

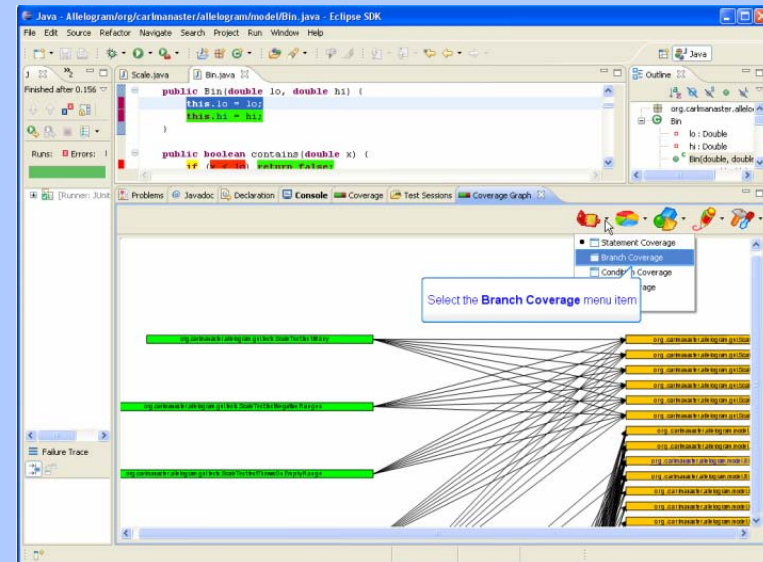
# An Empirical Evaluation to Study Benefits of Visual versus Textual Test Coverage Information

Vahid Garousi

Negar Koochakzadeh

Software Quality Engineering  
Research Group

University of Calgary, Canada



Acknowledging funding and support from:

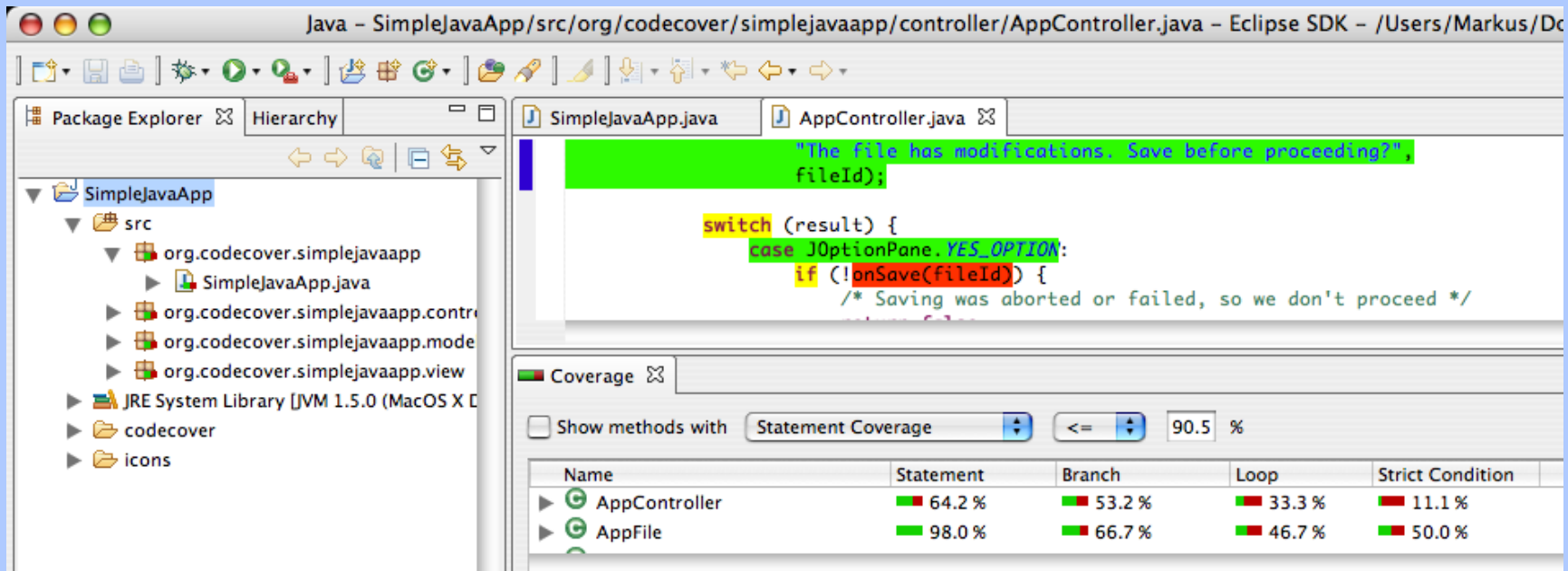


# Talk Outline

- **Background and Motivations**
- **Empirical Study - Goal**
- **Research Question**
- **Empirical Study - Setup**
- **Object of the Study**
- **Empirical Study - Execution**
- **Results**
- **Lessons Learned and Future Works**
- **Q/A**

# Background and Motivations: Existing Code Coverage Tools

- To support automated code coverage measurement and analysis...
- test coverage values are conventionally shown in percentages and are visualized by progress-bar-like green/red boxes in the existing coverage tools
- e.g., the CodeCover plug-in for the Eclipse IDE



Java - SimpleJavaApp/src/org/codecover/simplejavaapp/controller/AppController.java - Eclipse SDK - /Users/Markus/D

```

"The file has modifications. Save before proceeding?",
fileId);

switch (result) {
case JOptionPane.YES_OPTION:
if (!onSave(fileId)) {
/* Saving was aborted or failed, so we don't proceed */
return false;
}
}
}

```

Coverage

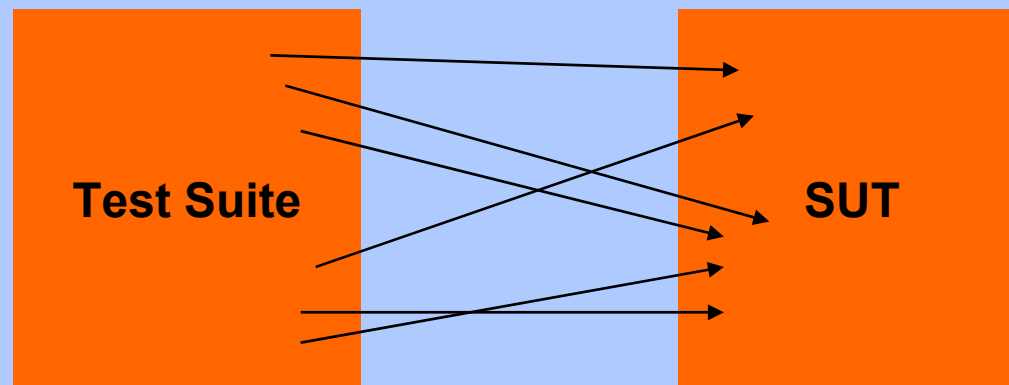
Show methods with Statement Coverage <= 90.5 %

Name	Statement	Branch	Loop	Strict Condition
AppController	64.2 %	53.2 %	33.3 %	11.1 %
AppFile	98.0 %	66.7 %	46.7 %	50.0 %

# Background and Motivations:

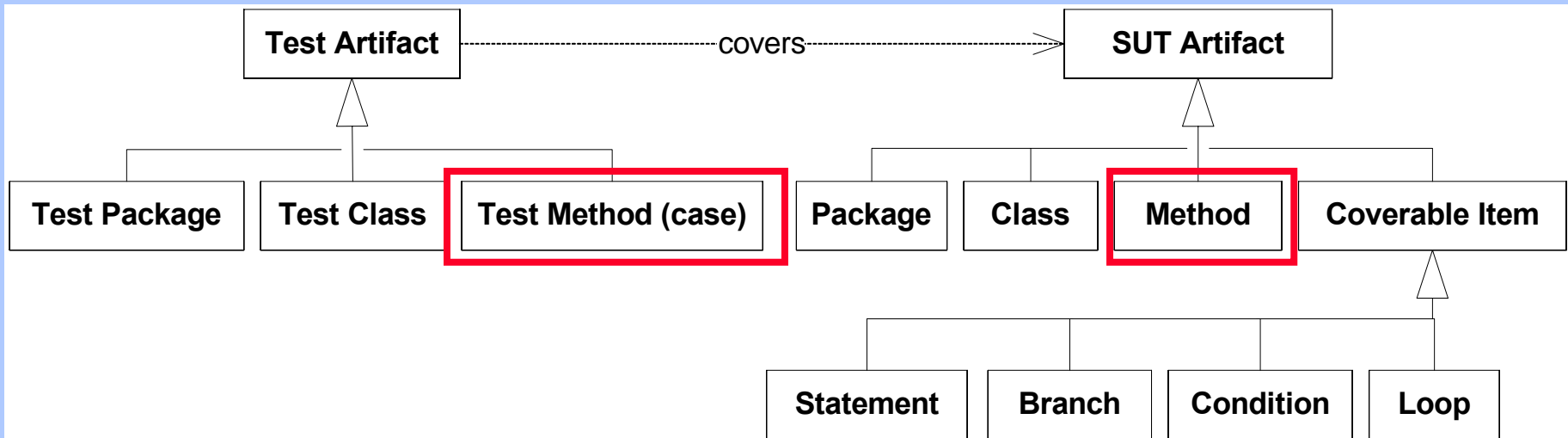
## However... (The need for Test Visualization)

- However with increasing size and complexity of code bases of both systems under test and also their automated test suites (e.g., based on JUnit)
- there is a need for visualization techniques to enable testers to analyze code coverage in “higher” levels of abstraction and in holistic manners
- e.g., which packages of the SUT are covered by a specific set of test cases? Two domains...

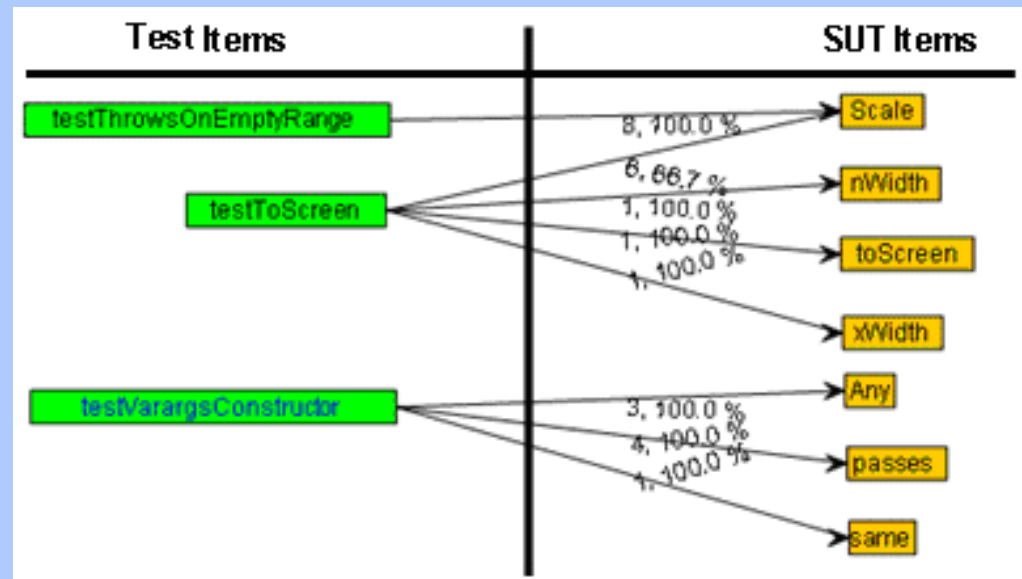


# Background and Motivations:

We have developed a tool to do that (an Eclipse plug-in)



- TeCReVis: A Tool for Test Coverage and Test Redundancy Visualization



# Talk Outline

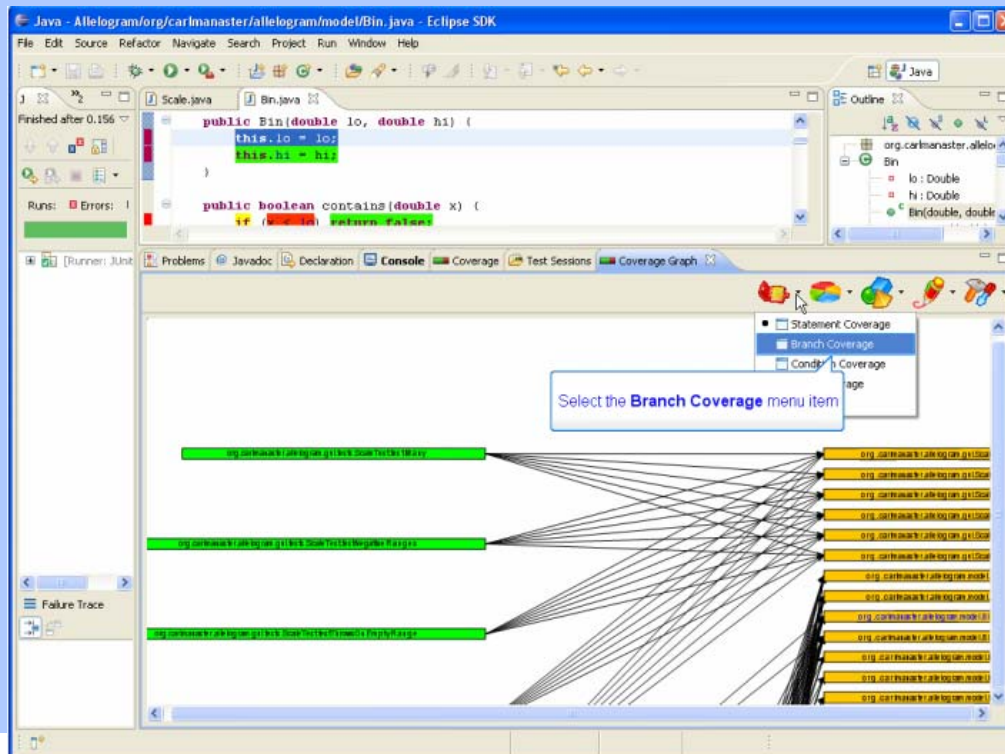
- Background and Motivations
- **Empirical Study - Goal**
- Research Question
- Empirical Study - Setup
- Object of the Study
- Empirical Study - Execution
- Results
- Lessons Learned and Future Works
- Q/A

# Empirical Study - Goal

- We wanted to conduct an Empirical Evaluation to study benefits of visual versus textual test coverage information
- and to assess the usability, effectiveness and usefulness of our tool in unit testing and test maintenance tasks
- The goal (using the GQM template):
- To analyze the benefits of *test coverage visualization*, for the purpose of *evaluating its effectiveness on fault localization* from the point of view of *project managers and software testers* in the context of *software maintenance*.

# Research Question

- Does the TeCReVis tool help human testers on average to localize faults more efficiently compared to the use of conventional code-coverage tools (which show only textual and progress-bar like coverage information)?



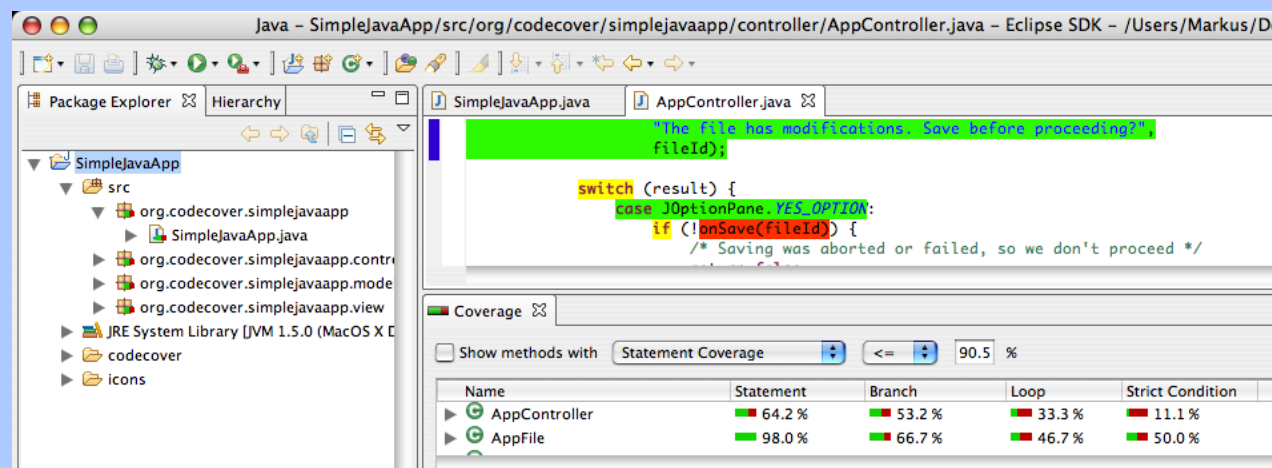


# Talk Outline

- Background and Motivations
- Empirical Study - Goal
- Research Question
- **Empirical Study - Setup**
- Object of the Study
- Empirical Study - Execution
- Results
- Lessons Learned and Future Works
- Q/A

# Empirical Study - Setup

- **Subjects: Eight graduate students (studying at the University of Calgary) in the field of software engineering**
- **The eight participants were divided into two groups**
- **TeCReVis was available only for the experimental group**
- **while the control group used the CodeCover coverage tool**



# Empirical Study - Setup

- In grouping the participants, we utilized rigorous methods as defined by empirical software engineering experts
- e.g., random assignment and careful blocking
- We did our best to make sure that the accumulative testing knowledge and experience of both groups were almost equal
- *Hypothesis (H1):* TeCReVis helps human testers on average to localize faults more efficiently.
- *Null Hypothesis (H0):* TeCReVis does not assist human testers with fault localization.

# A Metric to measure Fault Localization Efficiency

$$FLE(d) = \sum_{i=1}^n \frac{1}{t_i}$$

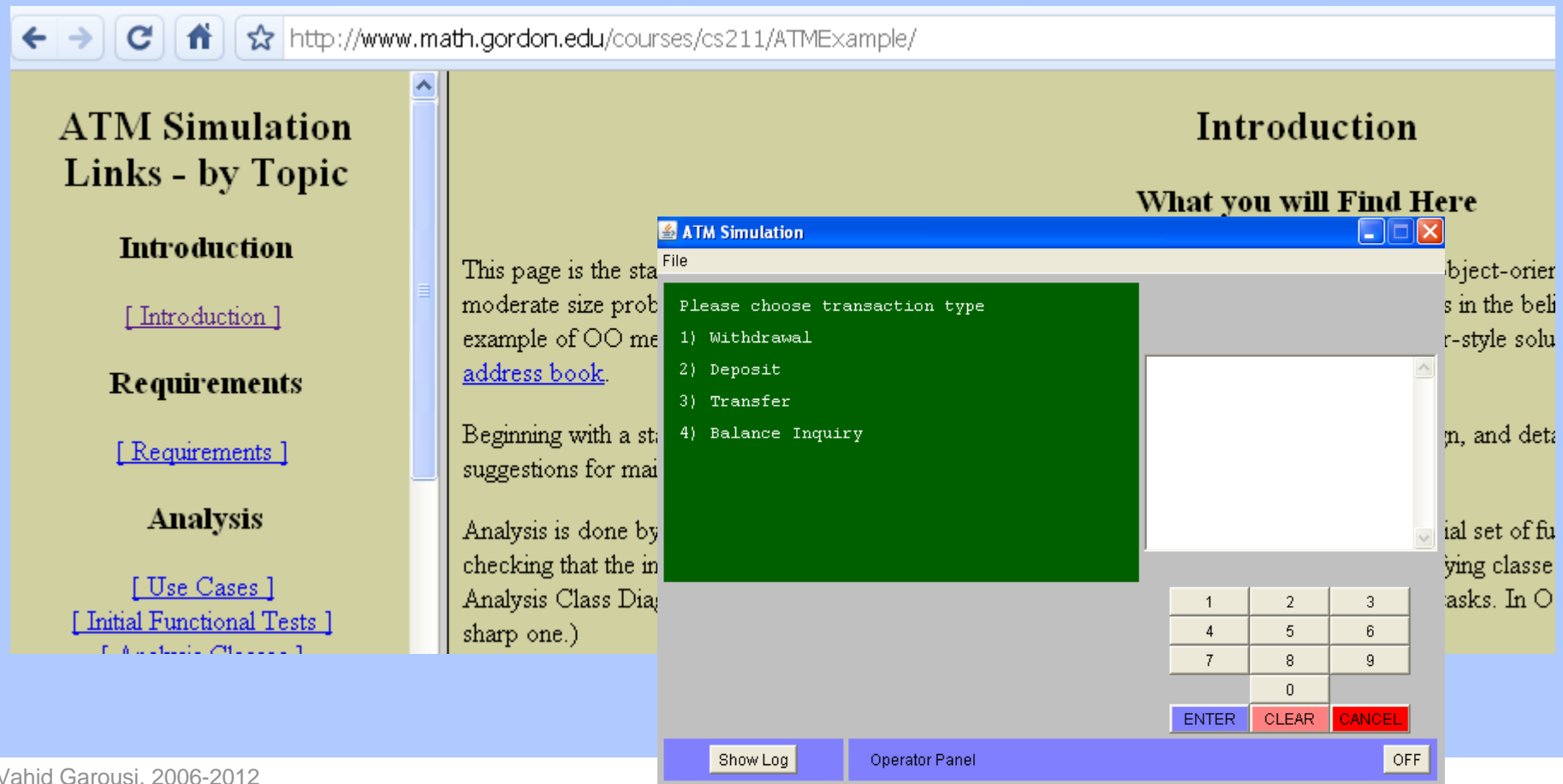
- $d$  is a human debugger and  $t_i$  is the amount of time that he/she has spent to locate the  $i$ -th fault.
- More time spent would result in less efficiency.

# Talk Outline

- Background and Motivations
- Empirical Study - Goal
- Research Question
- Empirical Study - Setup
- **Object of the Study**
- Empirical Study - Execution
- Results
- Lessons Learned and Future Works
- Q/A

# Object of the Study

- An open-source ATM machine simulation software
- 2,541 Java LOC



http://www.math.gordon.edu/courses/cs211/ATMExample/

## ATM Simulation Links - by Topic

### Introduction

[\[ Introduction \]](#)

### Requirements

[\[ Requirements \]](#)

### Analysis

[\[ Use Cases \]](#)  
[\[ Initial Functional Tests \]](#)  
[\[ Analysis Classes \]](#)

## Introduction

### What you will Find Here

This page is the start of a moderate size problem example of OO methodology. [address book.](#)

Beginning with a structured suggestions for main

Analysis is done by checking that the in Analysis Class Diagram sharp one.)

object-oriented s in the behavior-style solution, and details of a small set of functional classes asks. In O

**ATM Simulation**

File

Please choose transaction type

- 1) Withdrawal
- 2) Deposit
- 3) Transfer
- 4) Balance Inquiry

1	2	3
4	5	6
7	8	9
	0	

ENTER CLEAR CANCEL

Show Log Operator Panel OFF

# Object of the Study

- **To perform the fault localization process, we slightly revised this system by injecting into it three (realistic) faults.**
- **Since there was no unit test suite provided with the ATM implementation online, we created a test suite (containing 23 JUnit test methods) for version 1 of this system.**
- **This test suite was constructed to achieve full path coverage on the SUT's UML state-chart diagram.**
- **For replicability purposes, all of the developed JUnit test suite and the system's UML design models are available online. (see the URL in the paper)**

# Empirical Study - Execution

- **Participants were asked to find and locate three injected faults in the ATM system.**
- **Participants were asked to report the time of locating each fault, which were analyzed later by the authors to measure fault localization efficiency.**





# Talk Outline

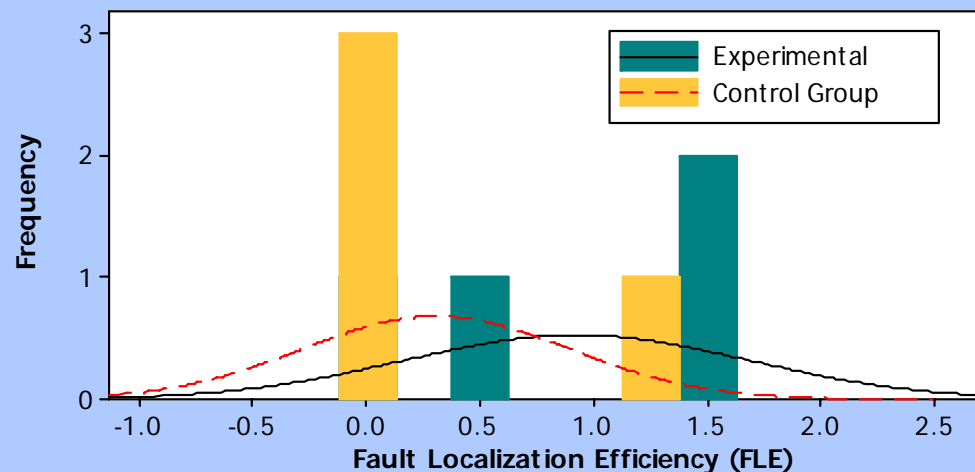
- Background and Motivations
- Empirical Study - Goal
- Research Question
- Empirical Study - Setup
- Object of the Study
- Empirical Study - Execution
- **Results**
- Lessons Learned and Future Works
- Q/A

# Results of the Experiment

Group	Participant	Time of locating Fault 1	Time of locating Fault 2	Time of locating Fault 3	Efficiency (FLE)
		All time values are in minutes.			
Experimental Group (used TeCReVis)	P1	20	2	1	1.55
	P2	24	*	*	0.04
	P3	18	1	2	1.55
	P4	23	2	*	0.54
Control Group (used CodeCover)	P5	*	*	*	0
	P6	27	*	*	0.03
	P7	22	7	1	1.18
	P8	*	*	*	0

# Results of the Experiment

- t-test was applied.
  - Two types of experiment errors ( $\alpha$  and  $\beta$ ) were as follows:
  - $\alpha=0.12$  and  $\beta=0.47$  (pass if only  $\alpha<0.05$ )
  - *Reminder:  $\alpha = P(H_0 \text{ is rejected} \mid H_0 \text{ is true})$  and  $\beta = P(H_0 \text{ is accepted} \mid H_0 \text{ is false})$ .*
- Null hypothesis ( $H_0$ ) cannot be rejected
- It is possible to say with confidence that TeCReVis helps human testers on average to localize faults more efficiently.



# Lessons Learned and Future Works

- We believe that, although we had tutorial part in our experiment first, learning curve in limited time of performing fault localization task in the experiment has affected our results.
- In other words, learning curve caused less effectiveness of using TeCReVis in localizing faults in limited time.
- All of the participants' answers were supportive of the usefulness of TeCReVis for fault localization.
- For instance, a participant of the experiment group said: *“I feel that, in large systems, this graph-based visualization can be very useful”*.
- Repeating the experiment with more subjects and more control.

# Talk Outline

- Background and Motivations
- Empirical Study - Goal
- Research Question
- Empirical Study - Setup
- Object of the Study
- Empirical Study - Execution
- Results
- Lessons Learned and Future Works
- **Q/A**