REFACTORING LabVIEW CODE

Terry Stratoudakis, PE
Certified LabVIEW Developer
Certified Professional Instructor

ALE System Integration
Melville, New York
December11, 2008

Overview

I. What is Refactoring?

II. Causes of ‘Bad Code’

III. When to Refactor Code

IV. Refactoring Guidelines

V. Block Diagram Cleanup

VI. VI Analyzer Toolkit

VII. References

Anyone can write code that a computer can understand.

Good programmers write code that humans can understand.

Refactoring: Improving the Design of Existing Code

What is Refactoring?

Code Changes

- Increase readability and maintainability
- Observable behavior remains the same

Has been around for years

not LabVIEW specific

A hard sell to Management

Also referred to as “cleaning up code”

Refactoring is not code optimization

Causes of ‘Bad Code’

I. Novice programmer
II. Rushed development
III. Prototype became final application
IV. Experimenting of new algorithms or design patterns

When to Refactor Code

I. Adding a feature to a VI

II. Debugging a VI

III. There is value in a VI that functions

IV. Plans to make VI part of reuse library

When to Rewrite VIs

I. VIs do not function

II. VIs satisfy small portion of your needs

III. VI needs re-architecting

    I. Can use sub-VIs

Refactoring Guidelines

I. Review code and understand it well

II. Create Test Plan

III. Keep backups or use source code control

IV. Keep changes simple

V. Test often

Review & Understand Code

I. Review documentation

II. Meet with original developers

III. Meet with operators

IV. Review code

V. Make your own notes, flow charts

VI. Run code

Create Test Plan

I. Refactoring emphasizes testing

II. Ensures that other parts do not “break”

III. Could use test driven development

    a. Make VIs that ‘test’ the refactored VIs

Keep Backups

I. Allows you to undo changes

II. Source Code Control is preferred

Examples: SourceSafe, CVS, SVN, Perforce

III. Can zip source folder and store

Keep Changes Simple

I. Make cosmetic improvements first

II. Enter notes directly into VIs

III. Allows you to get more familiar with code

IV. Deep changes may break code

Test Often

I. Minimal time loss when undoing changes

II. Changes may expose existing race conditions

III. Have operator run program

Block Diagram Clean-up (LabVIEW 8.6 only)

I. Very fast clean up of VI

II. Better for lower-level VIs

III. May obfuscate VIs with Design Pattern

IV. Can tweak parameters

Tools>>Options Block Diagram: Cleanup

V. Cannot cleanup partial Block Diagram

VI. Can undo

Block Diagram Clean-up (LabVIEW 8.6 only)

I. Very fast clean up of VI

II. Better for lower-level VIs

III. May obfuscate VIs with Design Pattern

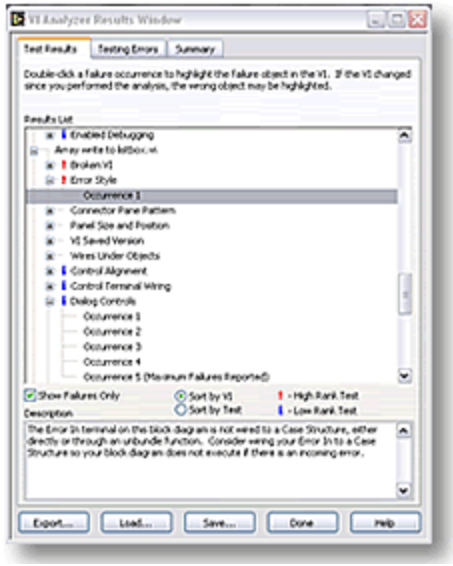
IV. Can tweak parameters

Tools>>Options Block Diagram: Cleanup

V. Cannot cleanup partial Block Diagram

VI. Can undo

VI Analyzer Toolkit



I. Automated application code review

II. Over 60 included tests

III. Customize tests for individual applications

IV. Programmatically configure and run tests

V. Report generation for documentation and for tracking progress of code quality

VI. Included in Developer Suite

References

- Refactoring: Improving the Design of Existing Code
  by Martin Fowler, Kent Beck, John Brant, William Opdyke, Don Roberts
  Publisher: Addison-Wesley Professional (July 8, 1999)
  Language: English
  ISBN-10: 0201485672
  ISBN-13: 978-0201485677

- Wikipedia – Code refactoring
  http://en.wikipedia.org/wiki/Code_refactoring

- Eyes on VIs Blog
  http://eyesonvis.blogspot.com/2008/08/automatic-block-diagram-clean-up-one-of.html

- ALE System Integration website:
  http://www.aleconsultants.com

- NI Week 2007 and 2008 Presentations on Refactoring

- National Instruments Website
  http://www.ni.com

ALE SYSTEM INTEGRATION

http://www.aleconsultants.com – info@aleconsultants.com

- LabVIEW, LabWindows/CVI, TestStand, Visual Studio
- Customers: Test Labs, Manufacturers, Mil/Aero, Finance
- Based in Long Island, New York – projects nationwide
- National Instruments Certified Alliance Partner
- Over 10 Years Test & Automation experience
- Expertise in variety of instrument manufacturers' products
- All developers have National Instruments Certification

NATIONAL INSTRUMENTS
Certified Alliance Partner

Terry Stratoudakis, P.E.

- B.S. and M.S. in Electrical Engineering, Polytechnic University
- NI Certified LabVIEW Developer and Certified Prof. Instructor
- New York State licensed Professional Engineer
- Former Assistant Adj. Prof. at NYC College of Technology
- Co-founder and President of ALE System Integration
- Worked at Underwriters Laboratories for six years
- Test & Control, OPC, DAQ, GPIB instrument control, sound & vibration analysis,, FPGA programming, and project management
- Member of the IEEE-Long Island Consultants Network