Software Engineering Experimentation

Example Experimental Design

Jeff Offutt
http://www.ise.gmu.edu/~offutt/
Robert Nilsson

Dialogue With a Student

- Robert is working on his PhD dissertation
- This summer decided to design an empirical study and suggested an initial experimental design
- I made some comments
- These slides summarize our discussion

Introduction

- Purpose: Assess applicability of a testing framework for testing timeliness properties of real-time software
- Measure ability to find hand-seeded timeliness faults
- Compare with randomly generated tests
- Difficulty : Finding software artifact to test on

Experimental Design

1. Design the application
2. Set up real-time platform
3. Instrument the target platform to obtain results of testing (in particular, timeliness results)
4. Implement application
5. Perform unit testing
6. Create versions of application with seeded faults

Experimental Conduct

- Model the system using tools in testing framework
- Generate timely-mutation tests automatically
- Run tests timely-mutation
- Run random tests
- Run all tests on all mutants

Comments on Experimental Design

1. Design the application

- Important to separate knowledge
  - Knowledge of the test method
  - Knowledge of the faults
  - Knowledge of the application
- Desires (to show Robert is right) must be separated from knowledge of faults and application
- Different people need to implement and create faults
- Laboratory software may not be "representative" of real software

Comments on Experimental Design

2, 3. Set up and instrument real-time platform

- Make sure that framework and instrumentation does not bias results
  - Possible confounding variable

Comments on Experimental Design

5. Perform unit testing

- Knowledge from generating one set of tests cannot influence the other set
  - Either generate automatically,
  - Or use different people
- If not, generate mutation-tests first
  - Ensures bias is in favor of random tests

Comments on Experimental Design

6. Seeding faults

- Testing technique is mutation testing
- Mutation introduces specialized faults into the program, then asks tester to find tests that "find" those faults (result in failure)
- VERY IMPORTANT:
  - Seeded faults must NOT look like mutants
  - If they do, this introduces a very strong bias
  - We measure how accurate our measurement is by comparing the meter-stick with the same meter-stick

Comments on Experimental Design

6. Seeding faults

- Creating non-mutant faults
  - Get somebody who knows nothing about mutation or testing to create faults (nothing about nothing)
  - Theoretically create non-mutant faults (I'm not sure how)
  - Mutants usually involve simple changes – create faults that are more "complicated" than mutants
  - Use naturally occurring faults, or faults that are designed on naturally occurring faults

