ESOF 522: Empirical Software Engineering

Dr. Clemente Izurieta
Montana State University
Terminology

Experimental Units/Objects

The artifacts on which the experiments are run:
- A patient in medicine
- Land in agriculture
- In software engineering:
  - A process
  - A part of a process, i.e. the design phase
  - A product
  - A design pattern
  - A section of code (module, class, method)
Terminology

Experimental Subjects

- The person(s) that applies the methods to the experimental units
- A team of developers
- A single developer
- Testers
- Program or project manager when the experimental units are processes

Results can vary significantly. Why?

- Subject’s experience
- Subject’s mood
Response/Dependent Variable
   The resource that is measured
   Experiment: To test the variations of variables
   Case Study: To observe changes
   The outcome of an experiment that is under analysis
   Quantitative
   Can be measured using direct or surrogate metrics
   An observation is the measure(value) associated with a single instance of a dependent variable obtained from a unitary experiment
   A unitary experiment is one occurrence of running an experiment by one subject on one unit
Terminology

**Parameters**
- Any characteristic (qualitative or quantitative) that does not influence, or we do not want it to influence the outcome of the experiment

**Factors or Independent Variables**
- Provoked variations
- Affect the results of the experiment
- The different values that an independent variable can have are called treatments, alternatives or levels
Terminology

Interactions
- Sometimes factors depend on the values of other factors. They are not really independent.
- Can influence the response variable
- **Factorial designs** allows us to take into consideration the effects of alternatives in factors and their interactions

Blocking Variables
- Not all variables can be set to some constant value
- Undesired variations from one experiment to another
  - Subjects: Not all subjects have the same experience
  - Experimental Units: There aren’t enough to experiment on
- Require a special type of design: **Block designs**
Terminology

Experimental Error

- Outcomes of performing experiments can never be equal
- Variations are inevitable
- Sometimes referred to as noise or uncertainty
  - Random
  - Systematic

How do we identify the sources of experimental errors?
Scientific Inquiry

- Prior Knowledge (Initial Hypothesis)
- Observe (what is wrong with the current theory?)
- Theorize (refine/create a better theory)
- Design (Design empirical tests of the theory)
- Experiment (manipulate the variables)
How to Experiment

You must have a theory/hypothesis

Iterate and Refine:

- Establish relationships between variables that shed light into new or incremental change to some phenomena
- Use/collection measurements that objectively quantify changes
- Show statistical significance

Validate observed phenomena
Relationships

Descriptive
- Surveys usually produce data that can be described. For example, a normal distribution

Correlational
- Case studies help build empirical models. These are just correlations! in some context unless all confounding factors are accounted for (not easily done in case studies)

Causal
- Experiments allow the development of theoretical models. By having full control, we can vary individual variables that explain causal relationships
Refinement

In Software Engineering there are no established theories that we can use as starting points for investigations but we do have empirical models.

Start with small, parsimonious models and make small changes.

Clearly exhaustive variation of variables is prohibitive so we use an experimental design.
Process of Experimentation

- **Objective Definition**
- **Design**
- **Execution**
- **Analysis**

- Hypothesis for testing
- Experimental Design
- Experimental Results
- Hypothesis Tested
Objective Definition
- Develop a new hypothesis
- Develop an incremental hypothesis from prior theoretical or empirical models
- Make the hypothesis quantifiable
- What questions are you trying to answer
- Do the metrics selected make sense? Do you have the correct surrogates?
Experimentation

Design
- Choose or develop an experimental design
  - There are well known design methods for controlled experiments, but it is much harder when dealing with case studies
  - Case studies are not a subset of experiments and have their own designs
- Determine (potential) variables that affect the experiment

Goal:
- Maximize the information and knowledge gained from performing as few experiments as possible
Experimentation

Execution
- Run the experiment and collect results
- Calculate descriptive statistics

Analysis
- **Goal:** Seek to find relationships between variables
  - Descriptive: Informal analysis
  - Correlation and Causal: Formal analysis that requires the use of statistics
- To establish causal relationships you need a significant corpus of empirical studies and theories
Experimentation

Variable A

Variable B

No statistical significance
Can we make Inferences about the data?

Variation is due to chance or a variable not considered

Statistical significance

Significance Testing

Are the variances in the data statistically significant

Some combination of values and variables causes the improvements

Statistical significance

Some combination of values

Means, variances, frequencies, ratios

Significance Testing

Are the variances in the data statistically significant

Statistical significance

Some combination of values and variables causes the improvements

Statistical significance

Some combination of values

Means, variances, frequencies, ratios
Design of an Experiment

What is your hypothesis?

Decide the type of study

Choose factors and parameters

Choose response variables and metrics
Example

Goal: Study the ease of creating a program using an aspect oriented approach and an OO approach

Null Hypothesis:
H₀: There is no difference between creating a program using an aspect oriented approach versus an OO approach

Alternate Hypothesis:
H₁: There is a difference between creating a program using an aspect oriented approach versus an OO approach
Example

Parameters:
- Problem complexity (low)
- Application type (single threaded iPhone app)
- Subjects from ESOF 522
- Code size ($500 < \text{LOC} < 1000$)

Factors:
- Programming Approach: Aspect Oriented or Object Oriented
Example

Response Variables:
- Can only be measured a-posteriori
- Must be quantitative for analysis
- We can have multiple response variables. Each may require a different analysis technique

Development process: schedule deviation, budget deviation, process compliance

Methods: efficiency, usability, adaptability

Resources: productivity

Products: reliability, maintainability, design correctness, level of code coverage
# Internal and External Metrics

<table>
<thead>
<tr>
<th>Entities</th>
<th>Internal Attributes</th>
<th>Metrics</th>
<th>External Attributes</th>
<th>Metrics</th>
</tr>
</thead>
<tbody>
<tr>
<td>Products</td>
<td>Specifications</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Size</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>- number of classes</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>- number of atomic process</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Reuse</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>- number of classes used without change</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>Maintainability</strong></td>
<td>person.months spent in making a change</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Functionality</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>- number of function points</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>Maintainability</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>number of modules affected by a change in another one</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Syntactic correctness</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>- number of syntactic faults</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>Maintainability</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Design</td>
<td></td>
<td><strong>Size</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>- number of modules</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>Maintainability</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Reuse</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>- number of modules used without change</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>Maintainability</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Coupling</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>- number of interconnections per module</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>Maintainability</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Cohesiveness</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>- number of modules with functional cohesion/total number of modules</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Code</td>
<td></td>
<td><strong>Size</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Non-comment lines of code (NCLOC)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>Quality</strong></td>
<td>defects/LOC</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Complexity</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>- number of nodes in a control flow diagram</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>- McCabe’s cyclomatic complexity</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>Usability</strong></td>
<td>hours of training before independent use of a program</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>Maintainability</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>days spent in making a change</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>Efficiency</strong></td>
<td>execution time</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>Reliability</strong></td>
<td>number of faults in a time t</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>mean time between failures</td>
</tr>
</tbody>
</table>