LabVIEW NXG Lessons Learned Overview

GDevCon N.A.

October 20, 2021

Presenter:

Eric Reffett

Chief Product Planner

Contributors:

Greg Richardson

Software R&D Distinguished Engineer

Deborah Burke

Software Product Planning Manager

JR Andrews

Software R&D Director

Agenda

- LabVIEW NXG Decision-Q4 2020
- NI’s goals for the retrospective
- Review themes and lessons learned
- Q&A

A brief word from Omid Sojoodi, Sr. VP, R&D at NI:

Changes NI made in Q4 2020

Listening to our users

Concerns around migrating or rewriting years of LabVIEW code

Focus energies on addressing future test and measurement needs

NI will be taking the following steps

1. Integrate the strengths of LabVIEW NXG into LabVIEW 2021(+)
2. Continue to advance NXG-based portfolio of software
    - NXG Web Module is now “G Web Development Software”
    - SystemDesigner technology used to create Hardware Configuration Utility
    - Expand capabilities of FlexLogger and VeriStand
3. Discontinue separate LabVIEW NXG product
    - Last release was version 5.1 in 2021

FlexLogger™

InstrumentStudio™

LabVIEW™

VeriStand™

Digital Pattern Editor™

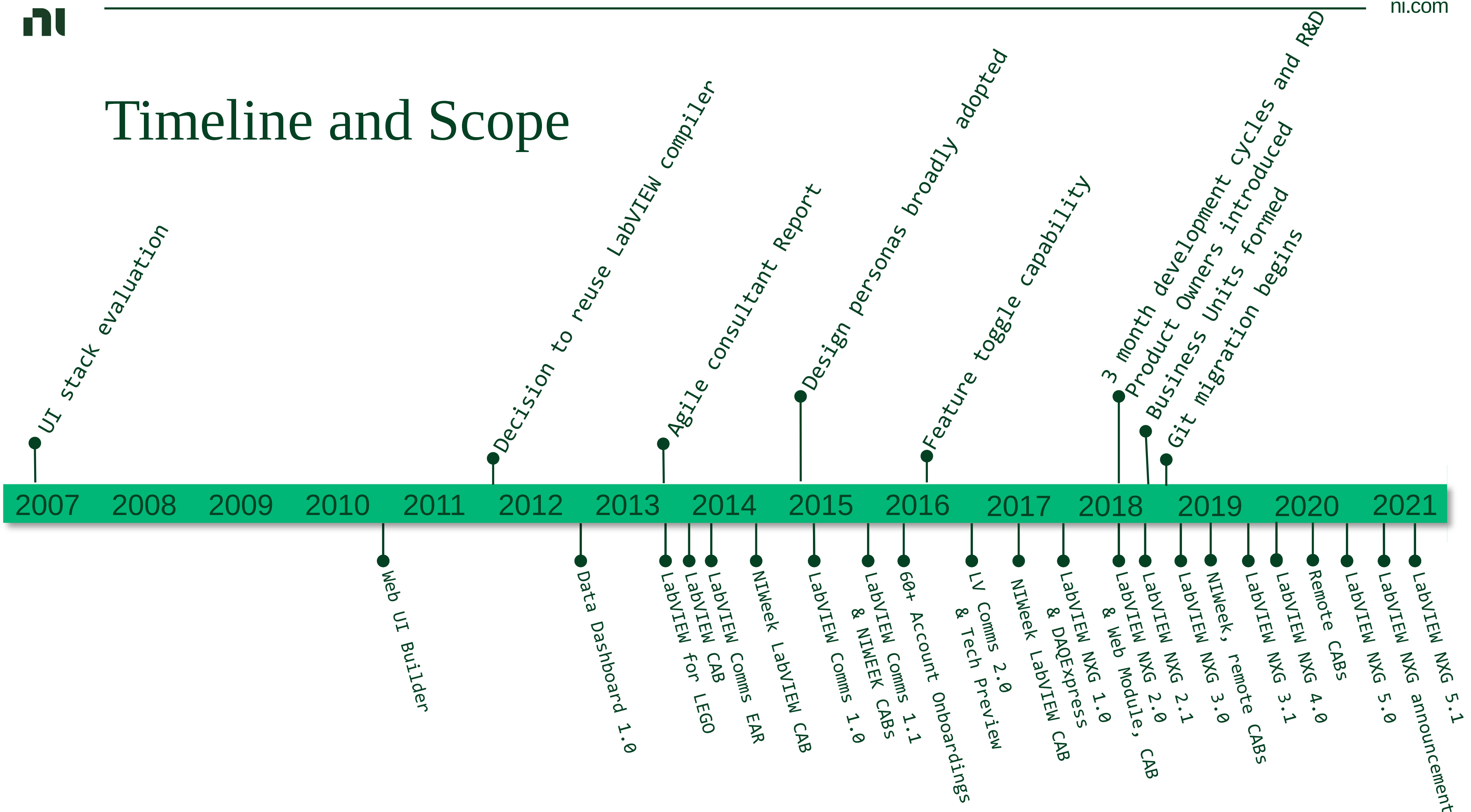
NI’s Goals for the Retrospective

Understanding key issues and challenges throughout the project lifespan

Reviewing the mitigations and changes we made

Sharing of experience and learning within the organization and externally

Timeline and Scope



2007 2008 2009 2010 2011 2012 2013 2014 2015 2016 2017 2018 2019 2020 2021

UI stack evaluation

Web UI Builder

Decision to reuse LabVIEW compiler

Data Dashboard 1.0

Agile consultant Report

LabVIEW for LEGO

LabVIEW CAB

LabVIEW Comms EAR

NIWeek LabVIEW CAB

Design personas broadly adopted

LabVIEW Comms 1.0

LabVIEW Comms 1.1
& NIWEEK CABs

60+ Account Onboardings

Feature toggle capability

LV Comms 2.0
& Tech Preview

NIWeek LabVIEW CAB

LabVIEW NXG 1.0
& DAQExpress

3 month development cycles and R&D

Product Owners introduced

Business Units formed

Git migration begins

LabVIEW NXG 2.0
& Web Module, CAB

LabVIEW NXG 2.1

LabVIEW NXG 3.0

NIWeek, remote CABs

LabVIEW NXG 3.1

LabVIEW NXG 4.0

Remote CABs

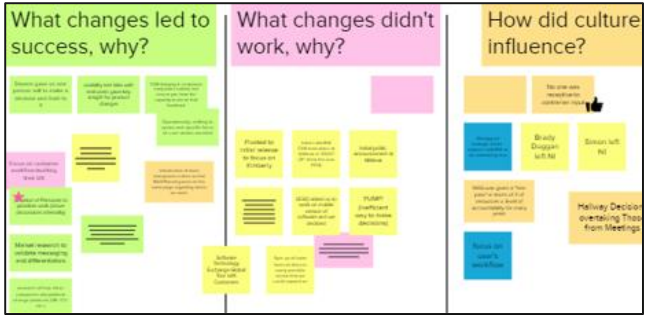
LabVIEW NXG 5.0

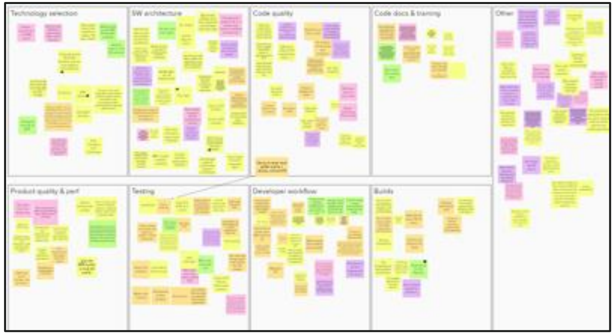
LabVIEW NXG announcement

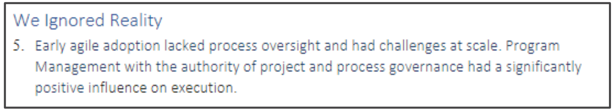
LabVIEW NXG 5.1

LabVIEW NXG Lessons Learned

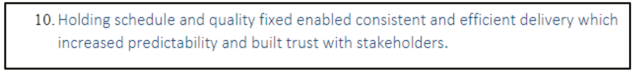
Our Approach

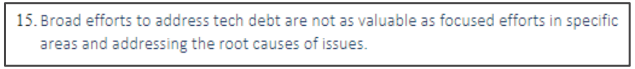

What changes led to success, why?
What changes didn't work, why?
How did culture influence?
Brady Duggan left NI
Simon left NI
Hallway Decision overtaking Those from Meetings
Focus on user's workflow
Software Technology Exchange/Shared Tool with Customers




We Ignored Reality

5. Early agile adoption lacked process oversight and had challenges at scale. Program Management with the authority of project and process governance had a significantly positive influence on execution.


10. Holding schedule and quality fixed enabled consistent and efficient delivery which increased predictability and built trust with stakeholders.


15. Broad efforts to address tech debt are not as valuable as focused efforts in specific areas and addressing the root causes of issues.

- Chose to drive internally vs. hiring external consultant
- Mural for brainstorming across virtual participants
- Interviewed key decision makers and IC's
- Identified 17 takeaways and condensed into three key themes
  - Process, Strategy and Technical

High Level Takeaways

Listen More

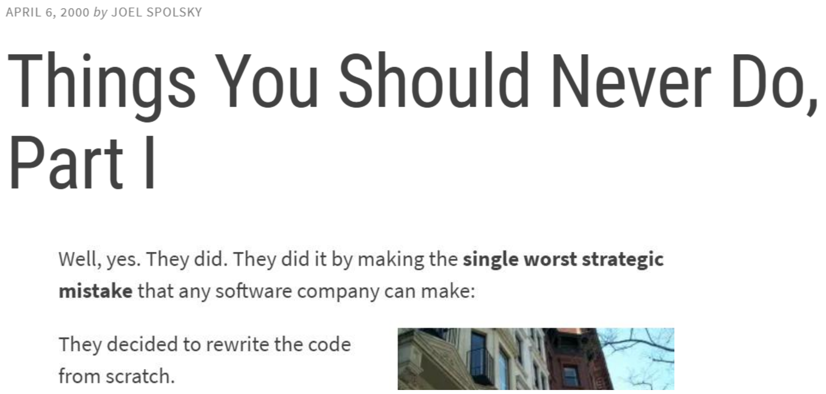
Solicit and Understand Feedback

Acknowledge Reality

Focus on Sustainable Technical Investment

Listen More

Perils of a Platform Rewrite

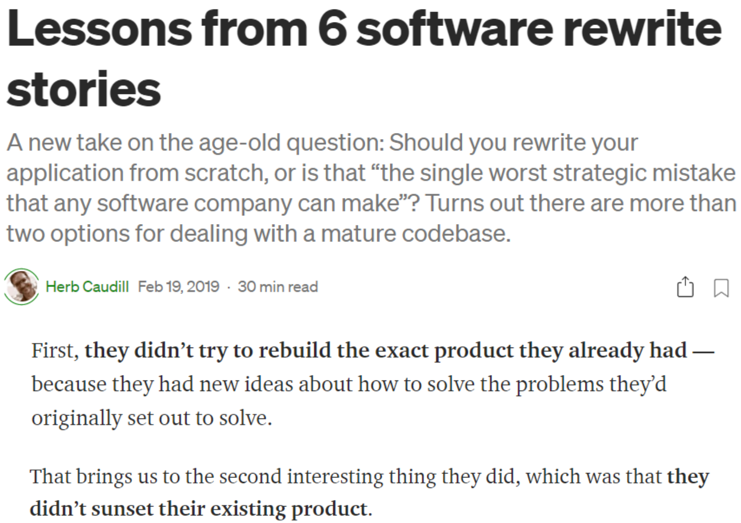


APRIL 6, 2000 *by* JOEL SPOLSKY

Things You Should Never Do, Part I

Well, yes. They did. They did it by making the **single worst strategic mistake** that any software company can make:

They decided to rewrite the code from scratch.



Lessons from 6 software rewrite stories

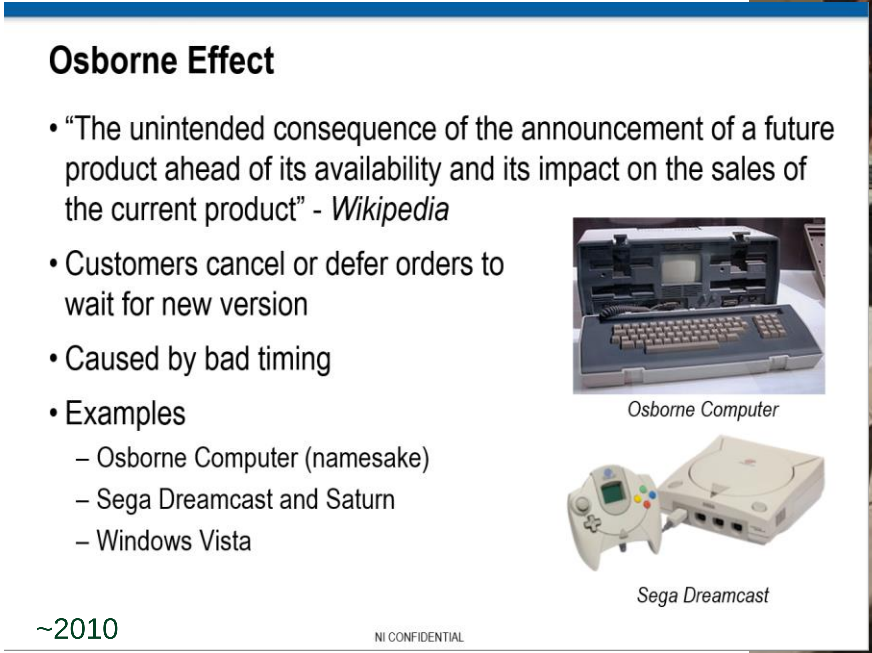
A new take on the age-old question: Should you rewrite your application from scratch, or is that “the single worst strategic mistake that any software company can make”? Turns out there are more than two options for dealing with a mature codebase.

Herb Caudill Feb 19, 2019 · 30 min read

First, **they didn’t try to rebuild the exact product they already had —** because they had new ideas about how to solve the problems they’d originally set out to solve.

That brings us to the second interesting thing they did, which was that **they didn’t sunset their existing product**.

“Inside-Out” Pursuit of Platform Replacement



Osborne Effect

- “The unintended consequence of the announcement of a future product ahead of its availability and its impact on the sales of the current product” - *Wikipedia*
- Customers cancel or defer orders to wait for new version
- Caused by bad timing
- Examples
  - Osborne Computer (namesake)
  - Sega Dreamcast and Saturn
  - Windows Vista

Osborne Computer

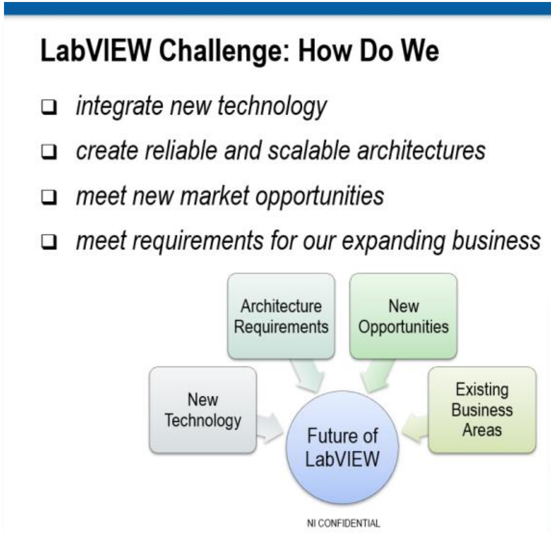
Sega Dreamcast

NI CONFIDENTIAL

~2010

~2016

“Inside-Out” Pursuit of Platform Replacement AND New Users



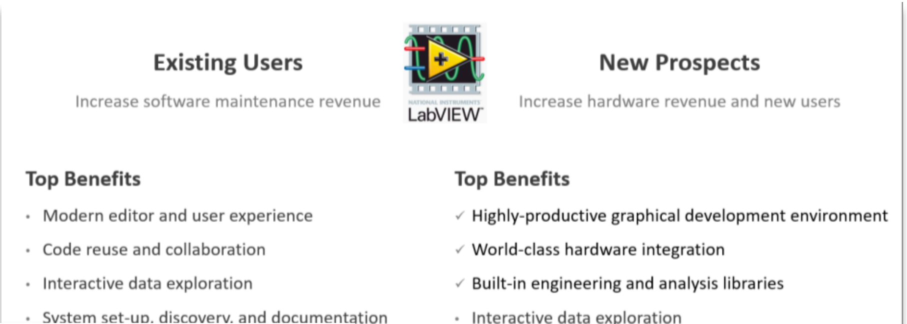
LabVIEW Challenge: How Do We

- ❑ *integrate new technology*
- ❑ *create reliable and scalable architectures*
- ❑ *meet new market opportunities*
- ❑ *meet requirements for our expanding business*

Architecture Requirements
New Opportunities
New Technology
Existing Business Areas
Future of LabVIEW

NI CONFIDENTIAL

2010



Existing Users

Increase software maintenance revenue

New Prospects

Increase hardware revenue and new users

Top Benefits

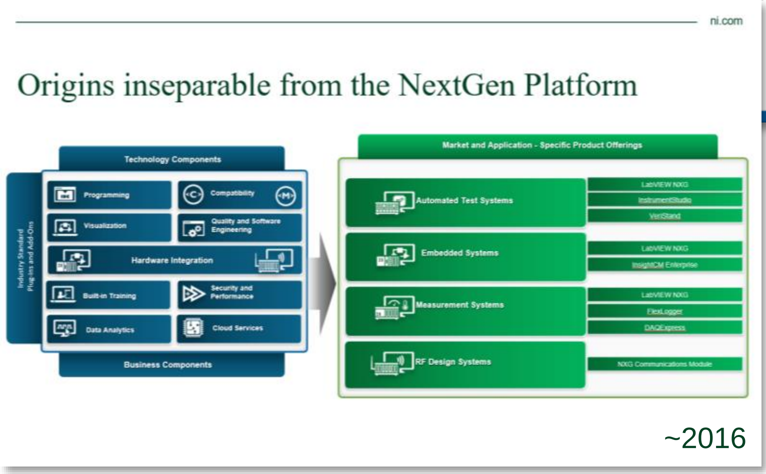
- Modern editor and user experience
- Code reuse and collaboration
- Interactive data exploration
- System set-up, discovery, and documentation

Top Benefits

- ✓ Highly-productive graphical development environment
- ✓ World-class hardware integration
- ✓ Built-in engineering and analysis libraries
- Interactive data exploration
- System set-up, discovery, and documentation

NATIONAL INSTRUMENTS

~2016



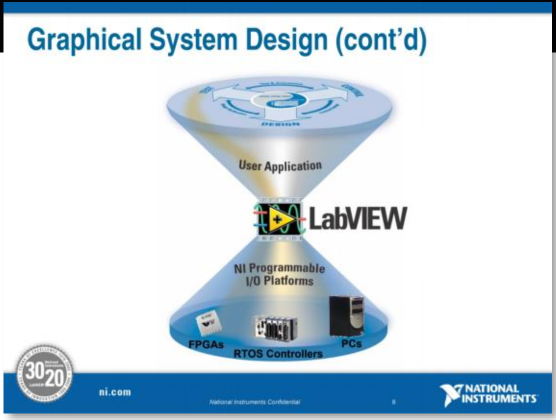
ni.com

Origins inseparable from the NextGen Platform

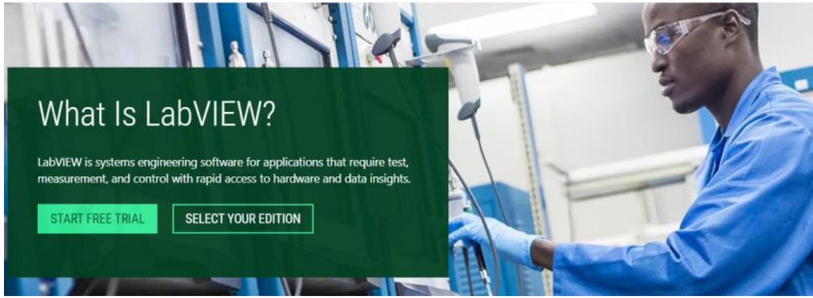
~2016

LabVIEW’s Focus Becoming More “Outside-In”

NIWeek 2012


Graphical System Design (cont’d)

~2006


What Is LabVIEW?

LabVIEW is systems engineering software for applications that require test, measurement, and control with rapid access to hardware and data insights.

ni.com today

Solicit and Understand Feedback

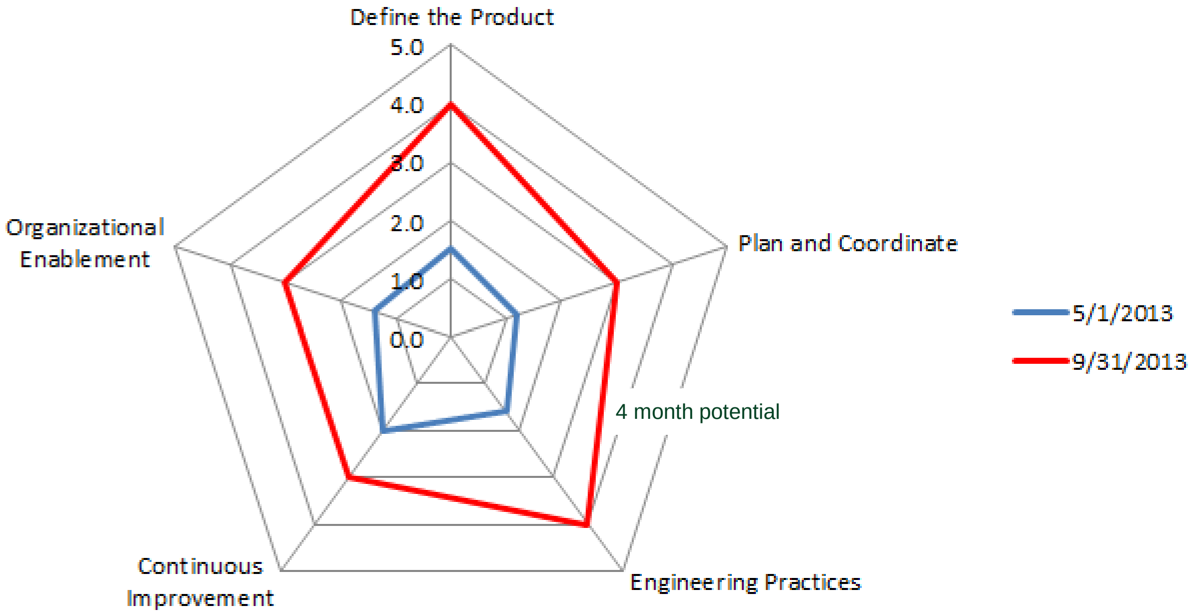
“Satisfaction with LabVIEW is strong, with 67% of survey participants saying they are “extremely/very satisfied” with the version they are currently using.

Barriers to LabVIEW NXG adoption are primarily that customers’ current version of LabVIEW is meeting needs and migrating to LabVIEW NXG would be too much work.”

-Q3 2020 LabVIEW Customer Market Research Study Summary

Agile Consultant Feedback

They thought we could make significant improvements quickly with clearer requirements and acceptance criteria, improved risk management and metrics but feedback we didn't understand



Embracing a “Speak Up Culture”

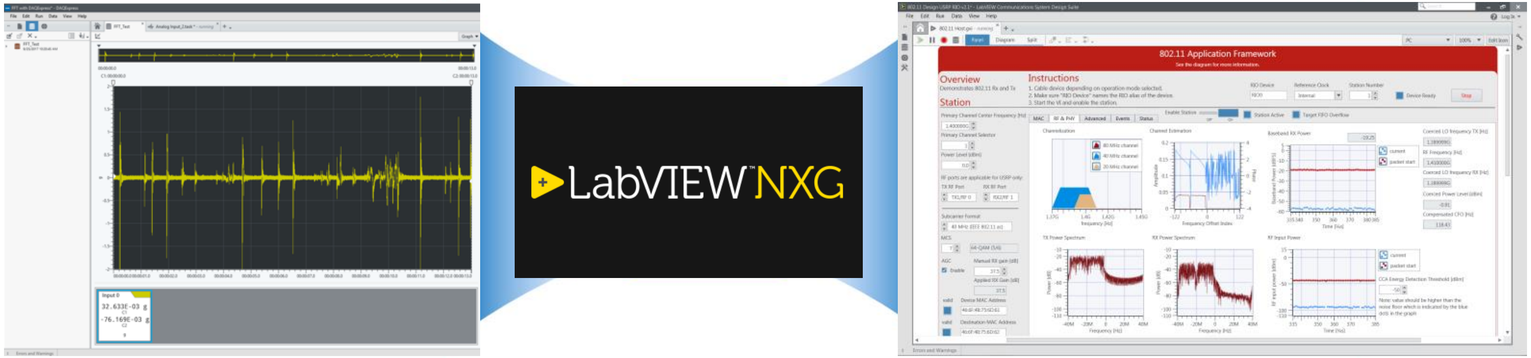
- Expressing concerns can be productive, doesn’t mean you aren’t ambitious
- Adopting a culture of learning vs. knowledge
- Valuing outside-in influence

Acknowledge Reality

Shipping Products Seen as Success Indicator

- DAQExpress & LabVIEW Communications strategy overlooked platform gaps
- We prioritized feature velocity, sacrificing quality, architecture, and technical debt

From Kindergarten to Rocket Science…


LabVIEW NXG
802.11 Application Framework
See the diagram for more information.
Overview
Demonstrates 802.11 Rx and Tx
Station
Instructions

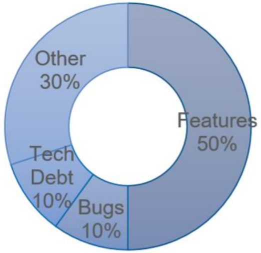
1. Cable device depending on operation mode selected.
2. Make sure "RIO Device" names the RIO alias of the device.
3. Start the VI and enable the station.

Knowing Our Velocity

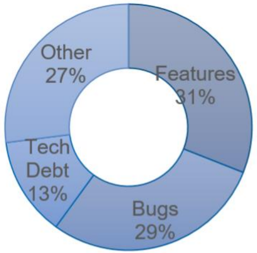
- Shifted from high level swag ‘person’ estimates to Person Quarters
- Moved from large feature estimates on long projects to vertical slices of value delivery
- Included bugs in total scope of work and adopted an age-based bug policy

Predictability and Keeping Commitments Builds Trust

Plan



Actual



21Q1 20.7 release	• 01 January 4 – January 29 • 02 February 1 – February 26 • 03 March 1 – March 26
Inter-Cycle Week	• March 29 – April 2
21Q2	• 04 April 5 – April 30 • 05 May 3 – May 28 • 06 May 31 – June 25
Inter-Cycle Week	• June 28 – July 2
21Q3	• 07 July 5 – July 30 • 08 August 2 – August 27 • 09 August 30 – September 24
Inter-Cycle Week	• September 27 – October 1
21Q4	• 10 October 4 – October 29 • 11 November 1 – November 26 • 12 November 29 – December 24
Inter-Cycle Week	• December 27 – December 31

Focus on Sustainable Technical Investment

Focus on Sustainable Technical Investment

- Performance
- Technology selection
- **Architectural simplicity**
- **Technical debt**
- Effective testing
- **Code quality**
- Public API
- Developer efficiency

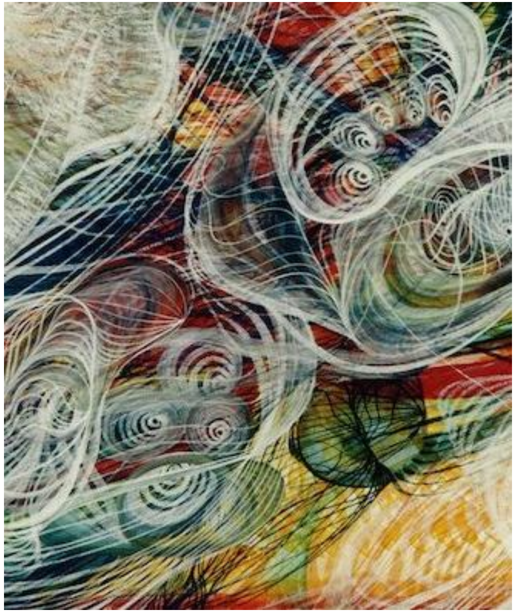
Challenge: Need Architectural Simplicity

“Reduce complexity”

- Agile consultants, 2013

“We don’t have so much a documentation problem but an abstraction problem.”

- NI developer, 2017



Architectural simplicity requires maintenance

Agile development == increased architecture evolution

Do This….

….Not This

Sustained Velocity is the key

- Keep code clean as you go
- Refactor as needed, but only what is needed

Introduced Code Owners to monitor code health

Consistently reevaluate abstractions

- Code reflects intent & developer mental model

Push changes through all code

- Feature work, not tech debt

Challenge: Technical Debt

TECHNICAL DEBT
I DON'T UNDERSTAND WHY IT TAKES SO LONG TO ADD A NEW WINDOW.
@VINCENTDNL

Schedule is sole priority

To go fast, go well

Target outcome not code

Focus on the things that will have the biggest impact

- Think “what will impact my future velocity”

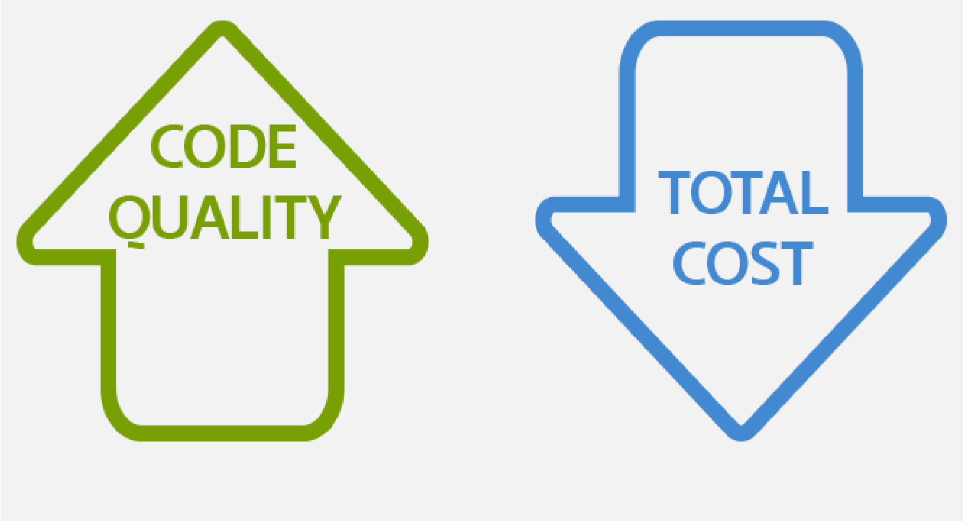
Conduct honest evaluation of impact

- Reduce time to add features
- Reduce bugs
- Improve developer sentiment

An example over time

- 2011 – document all public APIs
- 2018 – document key identified developer workflows

Challenge: Maintain Code Quality



Intuitive relationship

- Easier to modify
- Fewer bugs
- Happier developers

One major problem

- Improving code quality in existing codebase is daunting

Small change / magnified impact

Focus on small things that will have a big impact

- Enforced through code review

Emphasize readability in review

- Descriptive naming
- Focused function size

Instill a mindset of not letting things go

Results:

- Entire team empowered
- Creates willingness to refactor

High Level Takeaways

Listen More

Solicit and Understand Feedback

Acknowledge Reality

Focus on Sustainable Technical Investment

Questions?

- What features from LabVIEW NXG will come back to LabVIEW?
  - Shameless plug for my other session: “What comes next for LabVIEW”, Oct 21, 3pm
- What else did you want to hear about today?

- Please fill out our Event Survey for a chance to win one of two myRIO devices
  - https://tinyurl.com/35k3rcsc

When survey is active, respond at **pollev.com/devtoolsteam711**

Please give NI feedback on this presentation.

0 done

0 underway

Start the presentation to see live content. For screen share software, share the entire screen. Get help at **pollev.com/app**