30 Years of Regression Testing: Past, Present and Future

Jean Hartmann
Test Architect
Office Client Services - Test
What is Regression Testing?

From Wikipedia: “Software testing that seeks to uncover new software bugs or regressions in existing functional and non-functional areas of a system after changes, such as enhancements, patches or configuration changes, have been made to them. The intent of regression testing is to ensure that a change, such as a bug fix, did not introduce new faults. One of the main reasons for regression testing is to determine whether a change in one part of the software affects other parts of the software”.
30 (or More) Years Ago...

- 1972 - Appears in papers at 1st software testing conference
- 1970s-1980s - Becomes mainstream, appears in classic reference text
- Since 1980s - Interest from academia and industry
- Today, more useful than ever...
Regression Testing Strategies

- Retest-all - re-running entire test suite to ensure no regressions
  - Costly in terms of test execution and failure analysis
- Regression test selection (or selective revalidation) - re-running subset of tests for less cost and with same bug detection capability as retest-all strategy

Let’s take a look at regression testing and what has impacted it over the decades...
Major Industry Trends and Their Impact

- **Hardware**
  - Mainframes and mini-computers to desktop PCs and Sun (SPARC) workstations

- **Software**
  - Assembler to procedural and object-oriented languages

- **Testing**
  - Manual (error-prone) to automated (repeatable)

- **Impact on regression testing**
  - Code base and number of automated tests is growing
  - Test execution is now on demand
  - Test passes still taking a long time!
1980’s: Genesis

- Early 1980’s, Kurt Fischer proposed a mathematical approach
  - Based on operations research, specifically integer programming models
  - Goal - select an optimal (minimal) set of regression tests
  - Code coverage data to build the model
  - Mathematical solver to solve model - did not scale well at the time

Minimize \( c_1x_1 + c_2x_2 + \ldots + c_nx_n \)  
subject to
\[
\begin{align*}
a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n & \geq b_1 \\
a_{21}x_1 + a_{22}x_2 + \ldots + a_{2n}x_n & \geq b_2 \\
\vdots \\
a_{m1}x_1 + a_{m2}x_2 + \ldots + a_{mn}x_n & \geq b_m
\end{align*}
\]
\( x_1, x_2, \ldots, x_n \in \mathbb{Z}_+ \)  

(objective function)

(functional constraints)

(set constraints)
1980’s: Data flow-based Testing

- Much publicized, academic testing approach
- Based on static code analysis to compute so-called def-use (DU) pairs
- Goal - select tests based on changes to code variables (ripple effect)
- Suffered from limitations of static code analysis
  - Costly to compute
  - Interprocedural analysis was approximate
  - Feasibility of def-use pairs?
1990’s: Putting Theory into Practice

- Researchers realize tools were needed to demonstrate and reap benefits
- Active collaborations between academia and industrial research labs
- Spawned multi-million dollar tool development efforts
  - Tactic @ Siemens
  - ATAC @ Bellcore/Telcordia
  - TestTube @ AT&T Bell Labs
- Tools enabled larger empirical studies to be conducted
- Studies successful and well-publicized

- ...But, the issue was still adoption by business units!
2000’s: Large-Scale Application

- Large-scale application required research talent, domain expertise AND resources
- Magellan toolset is a mature, widely-used toolset within Microsoft
  - Includes code coverage, code churn and test selection technology
  - Windows Sustained Engineering (WinSE)
- Based on Fischer’s approach
- Achieving significant reductions in tests for rerun
2010’s and Beyond: Mobile and Online Services

Until Today...
- Mostly shrink-wrapped products, packaged and delivered on CDs/DVDs
- Development lifecycles have been extensive, e.g. three years for new Office release
- Post-checkin regression testing
  - Large, regular test passes to validate breadth and depth of product at regular milestones
  - Process owned by Test

Tomorrow...
- Shift towards more upstream quality and quicker deployment cycles
- Driven by mobile and online service demands
- Pre-checkin regression testing
  - On-demand, focused unit/component testing
  - Process owned by Dev
A Story with A Twist:
Integrated Circuit Testing

- IC or integrated circuits are extensively tested
- Large numbers of test vectors generated as part of a testbench (or test suite)
- Used to validate behavioral code written in VHDL or Verilog
- Goal is to reduce execution time (in simulator)
- Example of a software technique being applied to hardware problem 😊
Do-It-Yourself: Regression Test Selection Tools

Ingredients
- Code coverage tool that can generate trace data
- Code churn tool that can identify code changes
- Test selection tool or integer programming model solver

Method
1. Run your tests against your instrumented product code or covbuild
2. Identify two product builds and run code churn tool against the code
3. Input coverage and churn data into test selection tool to generate subset of tests

Result
- Subset of regression tests that exercise the changed portions of product (or not)
- Hook up to test case management (TCM) tool for automatic test execution
Summary

- Chronicled the journey of regression test selection over the past thirty years
- Highlighted some of the major milestones in its development
- Examined how major industry trends have influenced it
- Provided thought on how to apply it in your organization