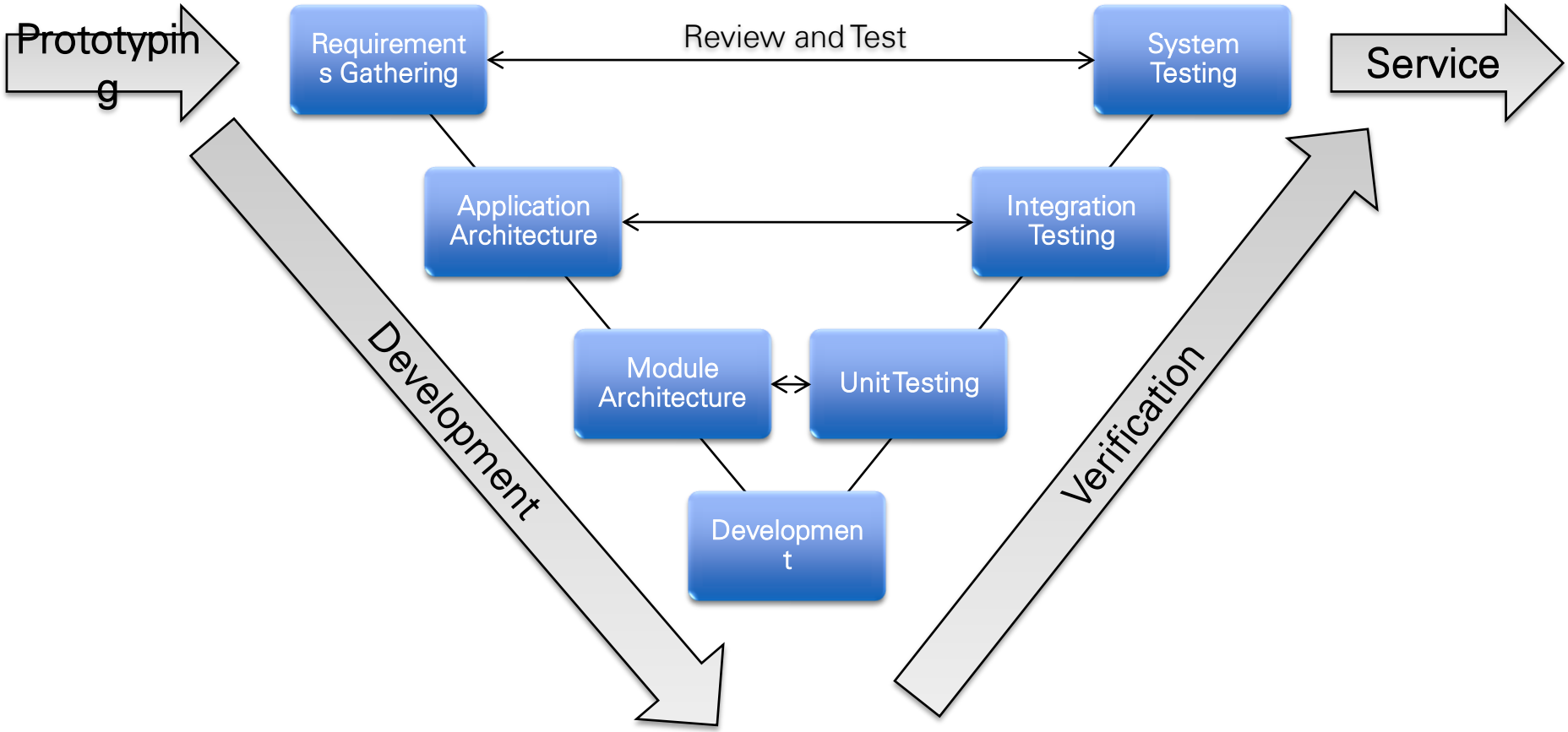
(just *some* of the most common LabVIEW development mistakes)

- ✓ No source code control (or Project)
- ✓ Flat file hierarchy
- ✓ 'Stop' isn't tested regularly
- ✓ Wait until the 'end' of a project to build an application
- ✓ Few specifications / documentation / requirements
- ✓ No 'buddying' or code reviews
- ✓ Poor planning (Lack of consideration for SMoRES)
- ✓ No test plans
- ✓ Poor error handling
- ✓ No consistent style
- ✓ Tight coupling, poor cohesion



NATIONAL INSTRUMENTS

# Software Engineering V-Model



# Agenda

Configuration Management and Source Code Control

Creating Reuse Libraries with VIPM

Requirements Tracking and Management

Static Code Analysis and Code Reviews

Dynamic Code Analysis

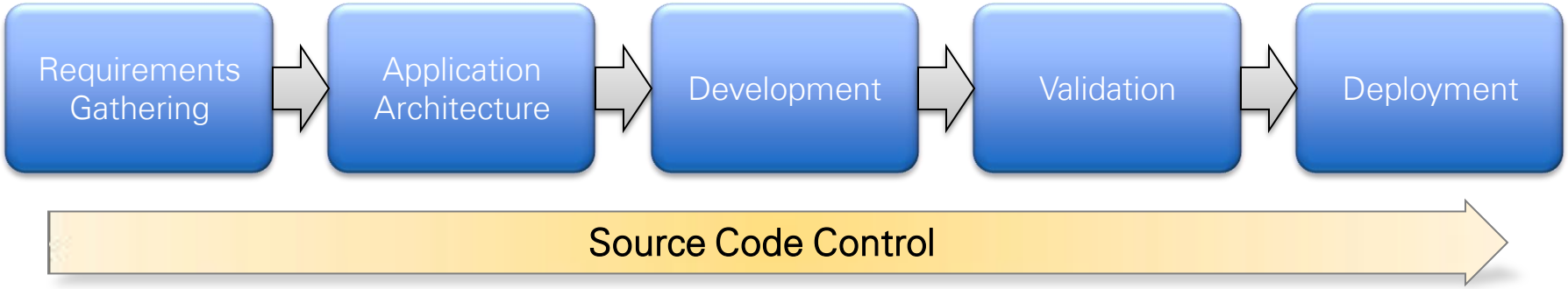
Regression Testing with the Unit Test Framework

# Configuration Management and Source Code Control



NATIONAL INSTRUMENTS

# Software Configuration Management Tools




- ✓ Provides remote, backed-up repository of code
- ✓ Manage multiple versions and branches
- ✓ Track and identify changes and their impact
- ✓ Communicate and share code across a team
- ✓ Reduces pain throughout process

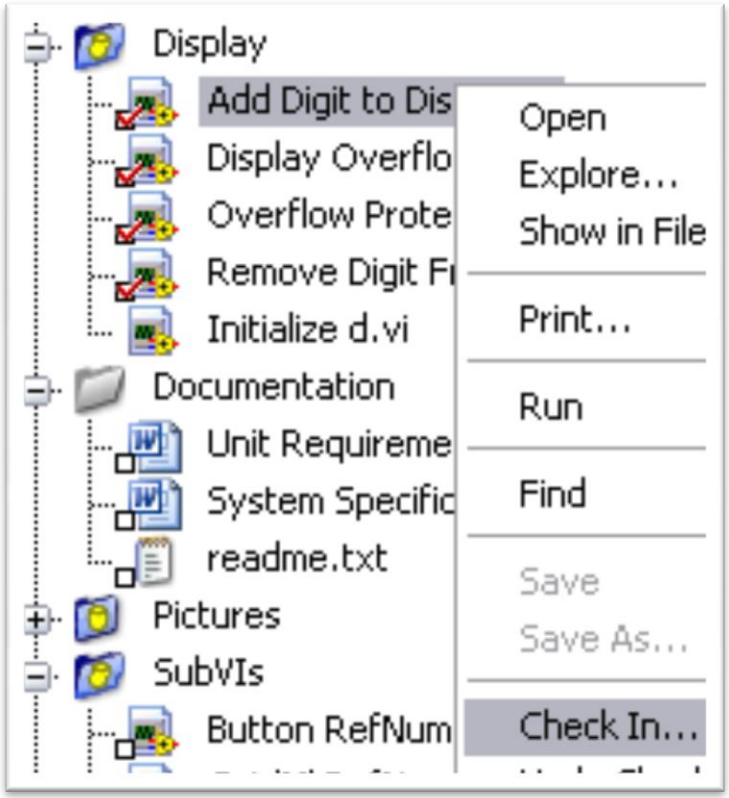


NATIONAL INSTRUMENTS

# A Wide Variety of SCC Tools Are Available

Any of them can be used together with LabVIEW

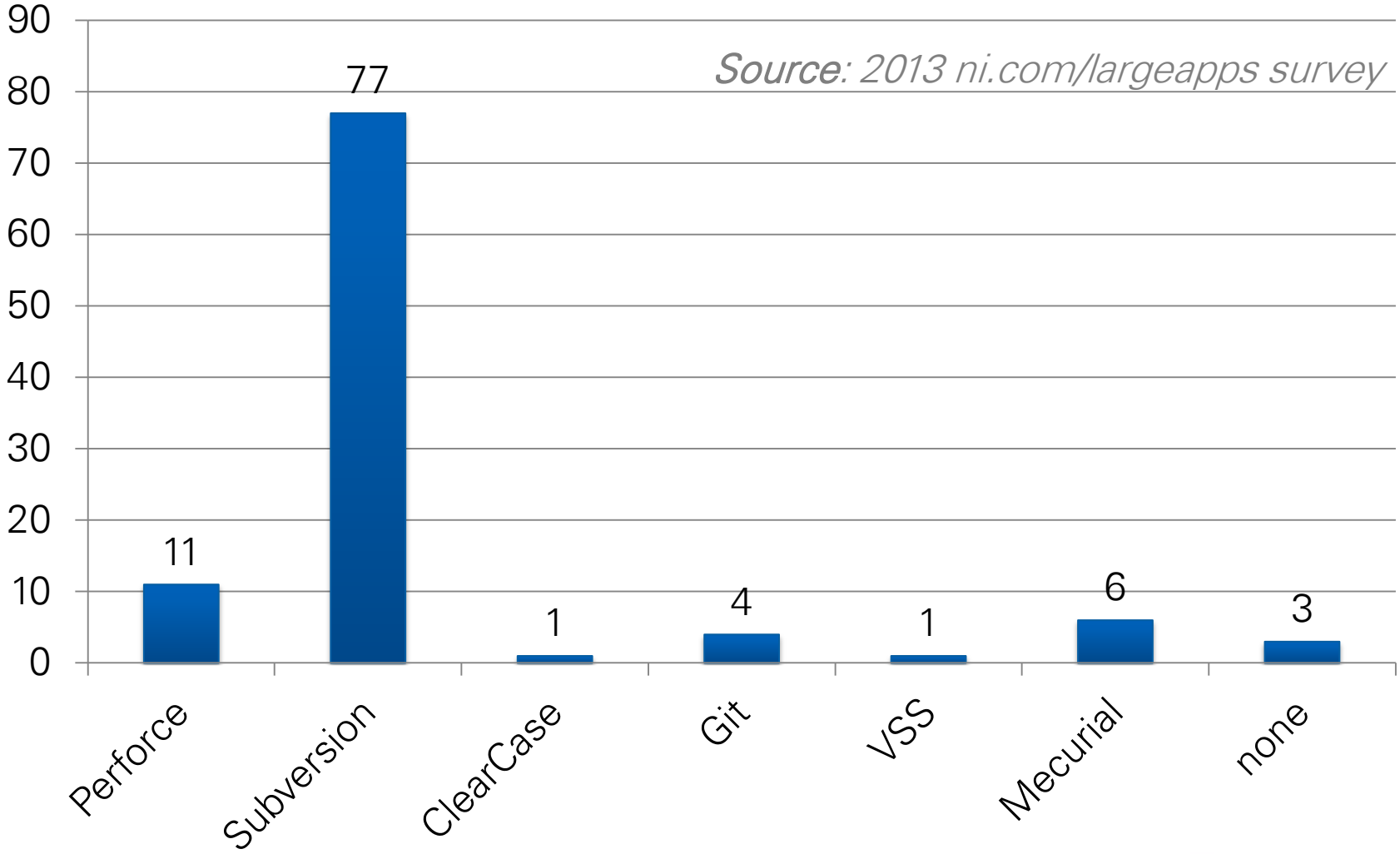
- Perforce
- Subversion
- Git
- Mecurial
- Microsoft Visual SourceSafe
- Microsoft Team System
- Rational ClearCase
- PCVS (Serena) Version Manager
- MKS Source Integrity
- Seapine Surround SCM
- Borland StarTeam
- Telelogic Synergy
- ionForge Evolution






NATIONAL INSTRUMENTS

# Popularity of SCC Options Amongst LabVIEW Programmers






NATIONAL INSTRUMENTS

# Demonstration

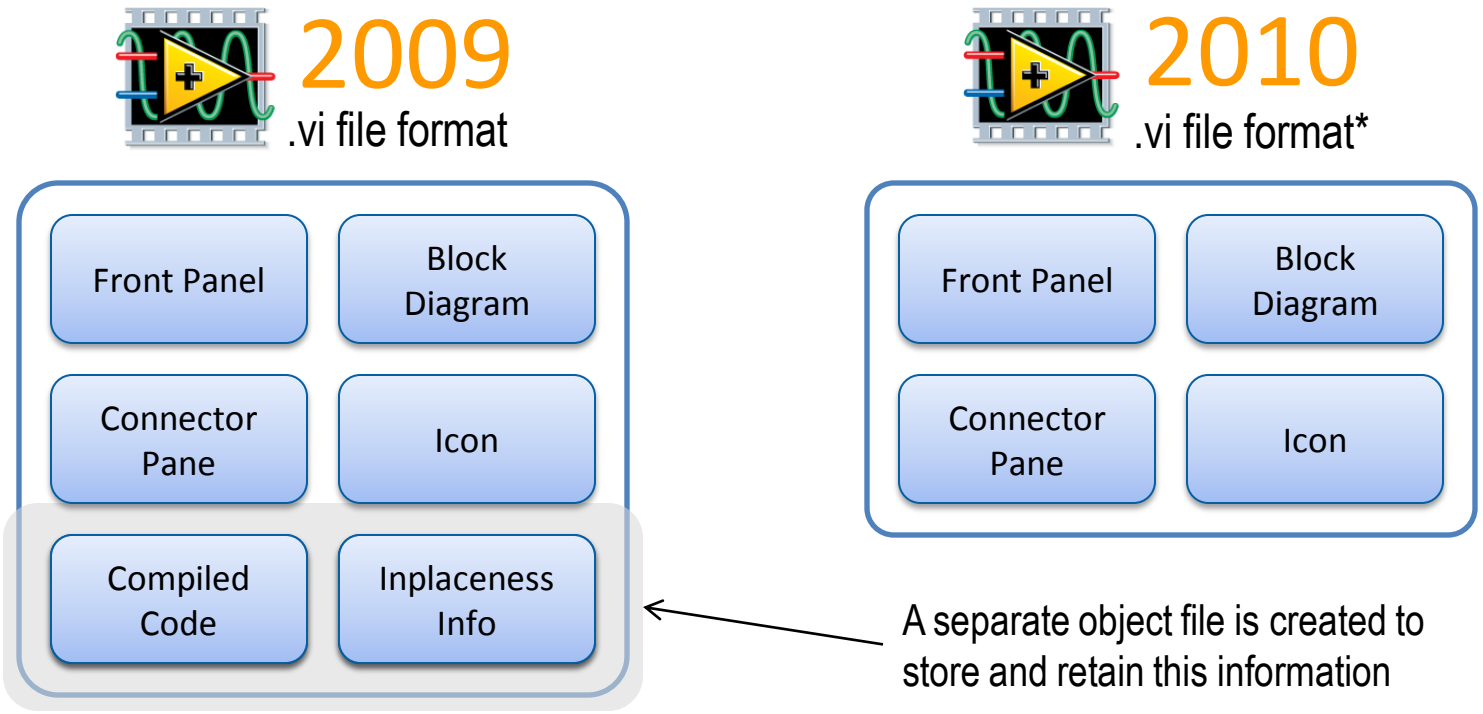
Using Source code Control with LabVIEW



NATIONAL INSTRUMENTS

# Separate Compiled Code From Source File

## Improved Source Code Control Integration



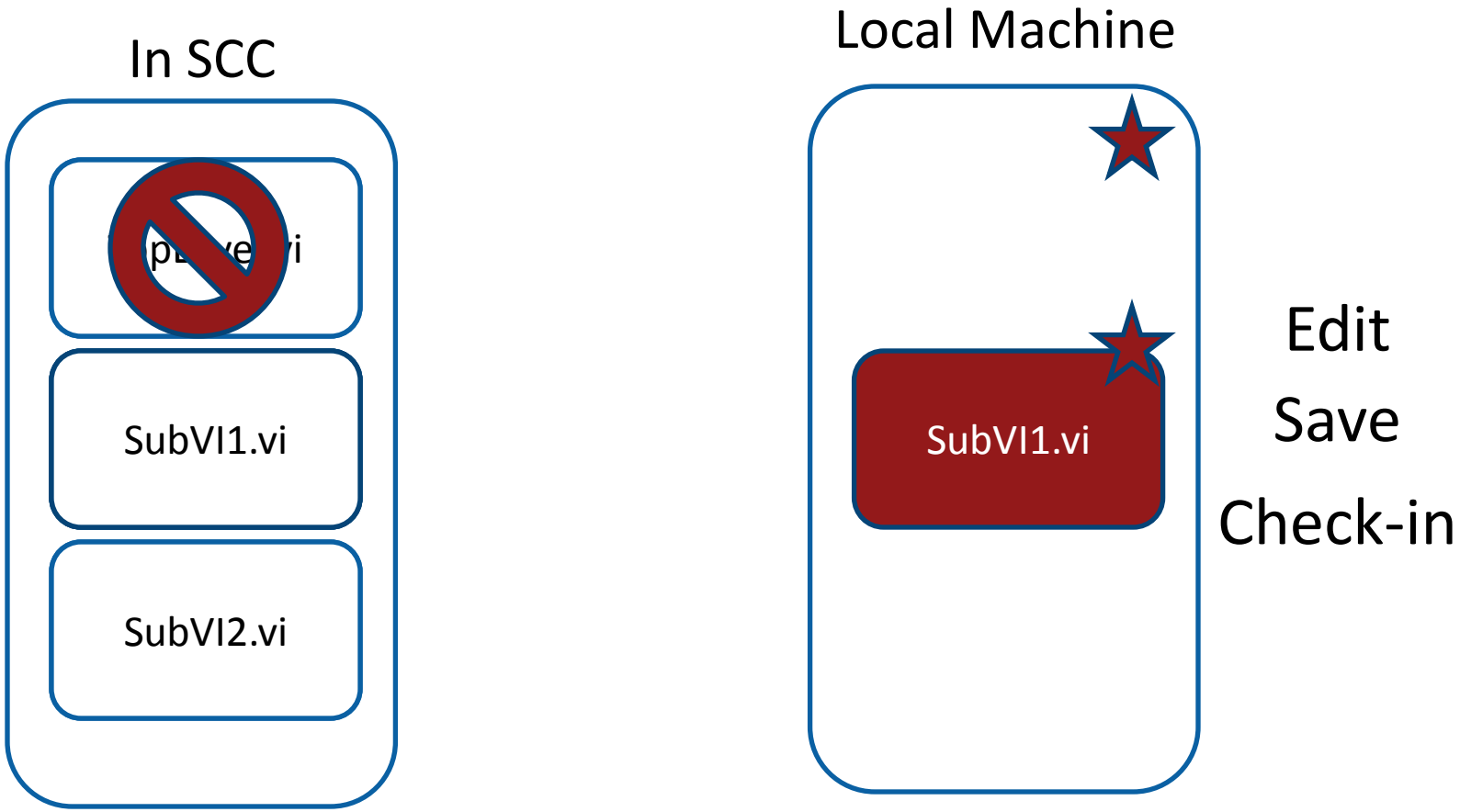

Eliminate the need to re-save and re-submit files to source code control unless the graphical source code has been changed by the developer

**this feature is not on by default and needs to be enabled from the VI Properties dialog*

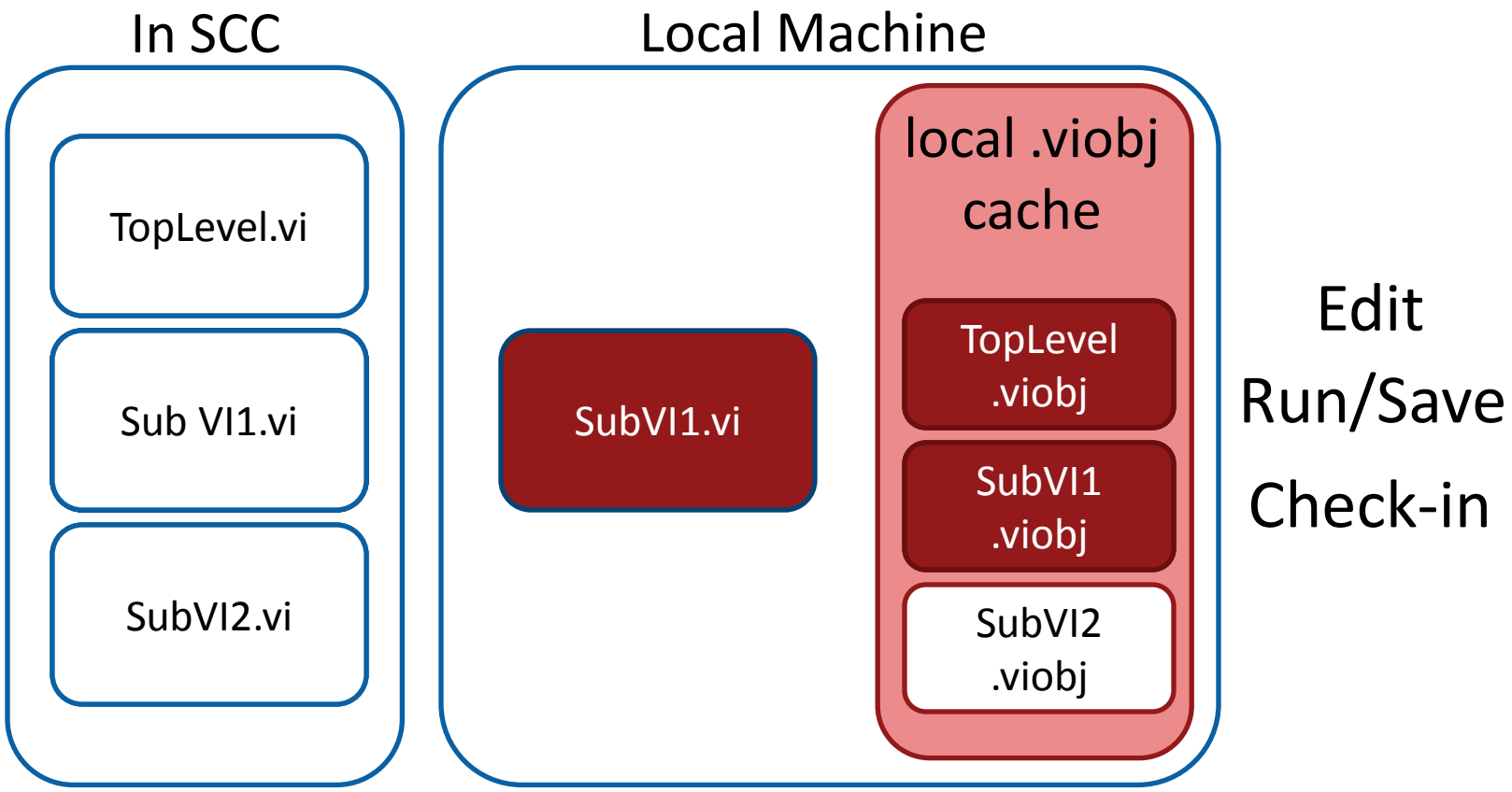


NATIONAL INSTRUMENTS

# Source Code Control Scenario: Today



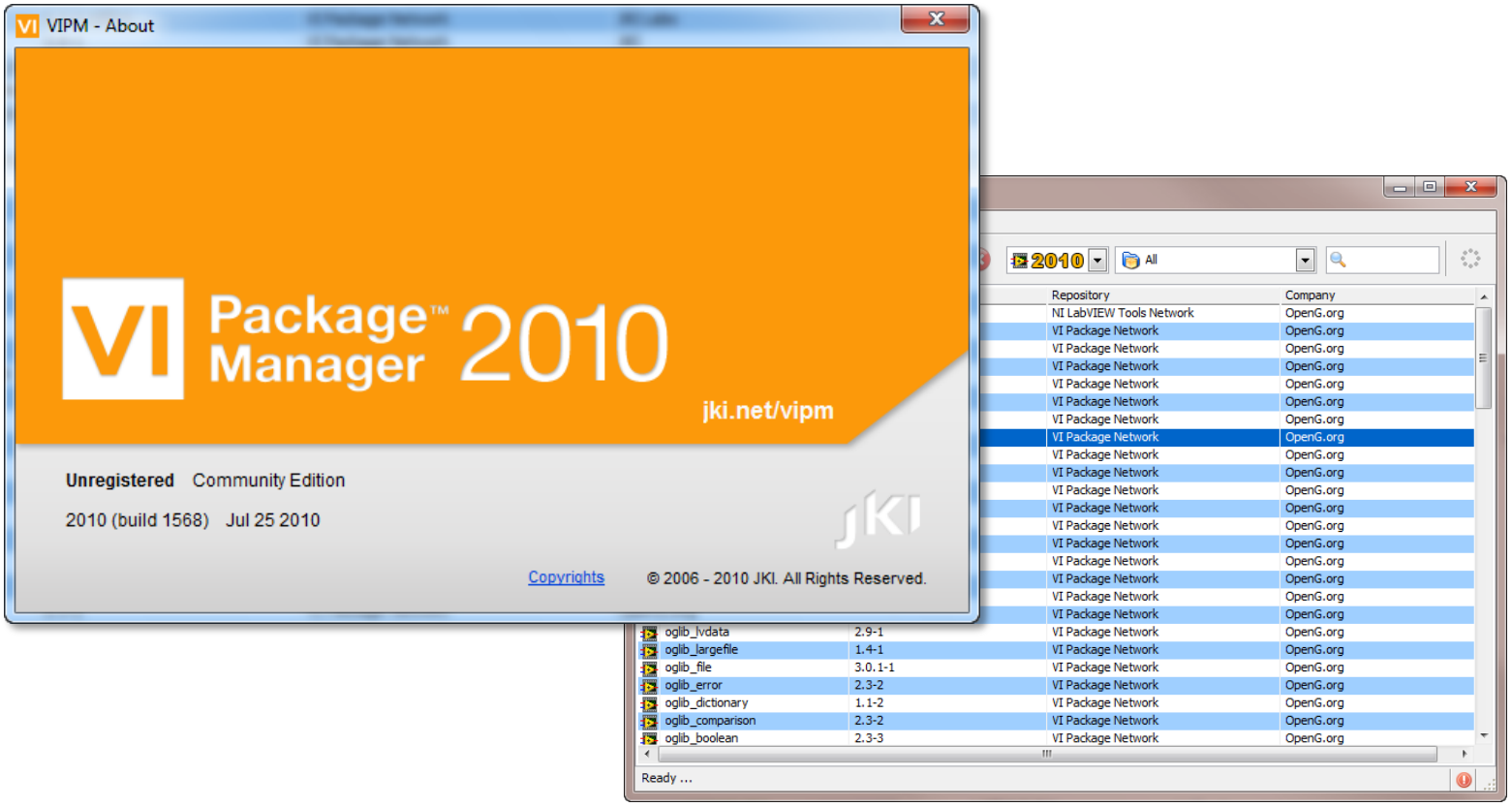
# Source Code Control Scenario: 2010






NATIONAL INSTRUMENTS

# VI Package Manager

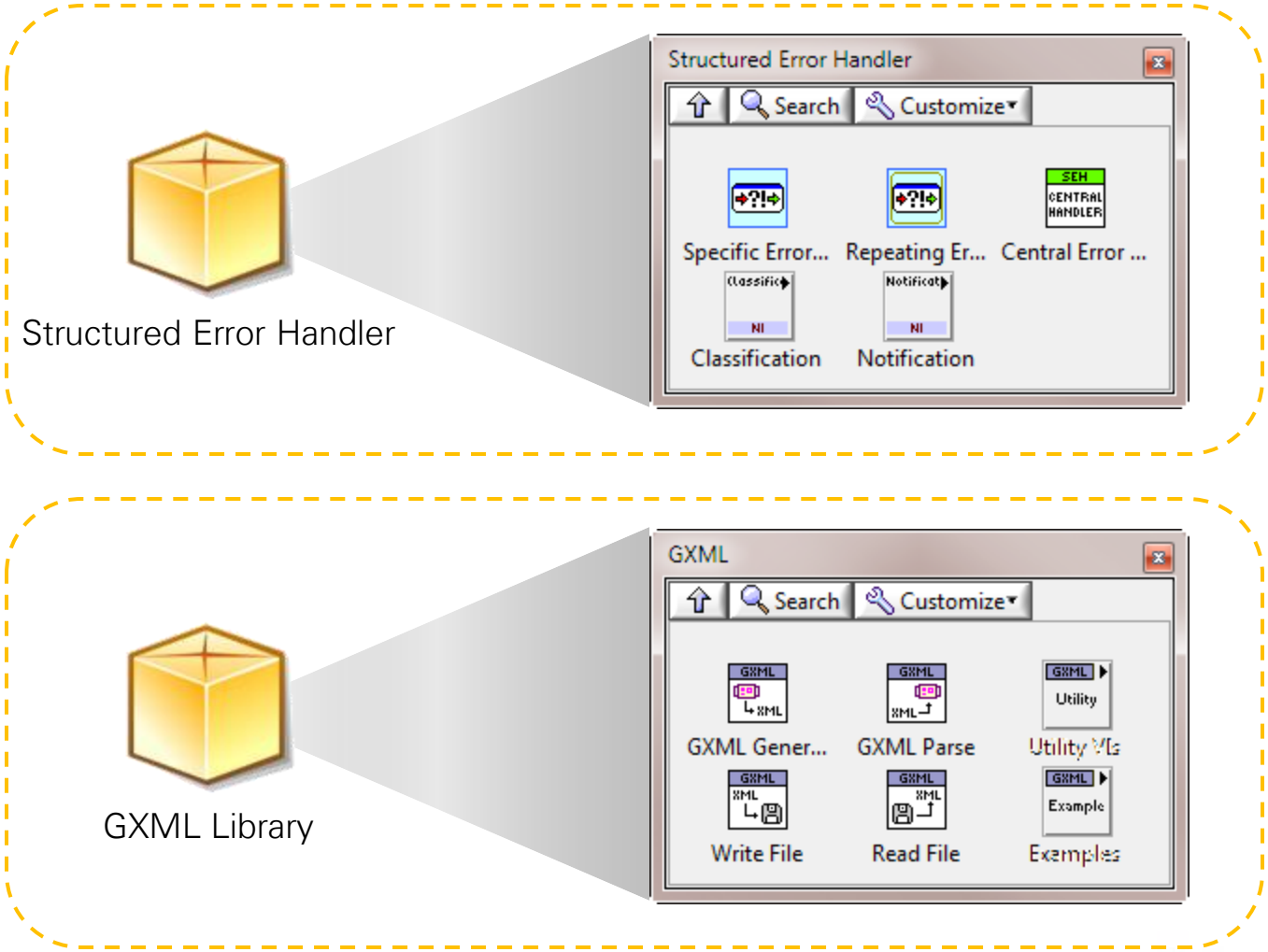



Build and manage packages of LabVIEW code



NATIONAL INSTRUMENTS

# Install and Manage VI Packages

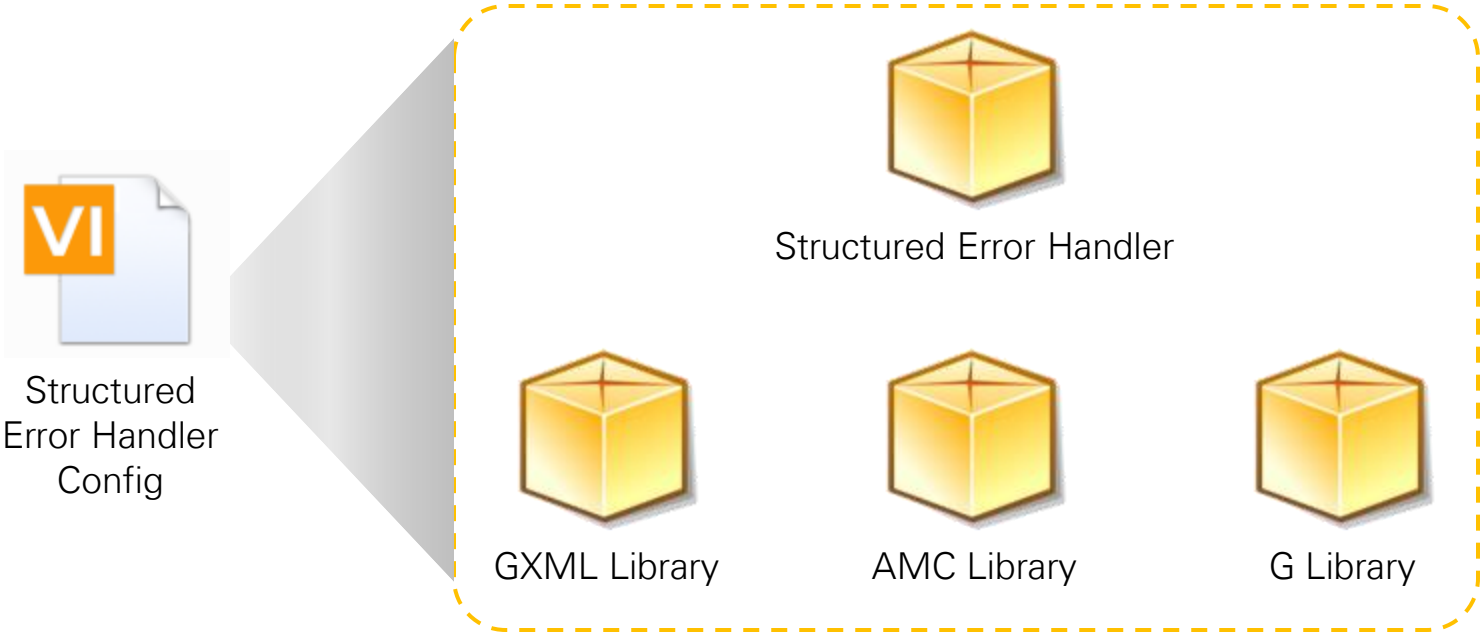




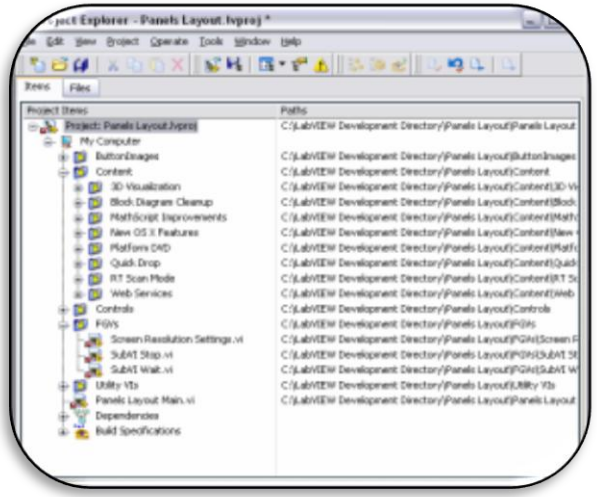

NATIONAL INSTRUMENTS

# Create VI Configuration Files

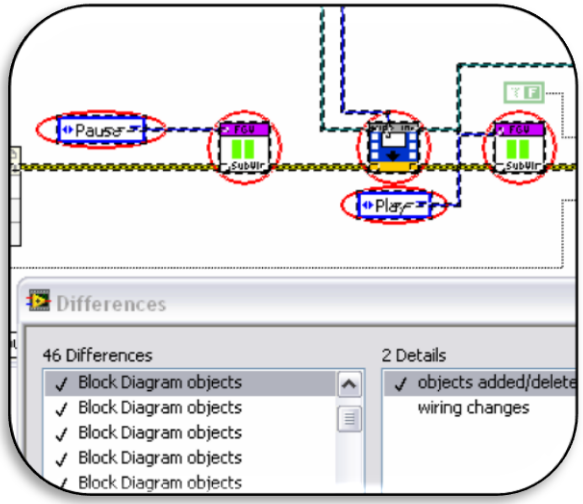
A single file that contains multiple packages.
Easily share and distribute code that depends upon multiple libraries.



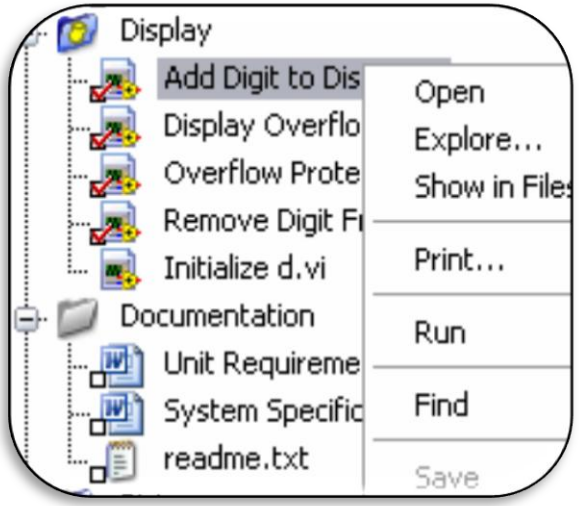
# Software Configuration Management for LabVIEW



***System Level View***

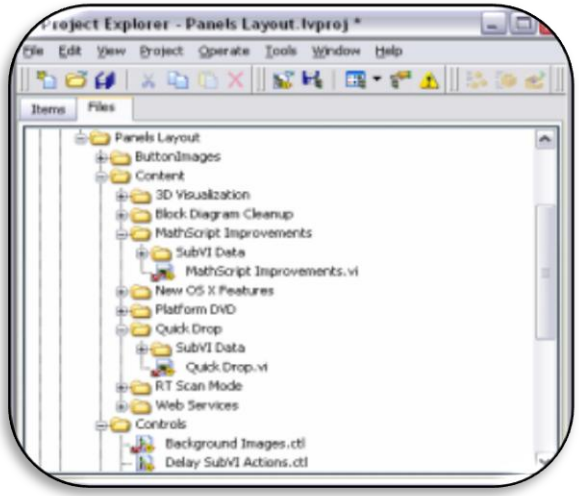



***Track Changes***

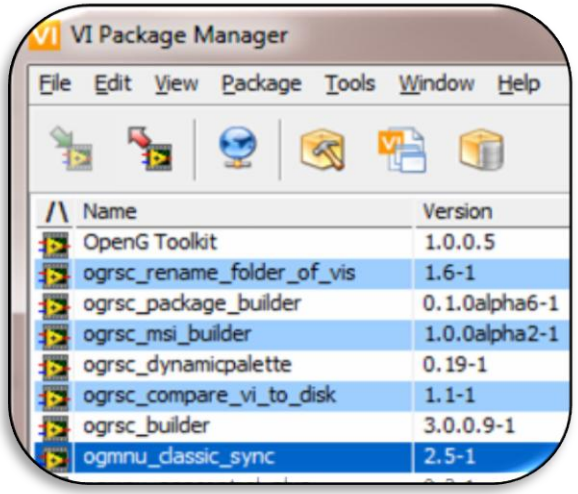



***Integrate with SCC***

***Merge Graphical Code***




***Manage Files and Links***



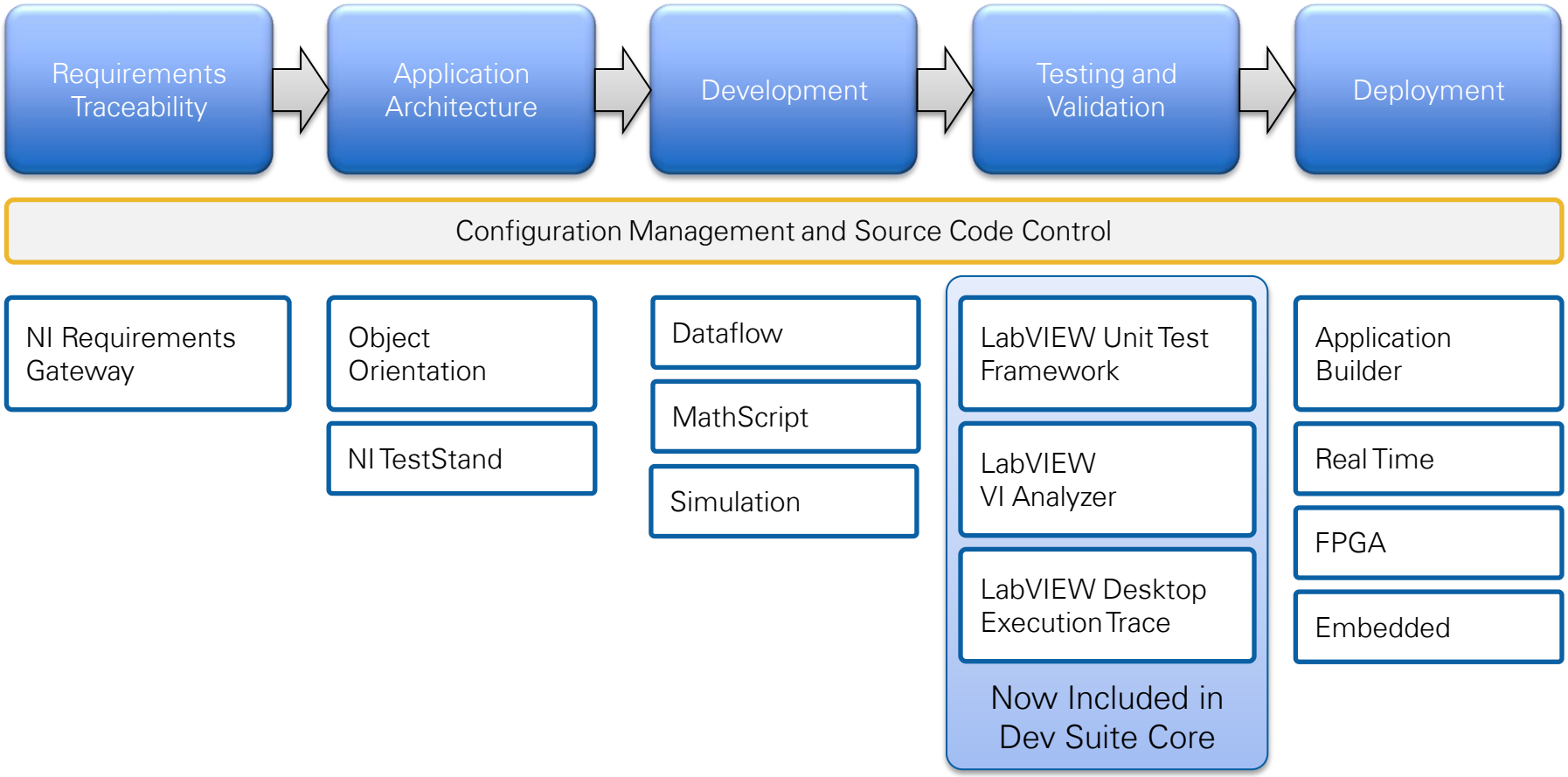

***Manage Reuse Libraries***

# Software Engineering Tools



NATIONAL INSTRUMENTS

# Software Engineering Tools for LabVIEW

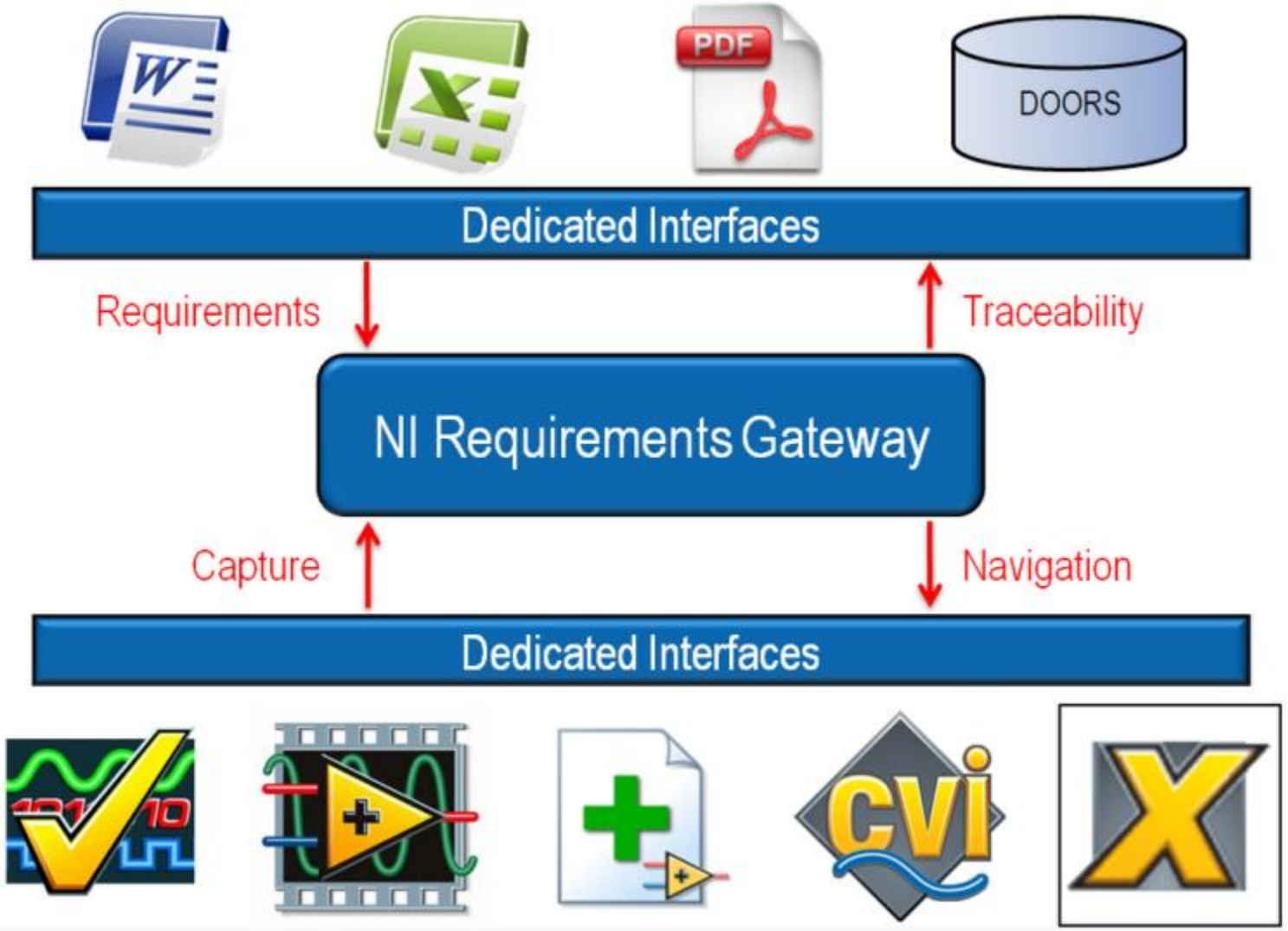



These tools improve and automate life-cycle management, code quality analysis and developer productivity in order to deliver high-quality systems in less time.



NATIONAL INSTRUMENTS

# Requirements Traceability Solution from NI






NATIONAL INSTRUMENTS

# Demonstration

Requirements Management with NI Requirements Gateway



NATIONAL INSTRUMENTS

# Establish or Adopt Development Guidelines

## Front Panel Style

- Fonts and Text Characteristics
- Colors
- Graphics and Custom Controls
- Layout
- Sizing and Positioning
- Labels
- Paths versus Strings
- Enumerated Type Controls versus Ring Controls
- Default Values and Ranges
- Property Nodes
- Key Navigation
- Dialog Boxes

## Style Checklist

- VI Checklist
- Front Panel Checklist
- Block Diagram Checklist

## Block Diagram Style

- Wiring Techniques
- Memory and Speed Optimization
- Sizing and Positioning
- Left-to-Right Layouts
- Block Diagram Comments
- Call Library Function Nodes and Code Interface Nodes
- Type Definitions
- Sequence Structures

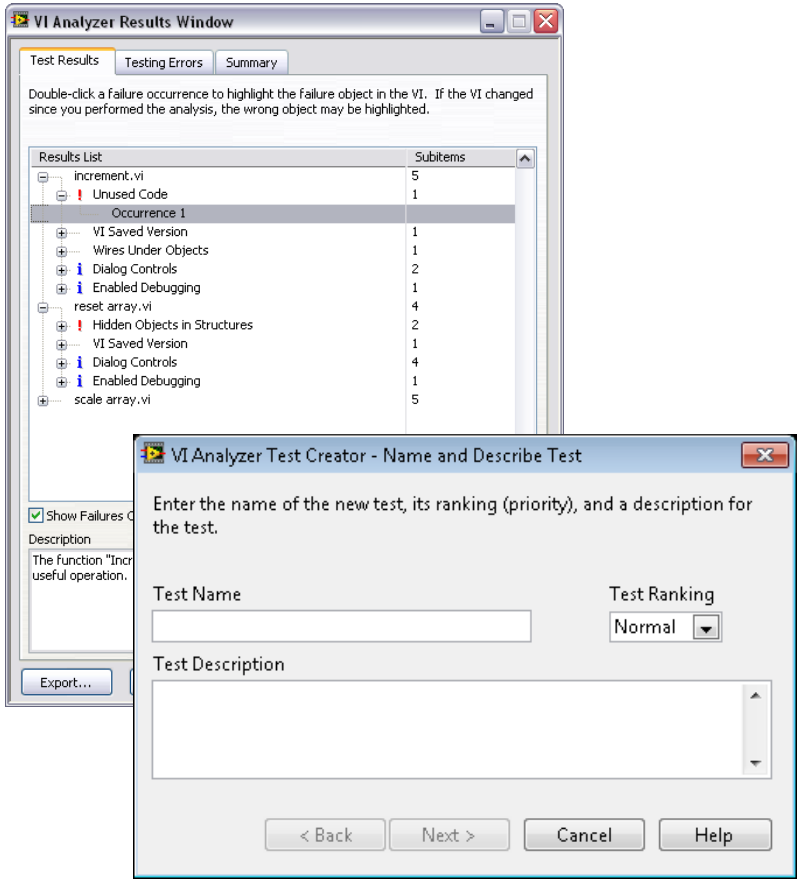
## Icon and Connector Pane Style

- Icons
- Example of Intuitive Icons
- Connector Panes

**NI Style Guideline:** ni.com/largeapps

# Preparing for a Code Review with VI Analyzer

- Automate code analysis with 80+ configurable tests
  - Performance
  - Style
  - Complexity
- Interactively inspect failures
- Generate custom reports
- Code complexity metrics
- Write your own tests with VI Scripting *LabVIEW 2010*






NATIONAL INSTRUMENTS

# Demonstration

## Static Analysis with VI Analyzer

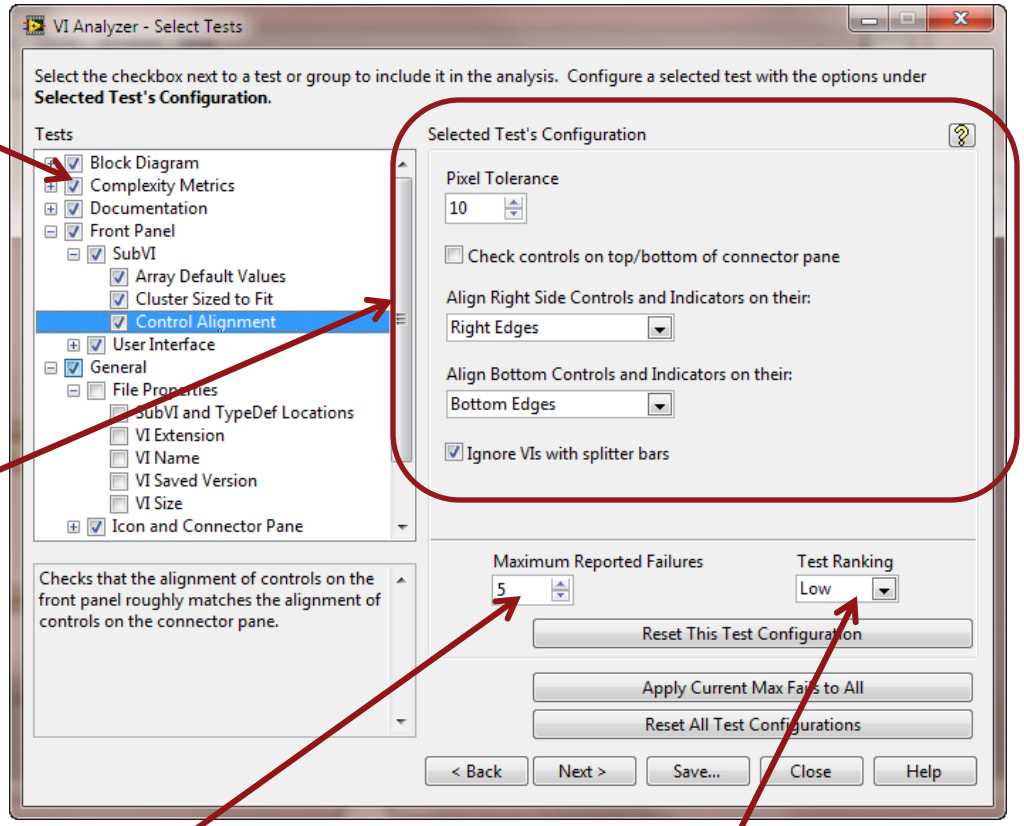


NATIONAL INSTRUMENTS

# Creating Custom Tests

Select tests to run

Test specific configuration settings



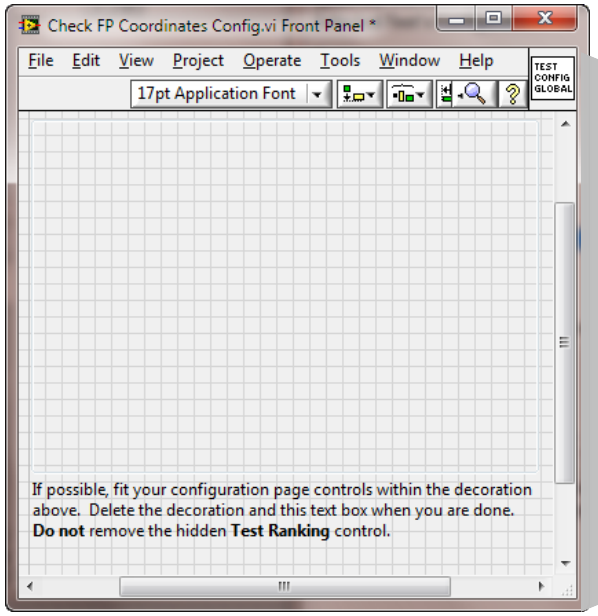

Number of failures to report

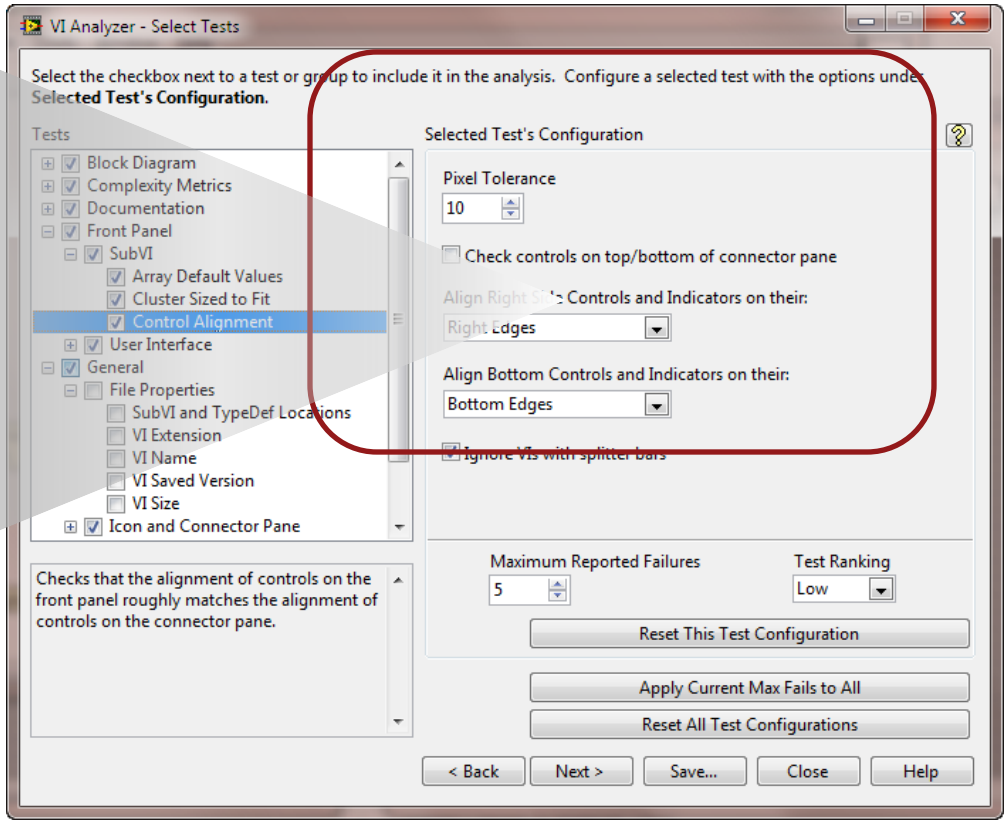
Test Priority



NATIONAL INSTRUMENTS

# Define Configuration Options




Access values in test code using Global variables

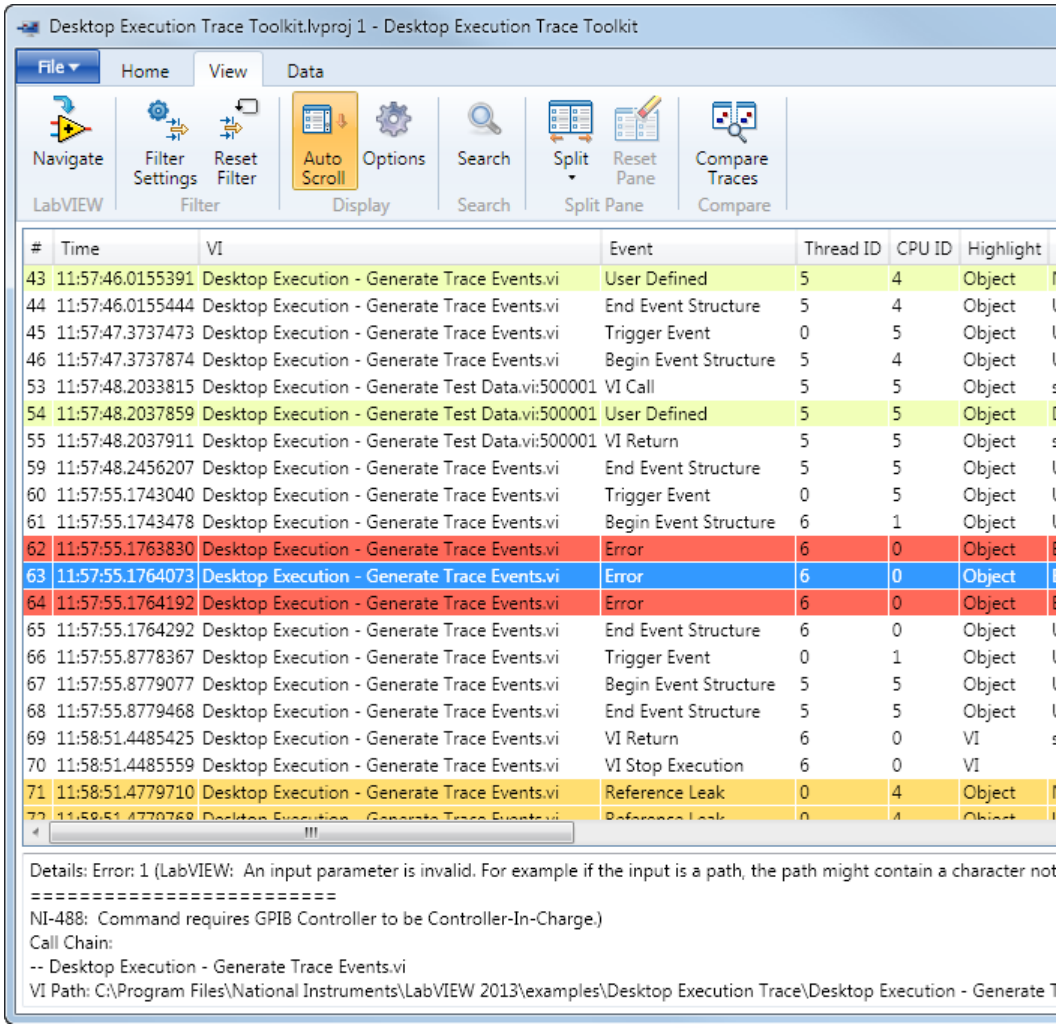


NATIONAL INSTRUMENTS

# LabVIEW 2013 Desktop Execution Trace Toolkit

## New Desktop Execution Trace Toolkit

- Reinvented user interface based on user feedback
- Capable of handling much larger traces
- Improved filtering and sorting options
- Comparison tool for diff'ing trace data



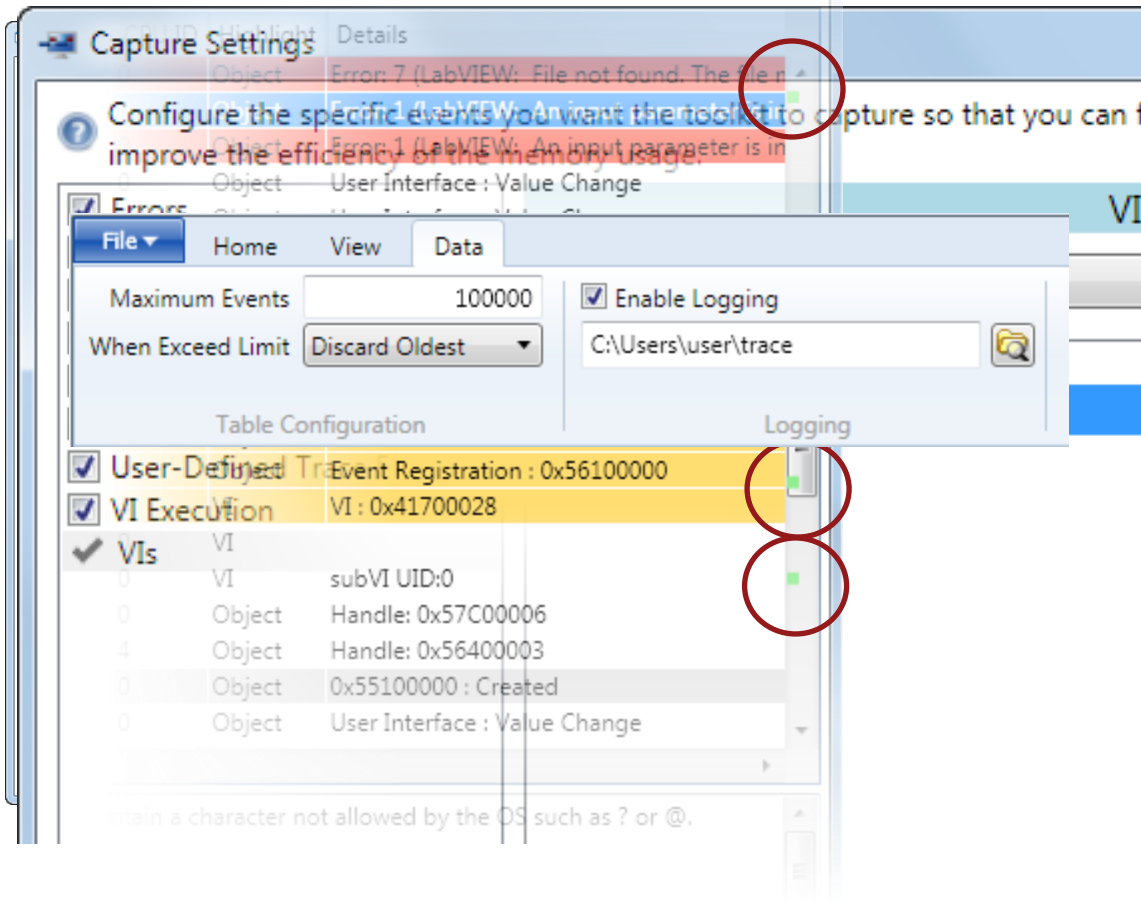



NATIONAL INSTRUMENTS

# LabVIEW 2013 Desktop Execution Trace Toolkit

## Feature Highlights

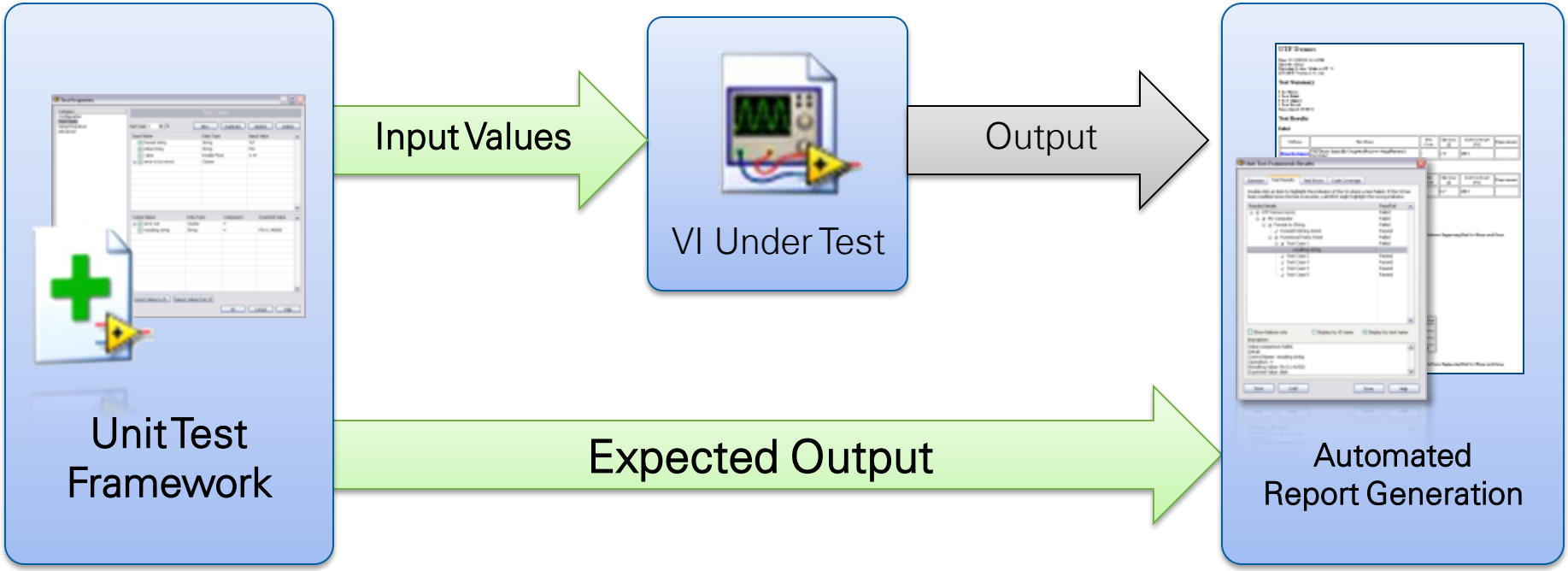
- Compare different sessions to examine behavioral changes
- User-requested trace configuration options
- Bookmarks make navigating multiple traces simple
- Automatic logging allows extended trace sessions






NATIONAL INSTRUMENTS

# LabVIEW Unit Test Framework

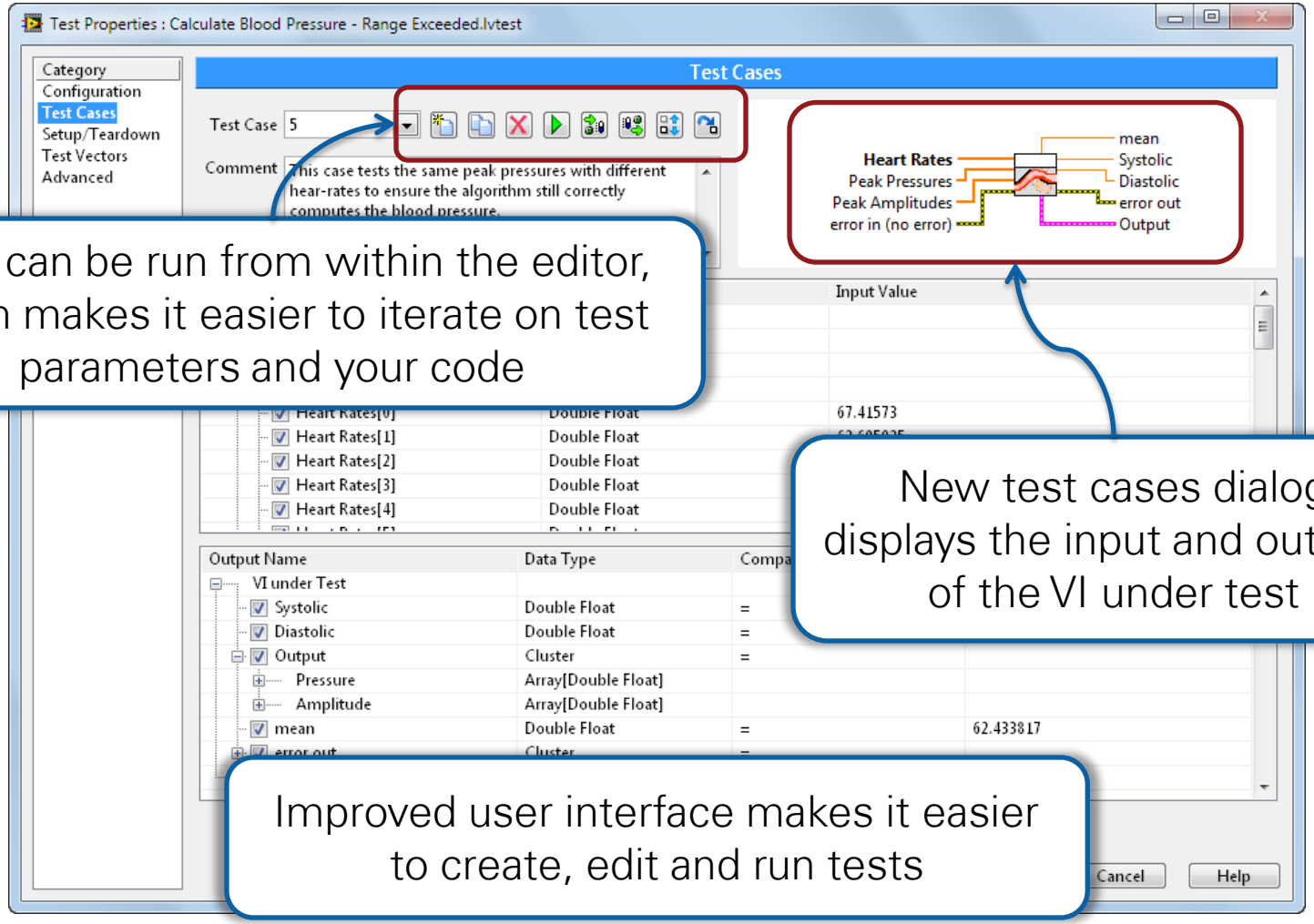



Test vector = Input value(s) + Expected output(s)



NATIONAL INSTRUMENTS

# LabVIEW 2013 Unit Test Framework




Tests can be run from within the editor, which makes it easier to iterate on test parameters and your code

New test cases dialog displays the input and outputs of the VI under test

Improved user interface makes it easier to create, edit and run tests

# Demonstration

Regression Testing with the Unit Test Framework



NATIONAL INSTRUMENTS

# Keep in Touch

Join the Advanced Development Community:

ni.com/community/largeapps

Download this code and slides from

bit.ly/lv_swe



NATIONAL INSTRUMENTS