Experimental Software Engineering (ESE)

6E1. An Example of Experiment Definition

Credits
Group: http://eseg.uniroma2.it/

Memo

Idea → Definition

Definition:: Goal && Informal hypotheses
→ Sketches on Variables & Measurements

Practical Example

Idea

Code reading might be useful to improve the quality of software games, such as produced in the reference company X.

In fact, in literature (e.g. see UMD-CS experiments as reported by TSE) some experiments have been published, which claim that Code Reading as a testing technique helps in finding more software defects than functional testing in some kinds of applications.

Practical Example

Goal

Improves the quality of the reference company's software games with no significant changes in the cost of testing.

Practical Example

Informal hypothesis

Using Code Reading rather than Functional testing improves the quality of the reference company's software games with no relevant changes for the testing cost.

Relevance of an Informal Hypotheses

Another hypothesis

What if in the previous informal hypothesis "*rather than*" should read "*in conjunction with*"

.

Consequences

Fault classification – Classes of faults better detected by each techniques – …

From the Idea to the formal Goal

Comparing *Code Reading* and an adapted *Functional Testing*'s *Effectiveness* and *Defect Detection Rate* **for** *OO event-driven Java software* **with** *sophomore* students **for** *research*.

GQM Template

Object of study	*Purpose*	*Perspective*	*Quality focus*	*Context*
Product	Characterize	Developer	Effectiveness	Subjects +
Process	Monitor	Modifier	Efficiency	Objects +
Model	Evaluate	Maintainer	Cost	Organization +
Theory	Predict	Project Manager	Reliability	etc.
Technology	Control	Corporate manager	Maintainability	
	Change	Customer	Portability	
		User		
		Researcher		

From the Idea to the GQM Goal

Evaluate *Code reading* vs. *Functional testing* (to be adapted as necessary) by **focusing** on *Effectiveness* and *Defect Detection Rate* in the **context** of a large Internet café, with *sophomore students* as subjects and academic *Java software games* as objects, in the **perspective** of the *researcher*.

Replication 1 (GQM Goal 2)

Re-evaluate *Code reading* vs. *Functional testing* (to be adapted as necessary) by **focusing** on *Effectiveness* and *Defect Detection Rate* in the **context** of an academic classroom, with *senior students* as subjects and academic *Java software games* as objects, in the **perspective** of the *researcher*.

Replication 2 (GQM Goal 3)

Re-re-evaluate *Code reading* vs. *Functional testing* (to be adapted as necessary) by **focusing** on *Effectiveness* and *Defect Detection Rate* in the **context** of an academic Master class laboratory, with *post-graduate granted Master students* as subjects and academic *Java software games* as objects, in the **perspective** of the *researcher*.

Experiment Definition:

Dictionary

- ***Code Reading***:

- ***Functional testing***:

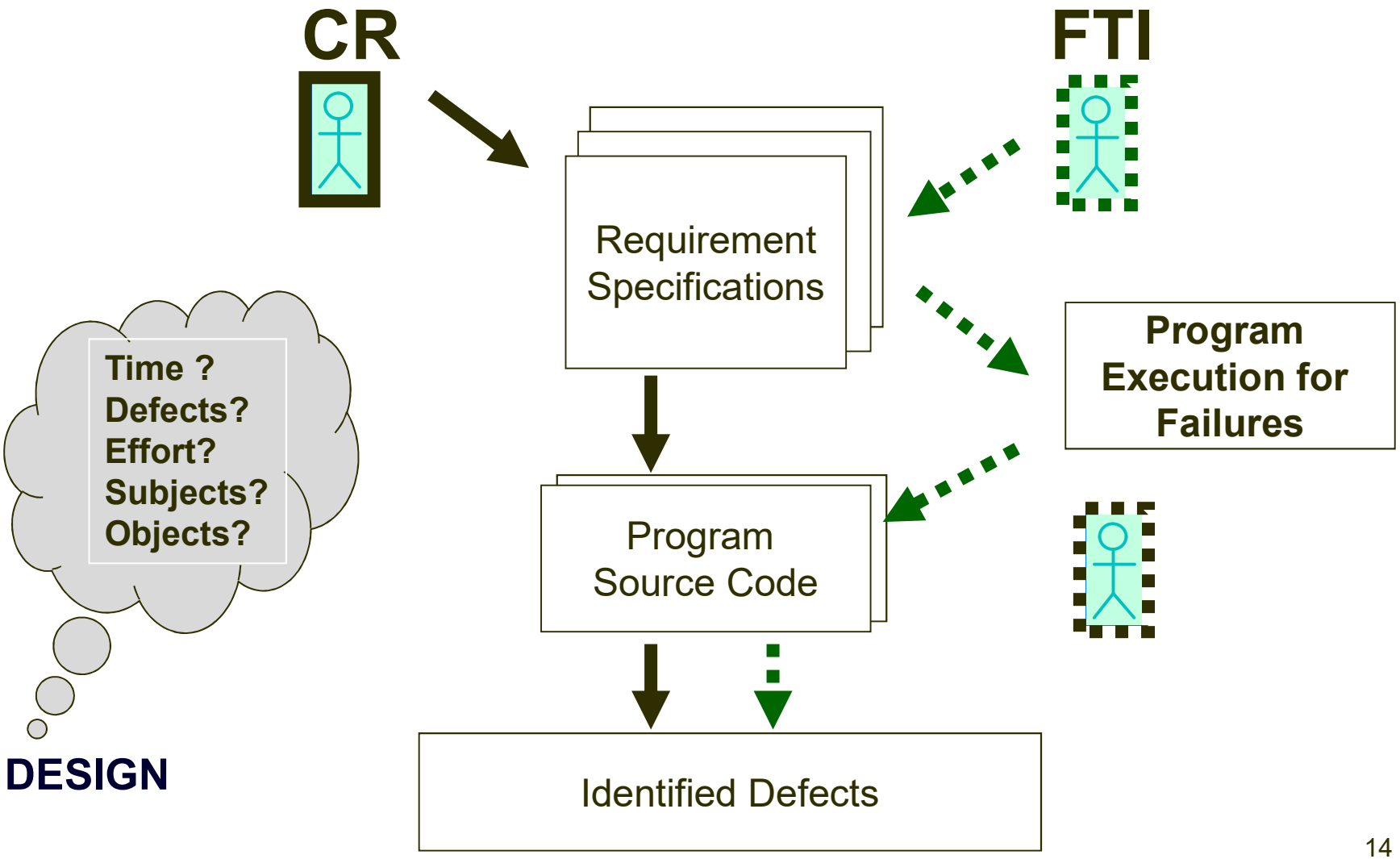
Experiment Definition:

Dictionary

- ***Code Reading*** (CR) is intended to detect software defects by entering directly to software artifacts. Once a defect is located, the location is reported; usually, the defect is also explained.
- ***Functional Testing (***FT) can detect unexpected behaviors owing to the existence of faults.
  - ***Functional Testing &*** [Defect] ***Identification*** (FTI) is an FT technique; once an unexpected behavior is detected, faults are identified in the code, e.g. they are located and motivated.

Experiment Definition: Dictionary

Techniques, Mechanisms



What Testers Have To Do

- **Code Readers** are given the program specification (intended function), some design documents (e.g. class diagram), and source code (derived function):
  - In order to detect defects, they read the code, understand behaviors, and compare the intended functionality with actual derived one. Eventually they define the class and line of code that could cause failure.
- **Functional Testers** are given the program specification and the executable version of source code.
  - They start detecting failures by comparing the intended functionality with the software actual behavior.
  - In order to identify defects, they pass to read code and possibly some other design artifacts (e.g. class diagram). Eventually they define the class and line of code that cause that failure.

Refining on the Definition of the Experiment

Experiment Definition

Experiment Conjecture

- **Baseline:** There are defects [say of kinds DA, DB and DC] in known quantities, which remains in the code of software games but should not.
- **Test manager conjecture**: Code Reading helps significantly more (e.g., +5% in the average) than Functional testing in detecting defects of those kinds.
- **Test manager motivation**: Reading was observed to perform better than Functional testing for some types of defects with expert testers (e.g. see UMD-CS experiments as reported by TSE).

Experiment Definition

Question(s)

Q1: Should we introduce code-reading in the company process for the development of software games?

Experiment Definition

ES Engineer Observation

Due to the specific type (Graphic software) of the investigated objects, FTI testers, similarly to users, interact with objects through user interfaces, rather than other means.

ES Engineer Reasoning

Because :

- The visibility of failures should be very high &&
- Time strictly necessary to detect such a failures should be very limited

As a Result :

- The FTI failure detection phase should have very minor effects.

ESE Preliminary Answer

In theory, we should **not** introduce CR in our Software Processes.

However, we need to give empirical evidence to such an answer.

Explaining the previous Preliminary Answer

FTI reading phase should perform *better* than CR with respect to the common task of identifying software defects in the implementation of games and any kind graphic-interface- bound software; that is because it is easy detect failures in the behavior of that kind of software, the FTI reading is driven by both known failures and software specifications, while the CR is driven by the software specifications *only*.

This should make a major difference.

The Experiment Idea

To match two fault detection techniques through an experiment in order to evaluate their performance with respect to different variables such as: **ad-hoc** for code reading techniques, **functional testing & fault identification** for testing techniques, time consuming, and several types of seeded faults.

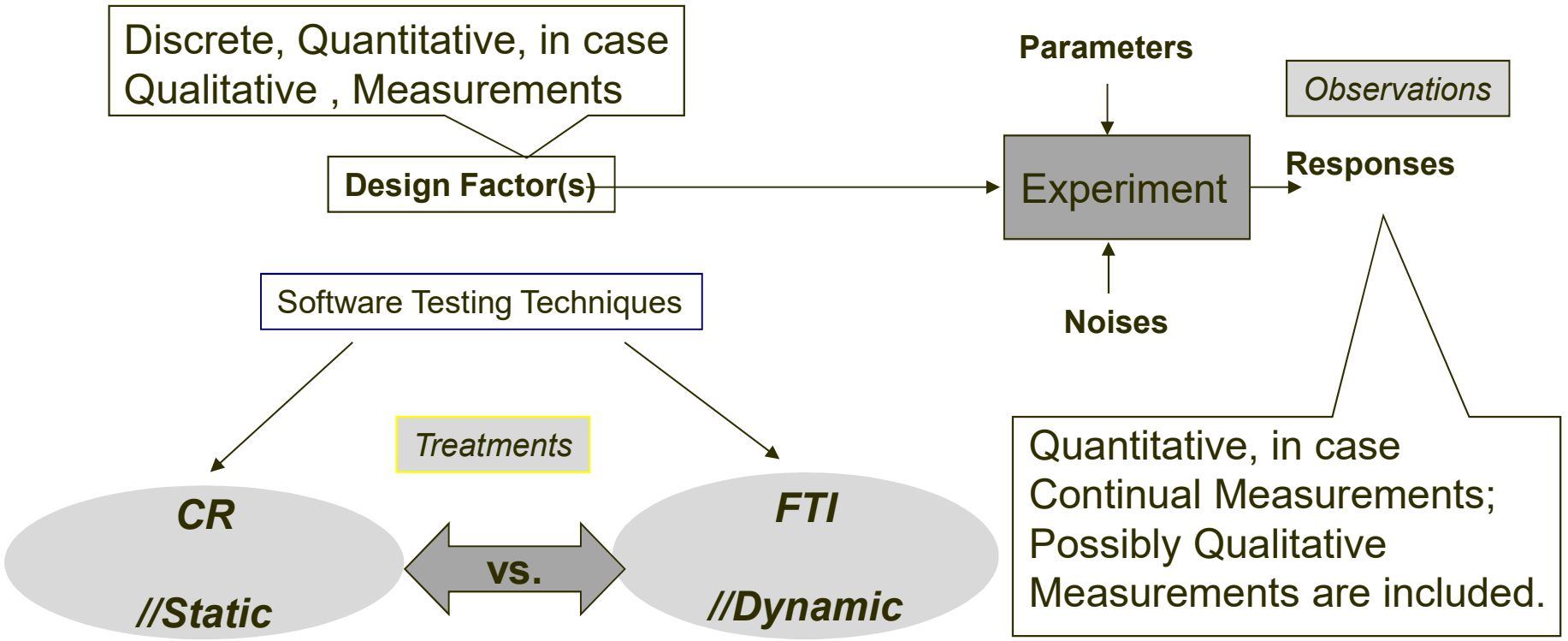
The Informal Hypothesis

The current technique, FTI, should perform better or not worse than CR for any type of defects for graphic-bound software, whatever the experience level of test people might be.

The ES Engineer

Experiment Definition

The Experiment View



Response Variables

- **Effectiveness:** Number of faults detected, in the average (Real number, Ratio scale).
- **Efficiency**: Number of faults detected per unit of detection time, in the average (Real number, Ratio scale).

Direct Measurements

- **Known Faults**: seeded faults plus new detected faults; (integer, Absolute scale).

Per subject:

- **Number of Positive Faults Detected:** (integer, Absolute scale).
- **Fault Detection Time**: (sec's, Ratio)

Indirect Measurements for the Response Variables

Technique Effectiveness $= \frac{\text{totalPositiveFaultsDetected}}{\text{totalKnownFaults}}$

where:

totalKnownFaults = (totalSeededFaults + newFound)* NumberOfSubjects

Technique Efficiency $= \frac{\text{totalPositiveFaultsDetected}}{\text{totalTestingTime}}$

Experiment Definition

Objects

- Flipper game
- Solitary card game
- Naval battle game

as developed in class and maintained by sophomores of previous academic years.

Subjects

- #100 Sophomores in Computer Science & Engineering
- #50 Seniors in Computer Science & Engineering
- #30 Post-graduate in a master course of software engineering (in a different location).

Replications

GQM. As academic researchers we want to analyze Code reading and Functional testing & fault-Identification, for the purpose of comparing their performance with respect to effectiveness and efficiency of testing event-driven OO Java software games for defect detection and classification, in the context of technology-bound software labs and CS students of **three** academic levels of experience and knowledge, and **two** universities, from the point of view of an ESE applied-research group, in the aims of transferring the resulting experiment process and guidelines to industrial settings in the mid-term.

Experiment Context Characterization:
Subject vs. Objects

		# Objects	
		One	*More than one*
# Subjects per Object	*One*		
	More than one		**Blocked subject-object study**