# Does Research Support the Effectiveness of Test Driven Development?

*Filippo Ricca*

*DISI, Università di Genova*

# Agenda

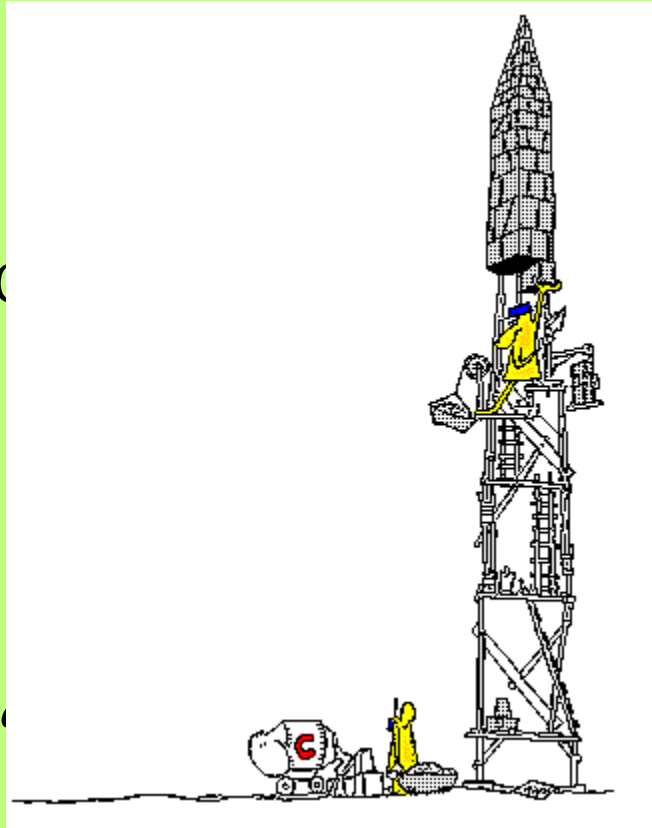
- What is TDD?
- A little bit of History!
- TDD vs. TLD (Test Last Development)
- (Supposed) Benefits and drawbacks of TDD
- TDD in action with JUnit
  - Current/Bank account
- Empirical evidence on TDD

# What is TDD?

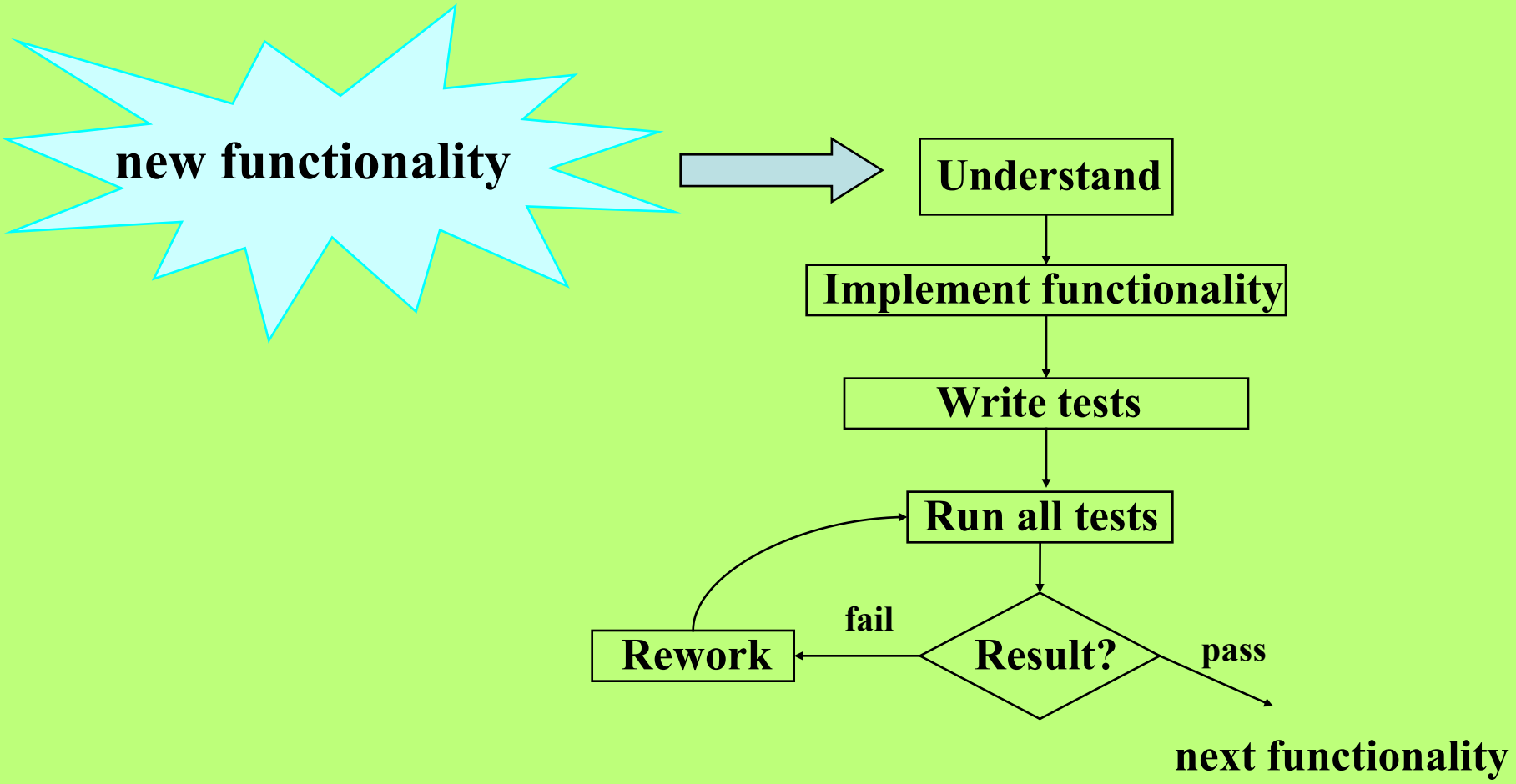
- **TDD is not**
  - a testing technique
  - all that new
    - used in the 1960's
      - NASA project
- **TDD is**
  - (micro) development and design technique
    - Test first
    - Refactoring
  - cornerstone of XP

# A little bit of History

- **In the early 1960s**
  - TDD used in NASA's Project Mercury
- **In 1996**
  - K. Beck "rediscovered" TDD during the C project
    - C3 = payroll system
    - C3 was never fully implemented!
    - C3 was eventually shut down!
- **In 1999**
  - *K. Beck, Extreme Programming Explained, Addison Wesley*
- **In 2003**
  - *K. Beck, Test-Driven Development by Example, Addison Wesley*

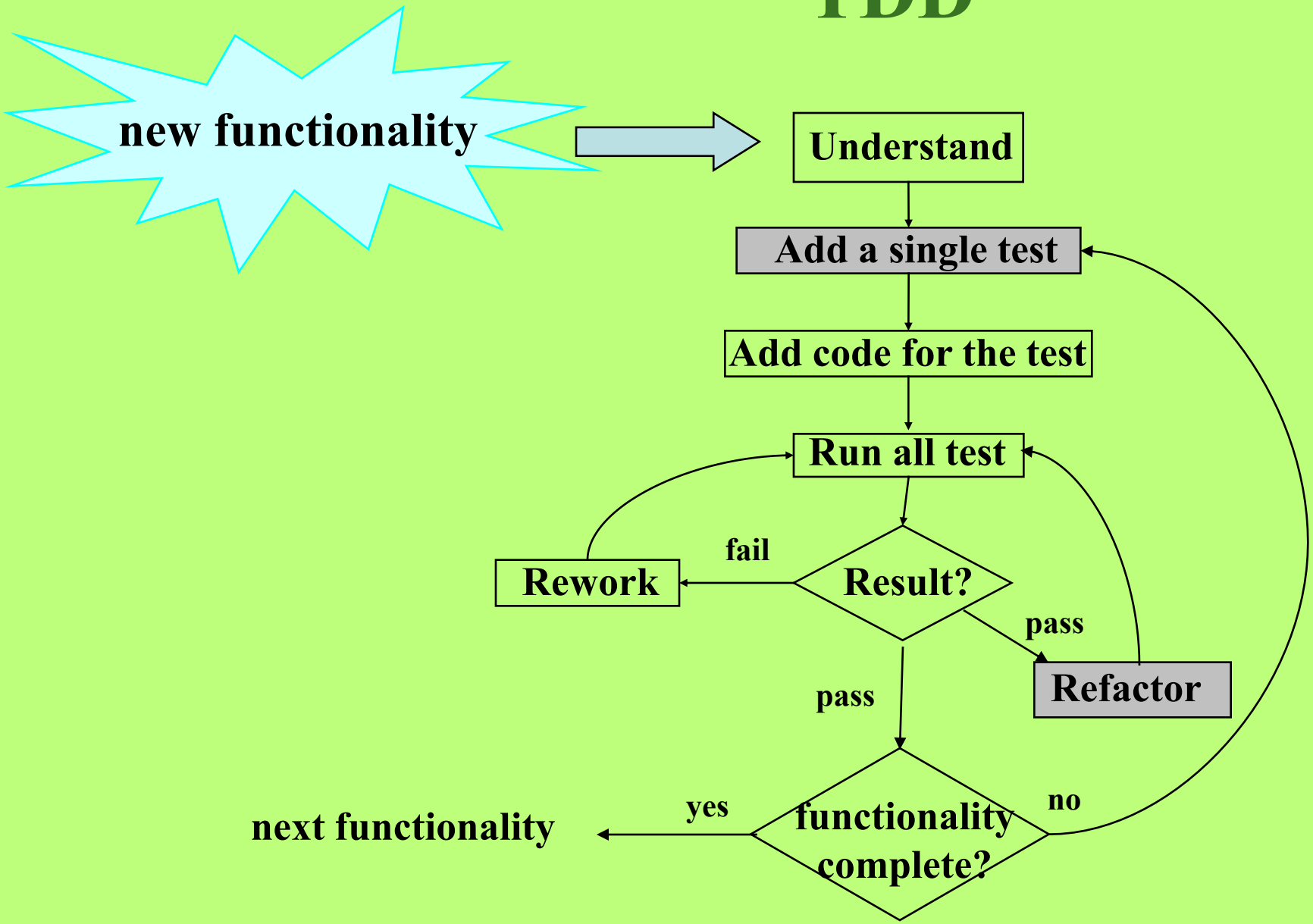


# Test last development




"new functionality" = class with methods

# TDD


new functionality
Understand
Add a single test
Add code for the test
Run all test
Result?
fail
Rework
pass
Refactor
pass
functionality complete?
yes
next functionality
no

# (Supposed) Benefits of TDD

- Code more cohesive and less coupled
- Simplify the design
- Improve software quality
  - in terms of decreased fault rates
- Improve testability
- Augment the test coverage
- Improve documentation
- Simplify maintenance/restructuring
  - Regression suite

Is it true?

If yes, direct consequence or indirect of TDD?

# (Supposed) Drawbacks of TDD

- Limited by “skill&experience” of developers
  - No upfront design
- Some codes are hard to test with TDD
  - GUIs
  - Effort to write mock objects
- Effort required is high
  - in terms of time
- “Tight Coupling” with automatic testing tools
  - JUnit

# JUnit

- JUnit is a **unit test environment** for Java
  - Writing test cases
  - Executing test cases
    - Pass/fail? (red/green bar)
- Developed by *Eric Gamma* and *Kent Beck*
- **Framework** providing all the tools for testing
  - Framework=set of Java classes and conventions to use them
- Integrated into **Eclipse**

# Stack example: main

JUnit tests "substitute the use of **main**"

```
public class Stack {
   …
      public static void main (String[] args) {
         Stack aStack = new Stack();
         if (!aStack.isEmpty()) {
               System.out.println ("Stack should be empty!");
         }
         aStack.push(10);
         aStack.push(-4);
         System.out.println("Last element:" + aStack.pop());
         System.out.println("First element: " + aStack.pop());
      }
}
```

```
public class Stack {
   isEmpty()
   pop()
   push(...)
}
```

-4

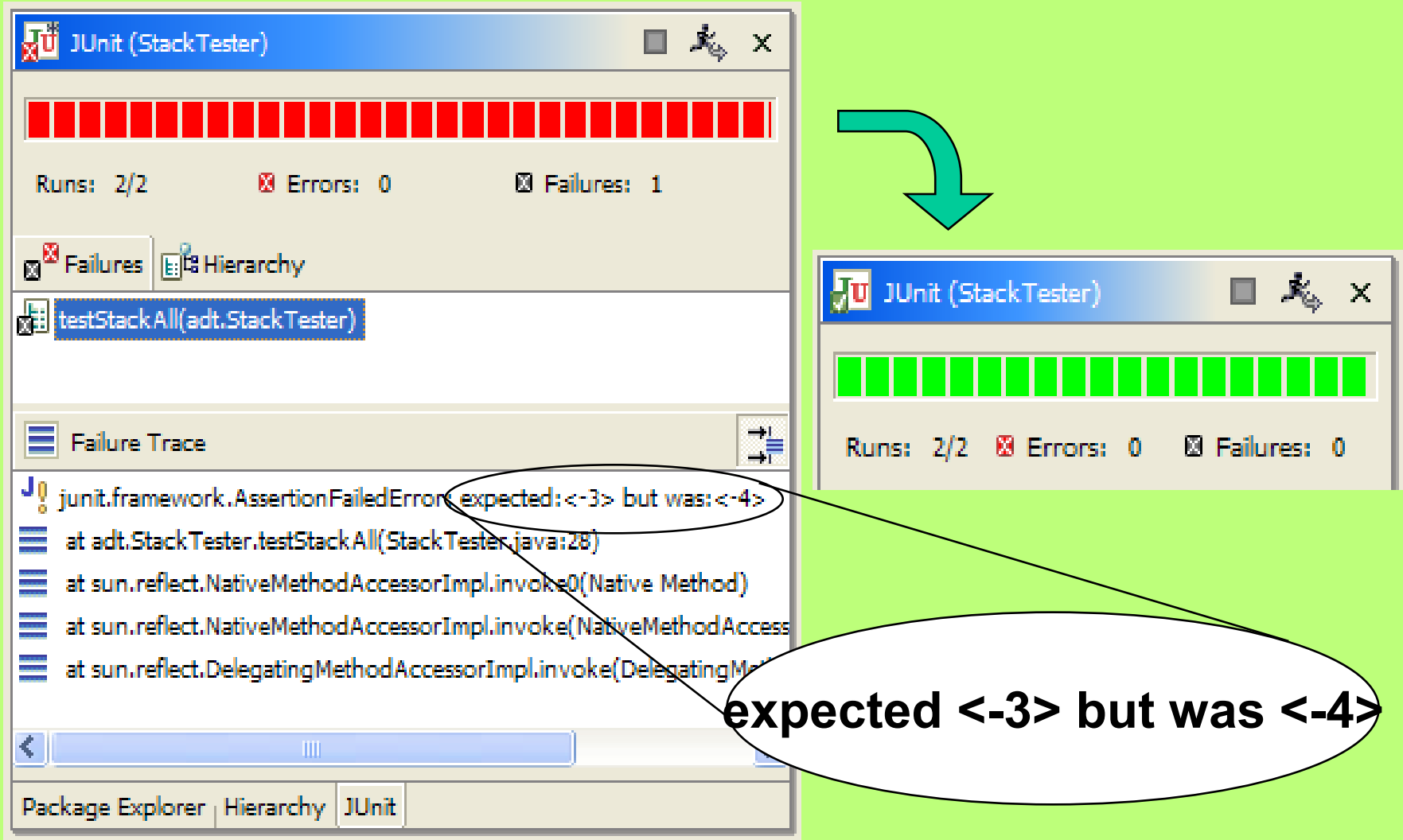
10

# Stack example: JUnit

```
public class Stack {
   isEmpty()
   pop()
   push(...)
}
```

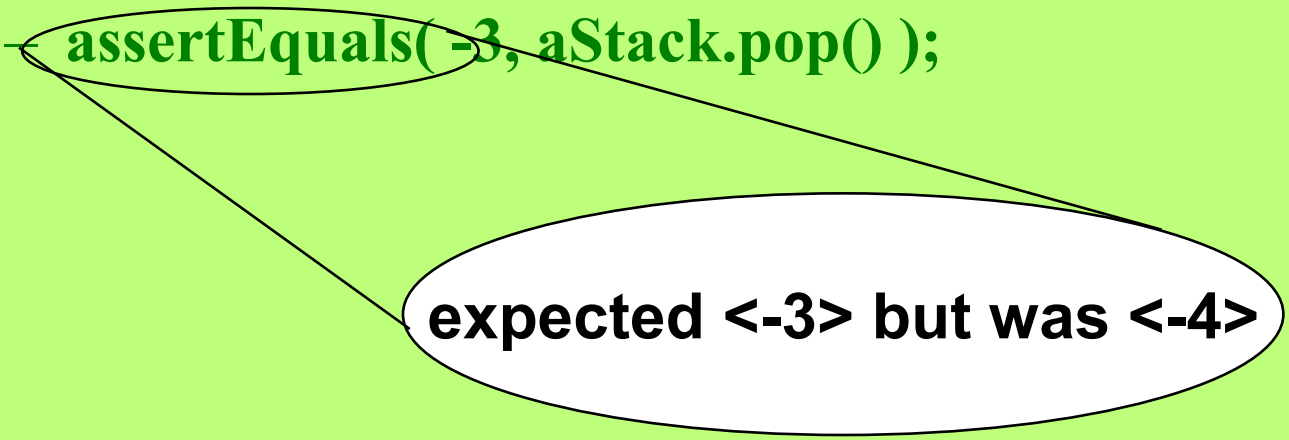
```
import junit.framework.TestCase;

public class StackTester extends TestCase {

 …

  public void testStack() {
    Stack aStack = new Stack();
    assertTrue("Stack should be empty!", aStack.isEmpty());
    aStack.push(10);
    assertTrue("Stack should not be empty!",!aStack.isEmpty());
    aStack.push(-4);
    assertEquals(-4, aStack.pop());
    assertEquals(10, aStack.pop());
  }
}
```

# Red / Green Bar



# Failures vs. Errors

- **Failure**: assertion not satisfied
  - **assertEquals( -3, aStack.pop() );**



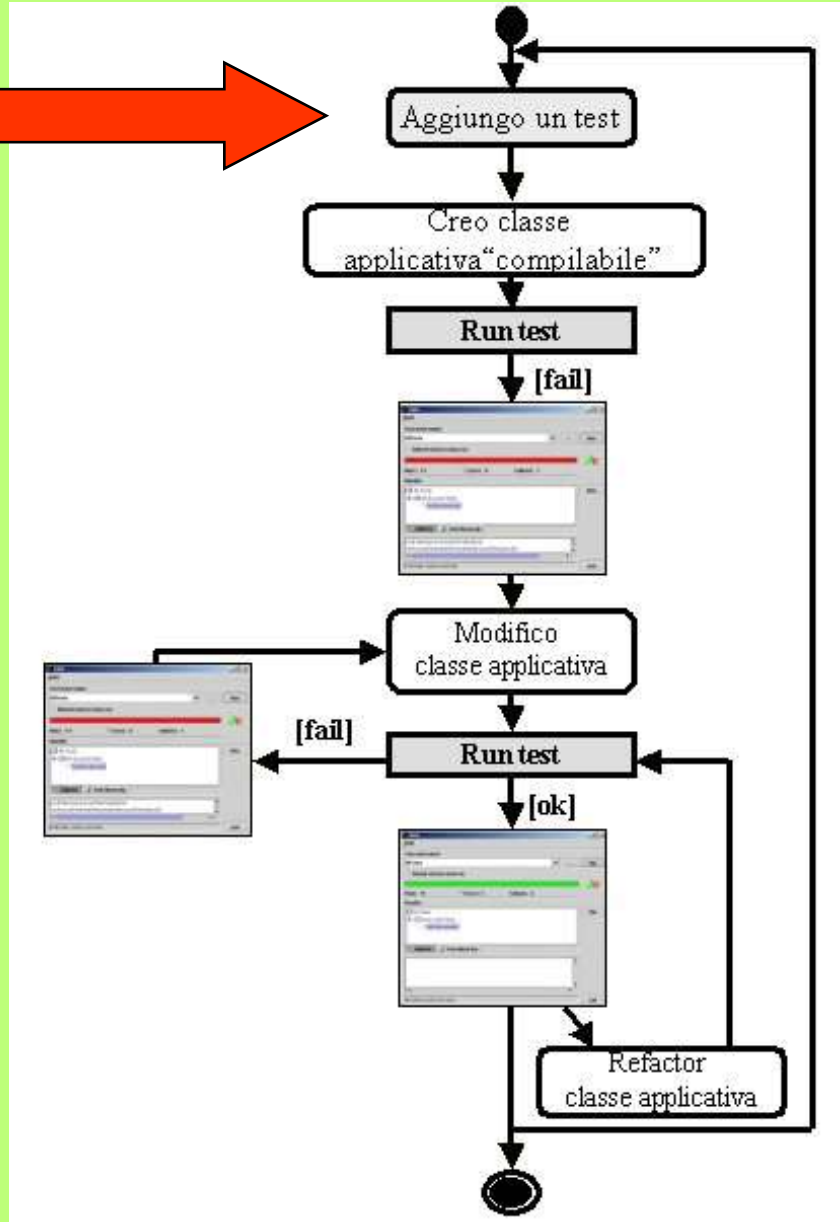

- **Error**: run time error
  - **assertEquals( -3, saldo.getSaldo() );**

java.lang.ArrayIndexOutOfBoundsException …

# TDD in practice …

- **New Functionality:**
  - Create a class current account (bank account)
    - deposit (*deposito*)
    - withdraw (*prelievo*)
    - settlement (*saldo*)

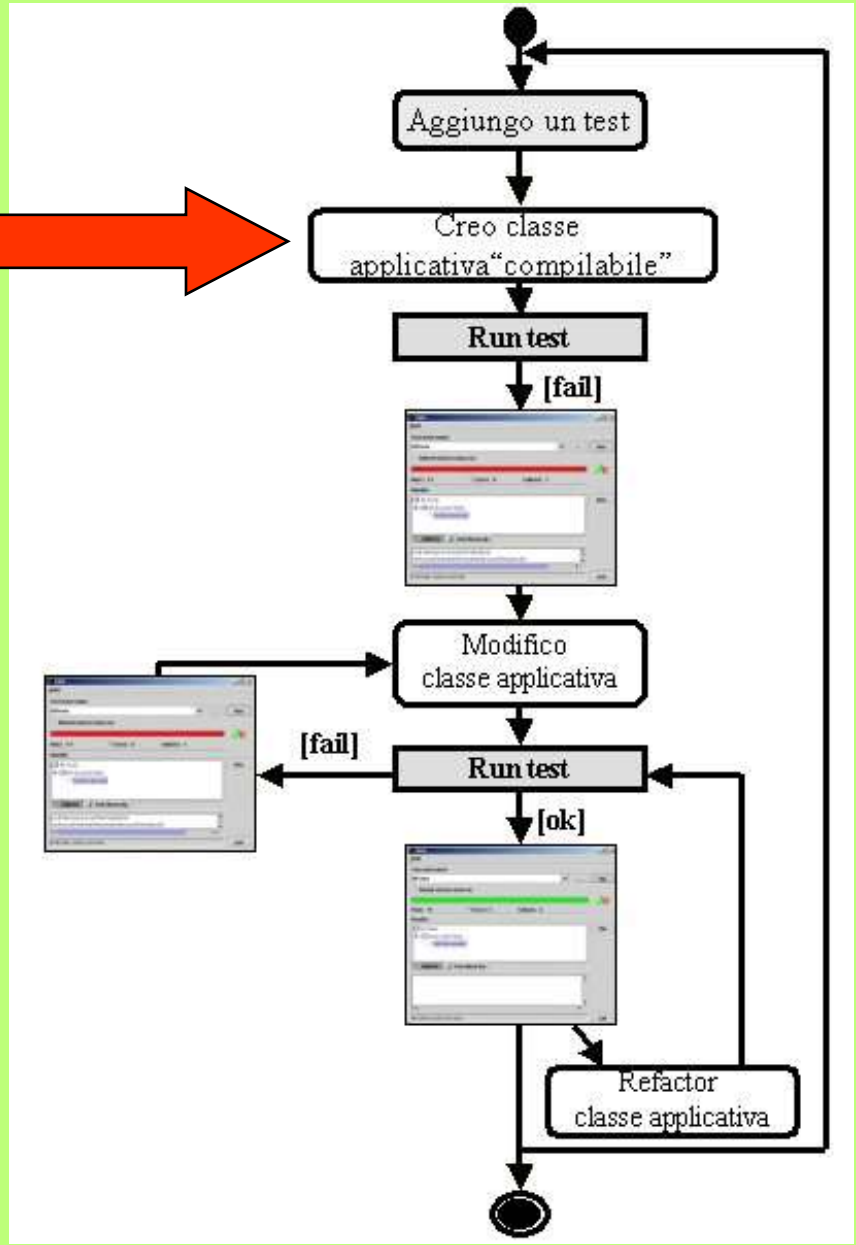
**Example:**


Aggiungo un test
Creo classe applicativa"compilabile"
Run test
[fail]
Modifico classe applicativa
[fail]
Run test
[ok]
Refactor classe applicativa

# Add a single test

```
import junit.framework.TestCase;

public class Test_account extends TestCase {

    public void testSettlement() {
        CAccount c = new CAccount();
        c.deposit(12);
        c.draw(-8);
        c.deposit(10);
        assertTrue(c.settlement() == 14);
    }
}
```

*"assertion"* 

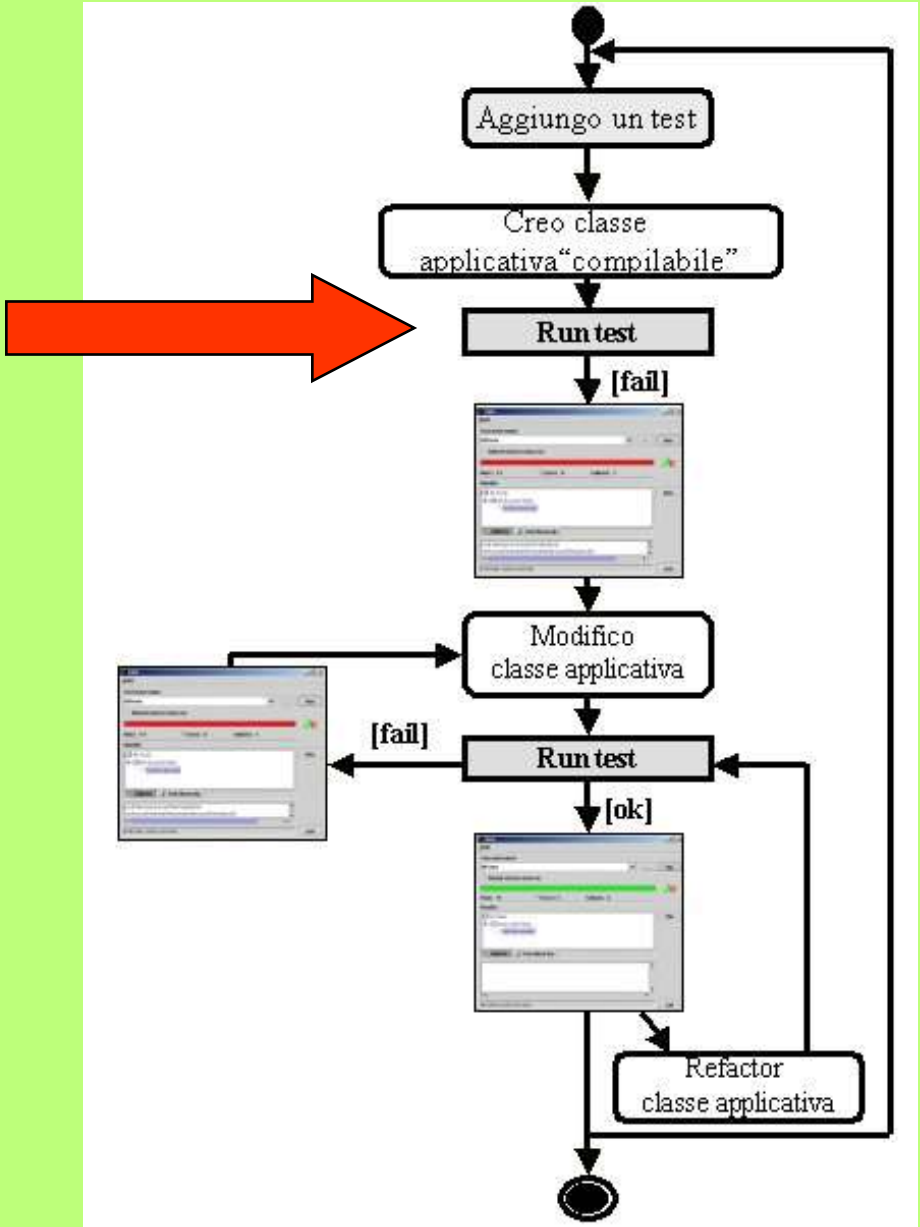

Aggiungo un test
Creo classe applicativa"compilabile"
Run test
[fail]
Modifico classe applicativa
[fail]
Run test
[ok]
Refactor classe applicativa

# Add the skeleton of class

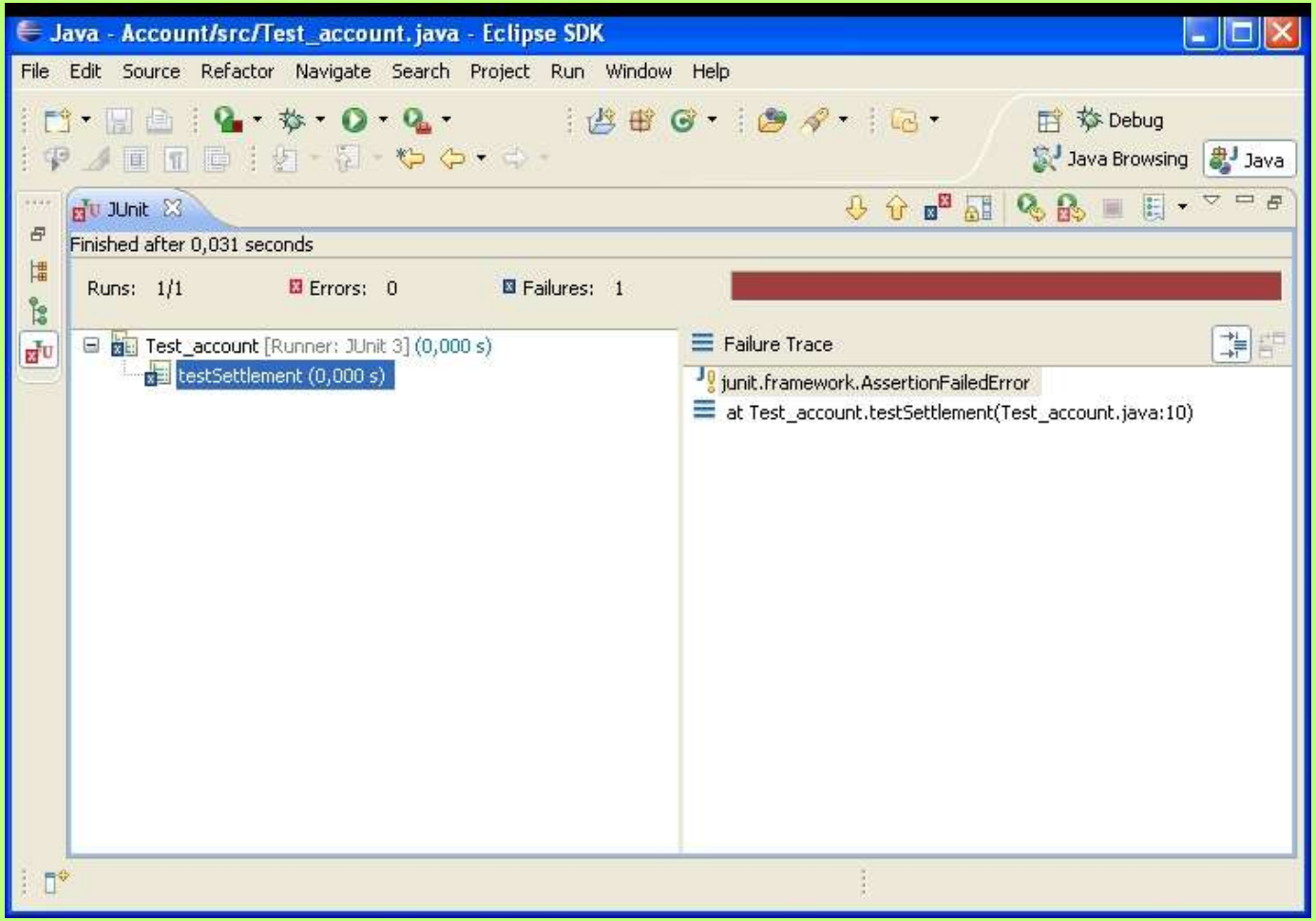
```
public class CAccount {

  public void deposit(int i) {
    // Auto-generated method stub
  }
  public void draw(int i) {
    // Auto-generated method stub
  }
  public int settlement() {
    // Auto-generated method stub
  return 0;
  }
}
```

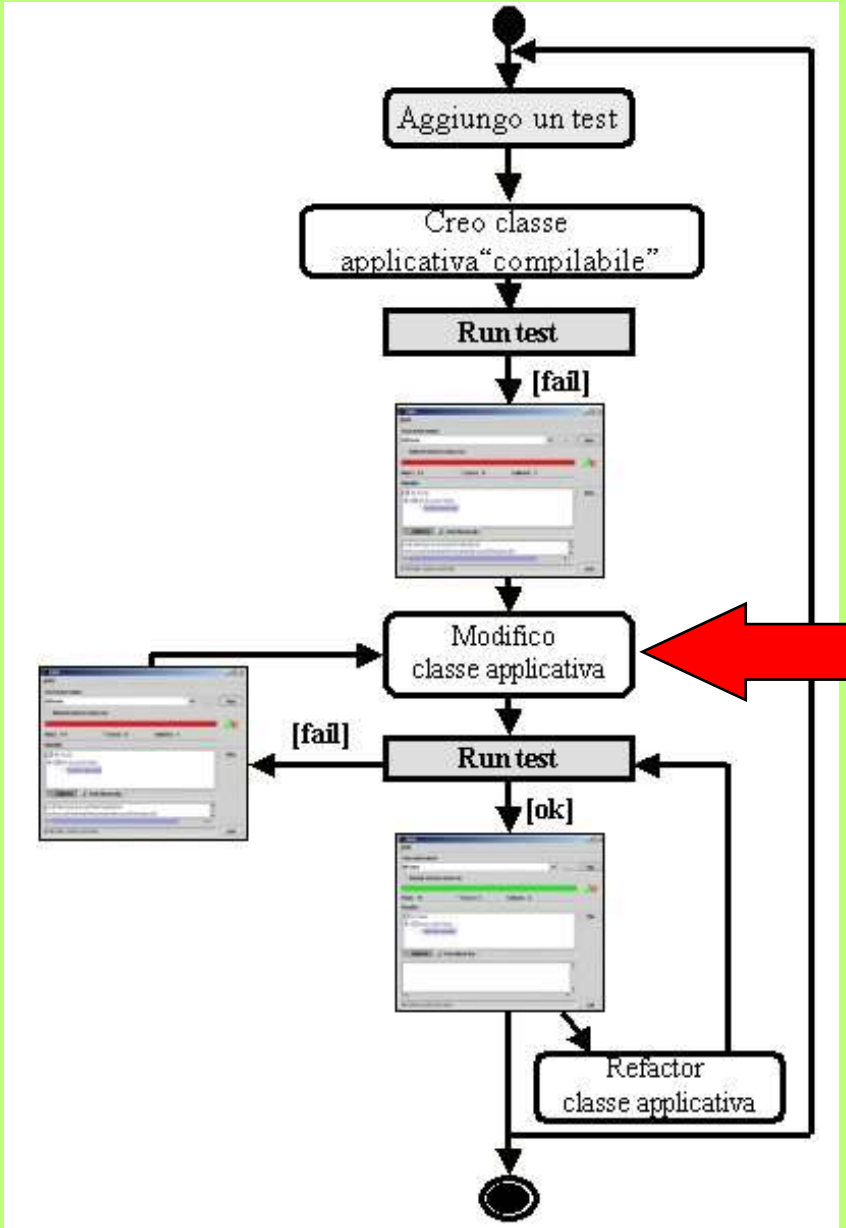
```
import junit.framework.TestCase;

public class Test_account extends TestCase {

    public void testSettlement() {
        CAccount c = new CAccount();
        c.deposit(12);
        c.draw(-8);
        c.deposit(10);
        assertTrue(c.settlement() == 14);
    }
}
```

*"Eclipse auto-completion"*


Aggiungo un test
Creo classe applicativa"compilabile"
Run test
[fail]
Modifico classe applicativa
[fail]
Run test
[ok]
Refactor classe applicativa

# Run JUnit (1)

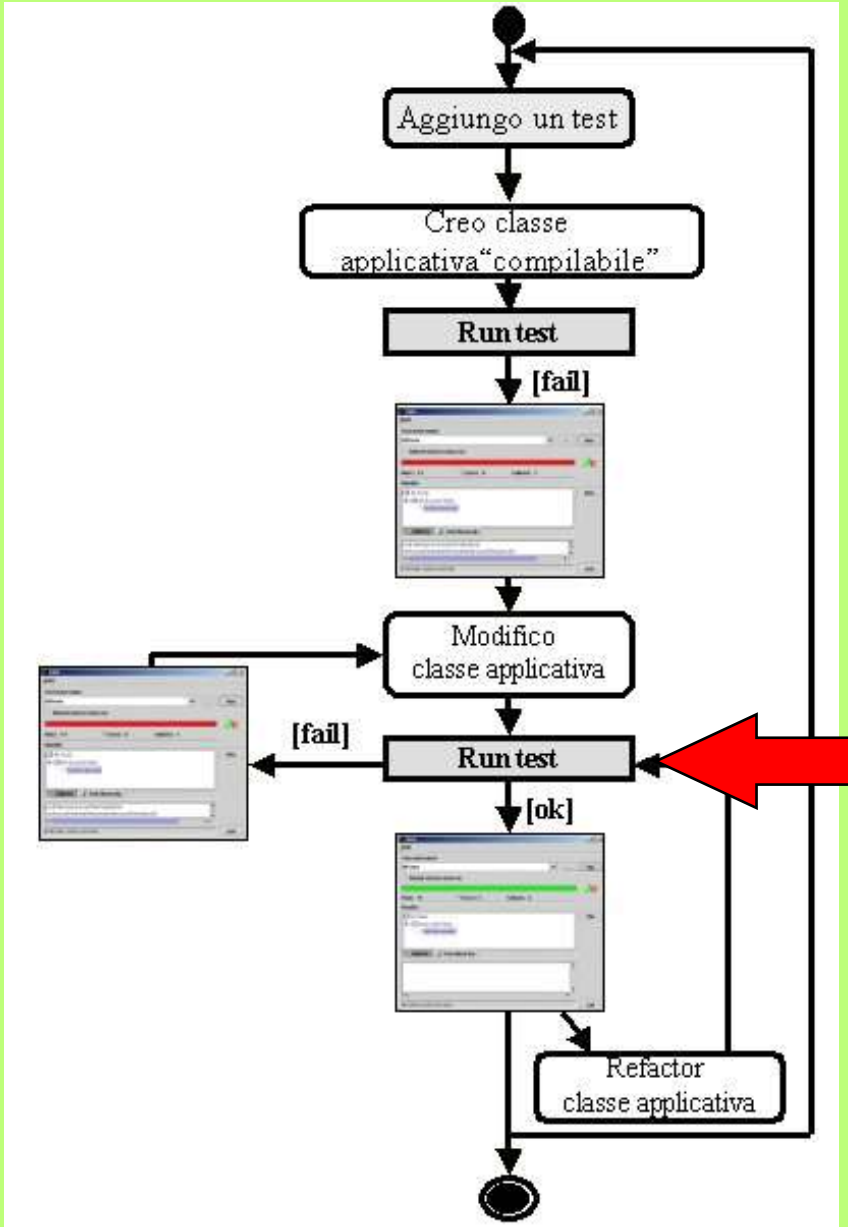

Java - Account/src/Test_account.java - Eclipse SDK
File Edit Source Refactor Navigate Search Project Run Window Help
Debug
Java Browsing
Java
JUnit
Finished after 0,031 seconds
Runs: 1/1
Errors: 0
Failures: 1
Test_account [Runner: JUnit 3] (0,000 s)
testSettlement (0,000 s)
Failure Trace
junit.framework.AssertionFailedError
at Test_account.testSettlement(Test_account.java:10)


Aggiungo un test
Creo classe applicativa"compilabile"
Run test
[fail]
Modifico classe applicativa
[fail]
Run test
[ok]
Refactor classe applicativa

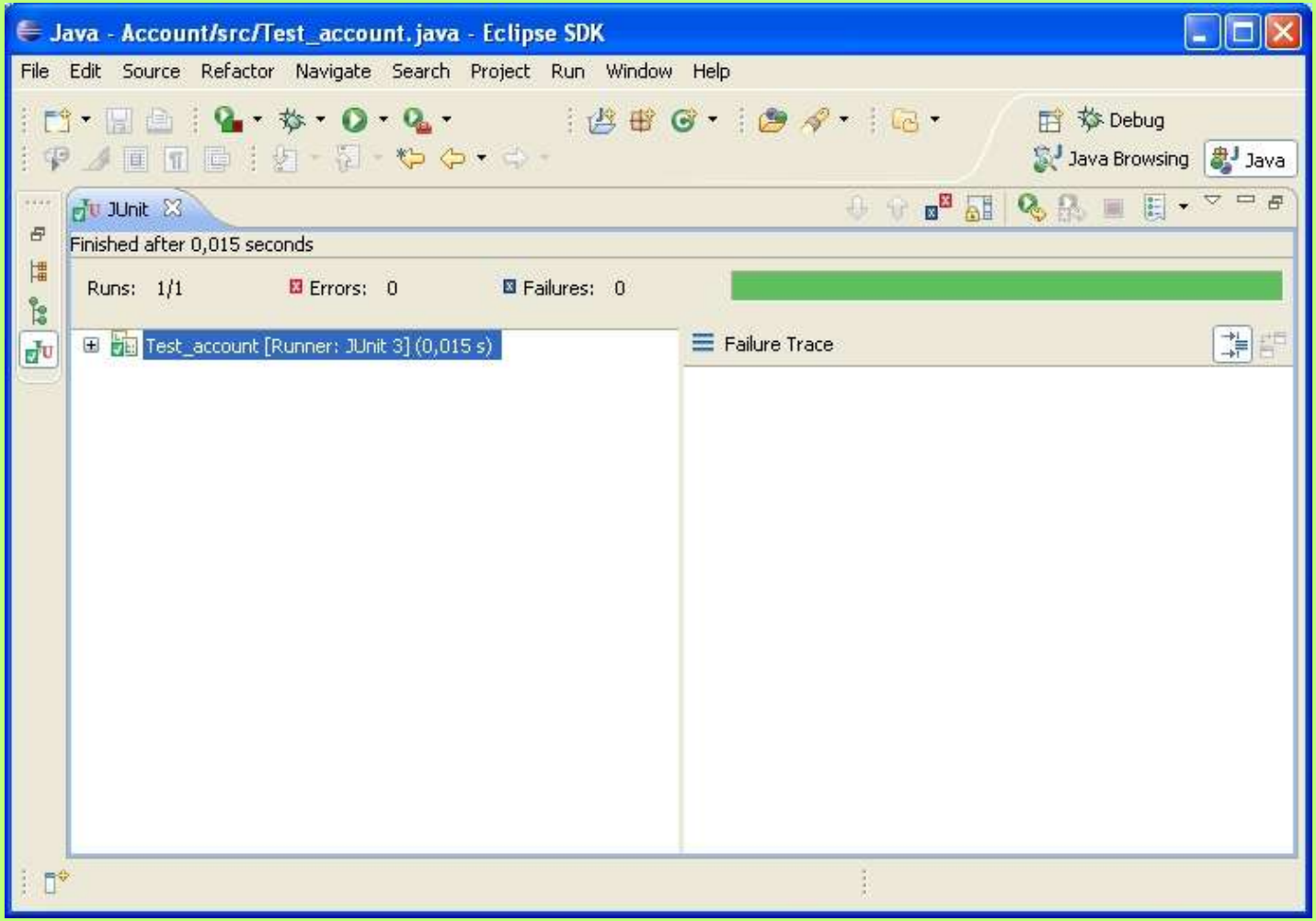
```
public class CAccount {

int account[]; int lastMove;

public CAccount(){
   lastMove=0; account=new int[10];}

public void deposit(int value){
   account[lastMove]=value;
   lastMove++;
}

public void draw(int value) {
   account[lastMove]=value;
   lastMove++;
}

public int settlement() {
  int result = 0;
  for (int i=0; i<account.length; i++) {
      result = result + account[i];
   }
  return result;
 }
}
```

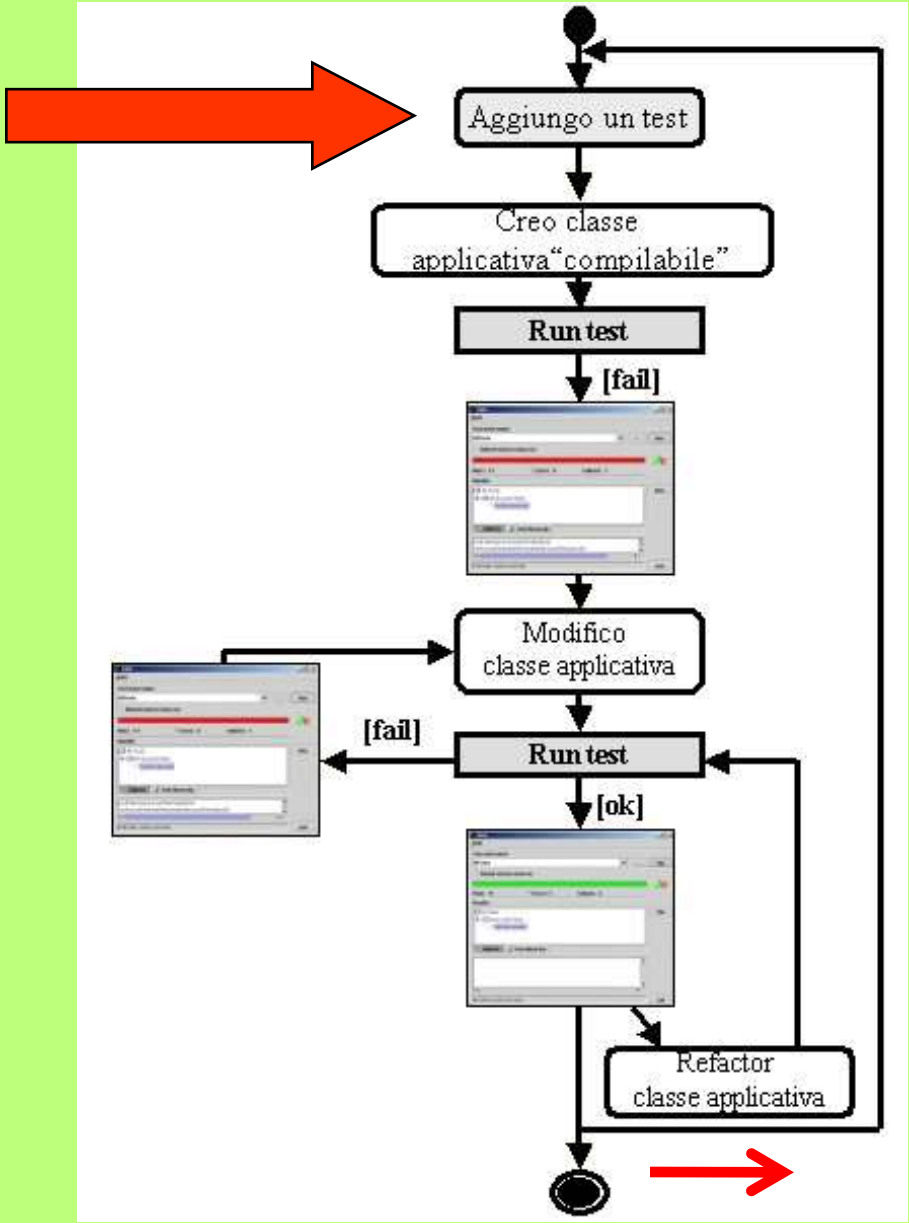
# Rework

```
import junit.framework.TestCase;

public class Test_account extends TestCase {

   public void testSettlement() {
     CAccount c = new CAccount();
      c.deposit(12);
      c.draw(-8);
      c.deposit(10);
      assertTrue(c.settlement() == 14);
   }
}
```


Aggiungo un test
Creo classe applicativa"compilabile"
Run test
[fail]
Modifico classe applicativa
[fail]
Run test
[ok]
Refactor classe applicativa

# Run JUnit (2)

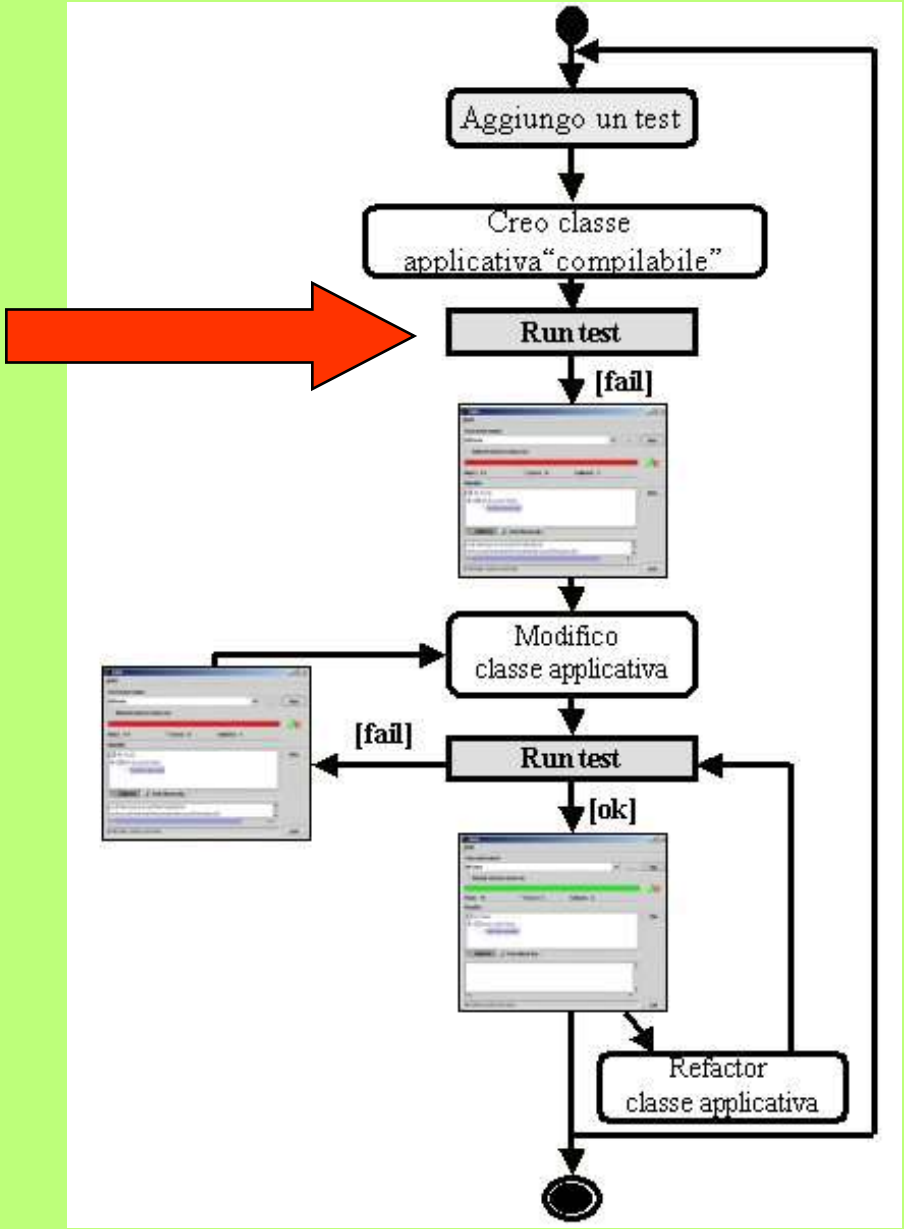

Java - Account/src/Test_account.java - Eclipse SDK
File Edit Source Refactor Navigate Search Project Run Window Help
Debug
Java Browsing
Java
JUnit
Finished after 0,015 seconds
Runs: 1/1
Errors: 0
Failures: 0
Test_account [Runner: JUnit 3] (0,015 s)
Failure Trace


Aggiungo un test
Creo classe applicativa"compilabile"
Run test
[fail]
Modifico classe applicativa
[fail]
Run test
[ok]
Refactor classe applicativa

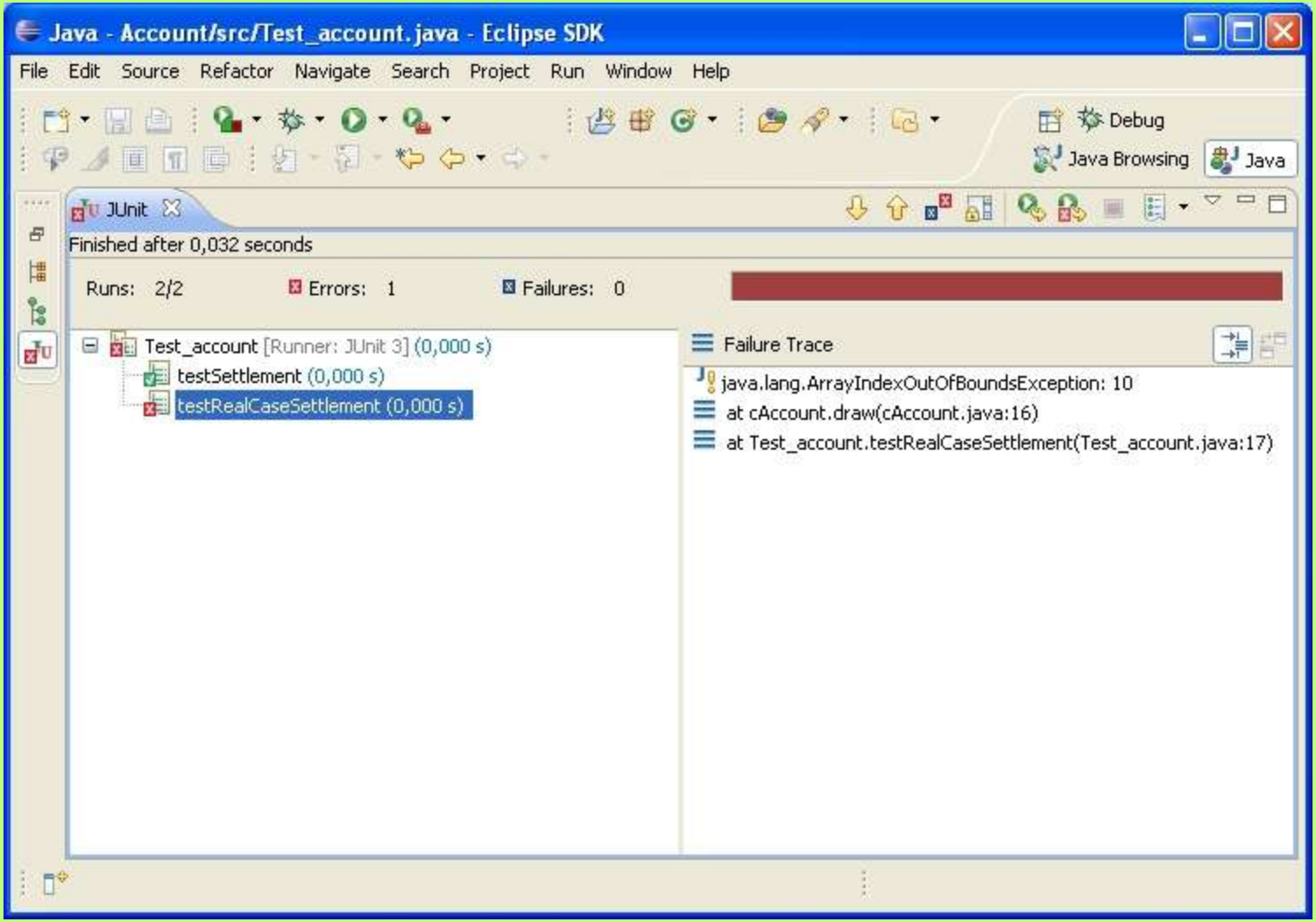
```
public class CAccount {

int account[]; int lastMove;

public CAccount(){
   lastMove=0; account=new int[10];}

public void deposit(int value){
   account[lastMove]=value;
   lastMove++;
}

public void draw(int value) {
   account[lastMove]=value;
   lastMove++;
}

public int settlement() {
  int result = 0;
  for (int i=0; i<account.length; i++) {
      result = result + account[i];
   }
  return result;
 }
}
```

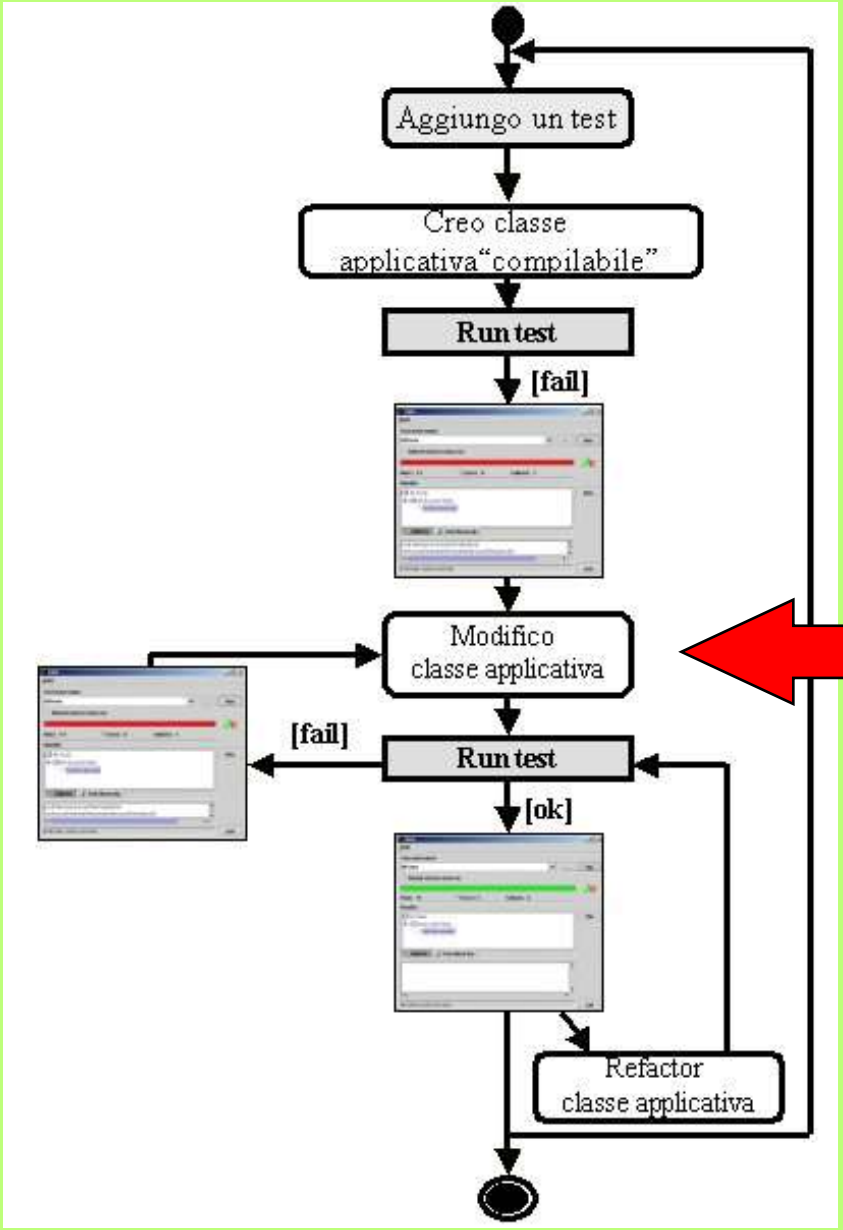
# Add a new test

```
public class Test_account extends TestCase {

public void testSettlement() {
     …..
}

public void testRealCaseSettlement() {
  CAccount c = new CAccount();
  for (int i=0; i <10 ; i++)
        c.deposit(1);
  c.draw(-10);
  assertTrue(c.settlement() == 0);
 }
}
```


Aggiungo un test
Creo classe applicativa"compilabile"
Run test
[fail]
Modifico classe applicativa
[fail]
Run test
[ok]
Refactor classe applicativa

# Run JUnit (3)

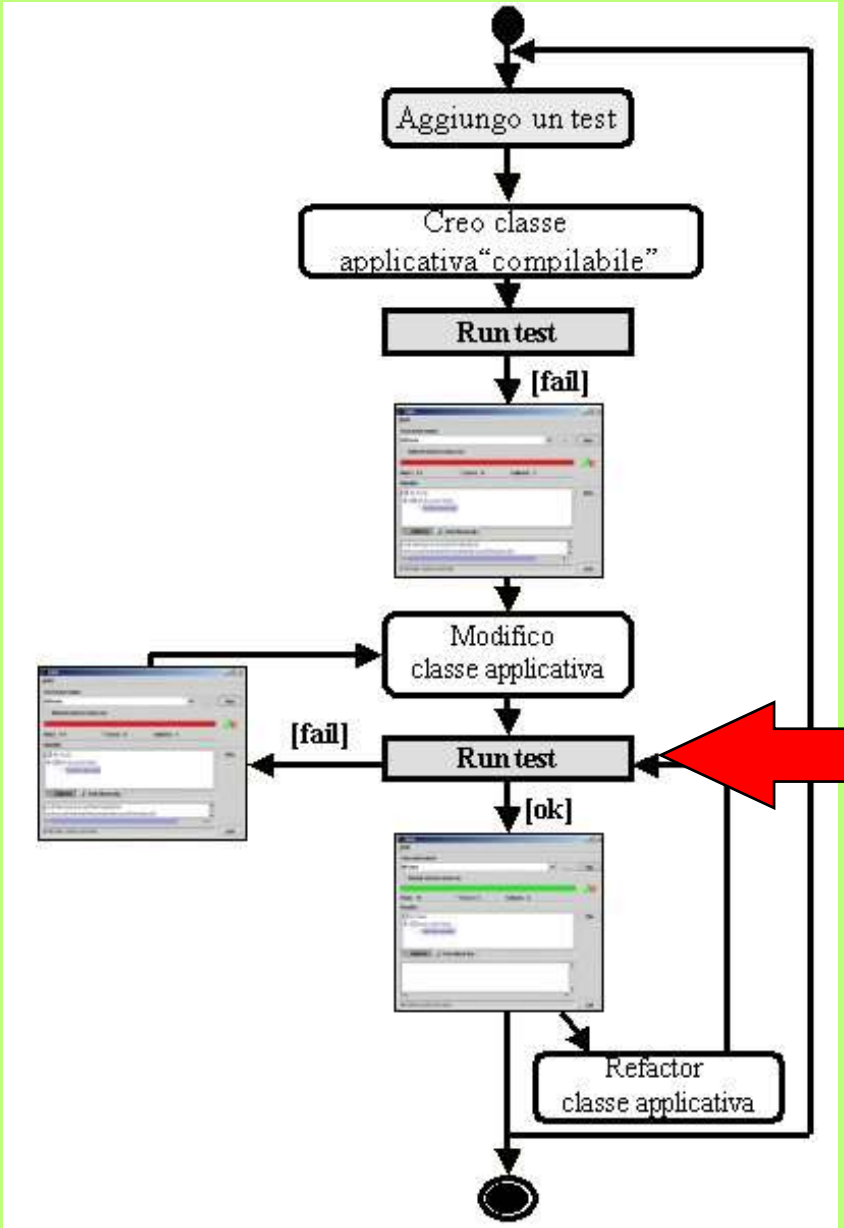

Java - Account/src/Test_account.java - Eclipse SDK
File Edit Source Refactor Navigate Search Project Run Window Help
Debug
Java Browsing
Java
JUnit
Finished after 0,032 seconds
Runs: 2/2
Errors: 1
Failures: 0
Test_account [Runner: JUnit 3] (0,000 s)
testSettlement (0,000 s)
testRealCaseSettlement (0,000 s)
Failure Trace
java.lang.ArrayIndexOutOfBoundsException: 10
at cAccount.draw(cAccount.java:16)
at Test_account.testRealCaseSettlement(Test_account.java:17)


Aggiungo un test
Creo classe applicativa"compilabile"
Run test
[fail]
Modifico classe applicativa
[fail]
Run test
[ok]
Refactor classe applicativa

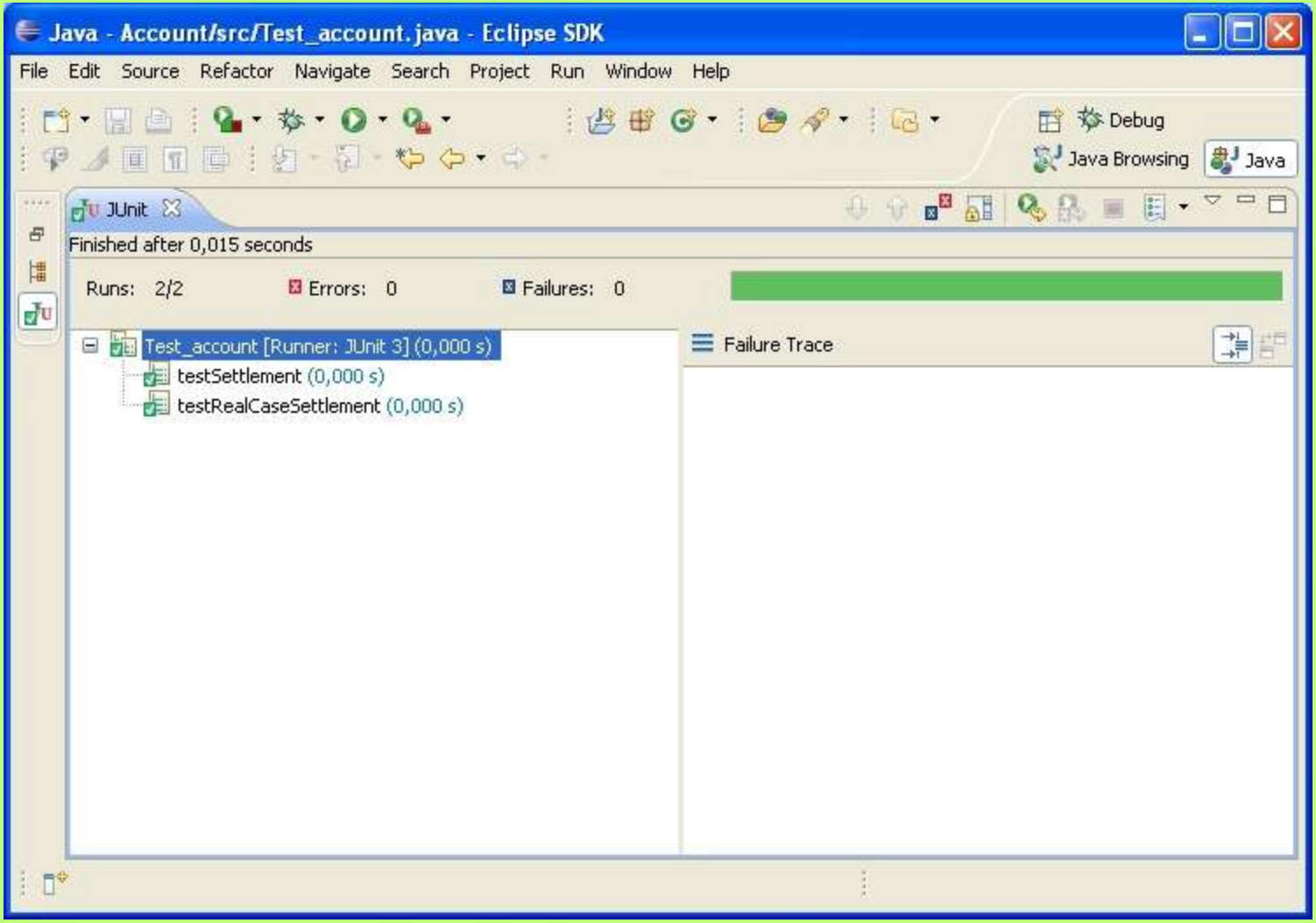
```
public class CAccount {
  List account = new LinkedList();
  public void deposit(int value){
    account.add(new Integer(value));
  }
  public void draw(int value){
    account.add(new Integer(value));
  }
  public int settlement() {
    int result = 0;
    Iterator it = account.iterator();
    while (it.hasNext()) {
      Integer valueI = (Integer)it.next();
      int val = valueI.intValue();
      result = result + val;
    }
    return result;
  }
}
```

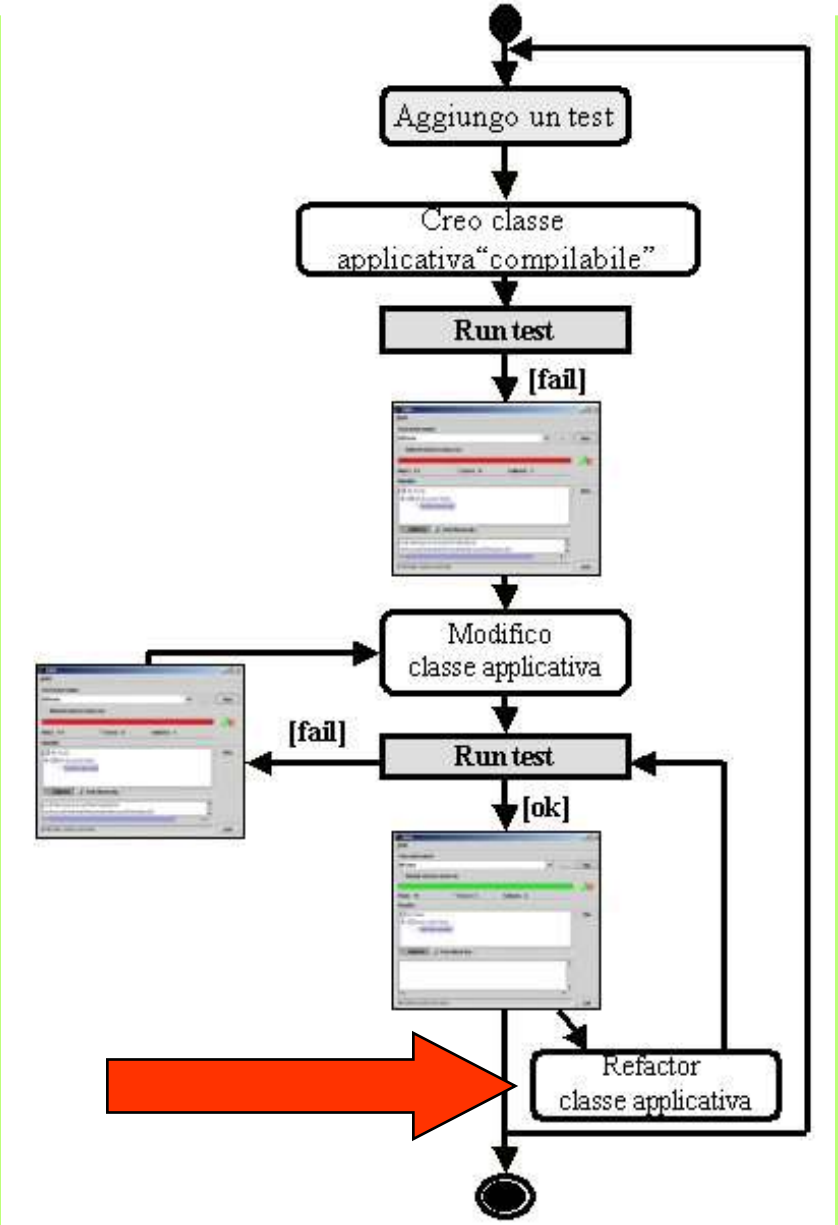
# Rework

```
public class Test_account extends TestCase {

public void testSettlement() {
      …..
}

public void testRealCaseSettlement() {
  CAccount c = new CAccount();
  for (int i=0; i <10 ; i++)
        c.deposit(1);
  c.draw(-10);
  assertTrue(c.settlement() == 0);
 }
}
```


Aggiungo un test
Creo classe applicativa"compilabile"
Run test
[fail]
Modifico classe applicativa
[fail]
Run test
[ok]
Refactor classe applicativa

# Run JUnit (4)


Java - Account/src/Test_account.java - Eclipse SDK
File Edit Source Refactor Navigate Search Project Run Window Help
Debug
Java Browsing
Java
JUnit
Finished after 0,015 seconds
Runs: 2/2
Errors: 0
Failures: 0
Test_account [Runner: JUnit 3] (0,000 s)
testSettlement (0,000 s)
testRealCaseSettlement (0,000 s)
Failure Trace

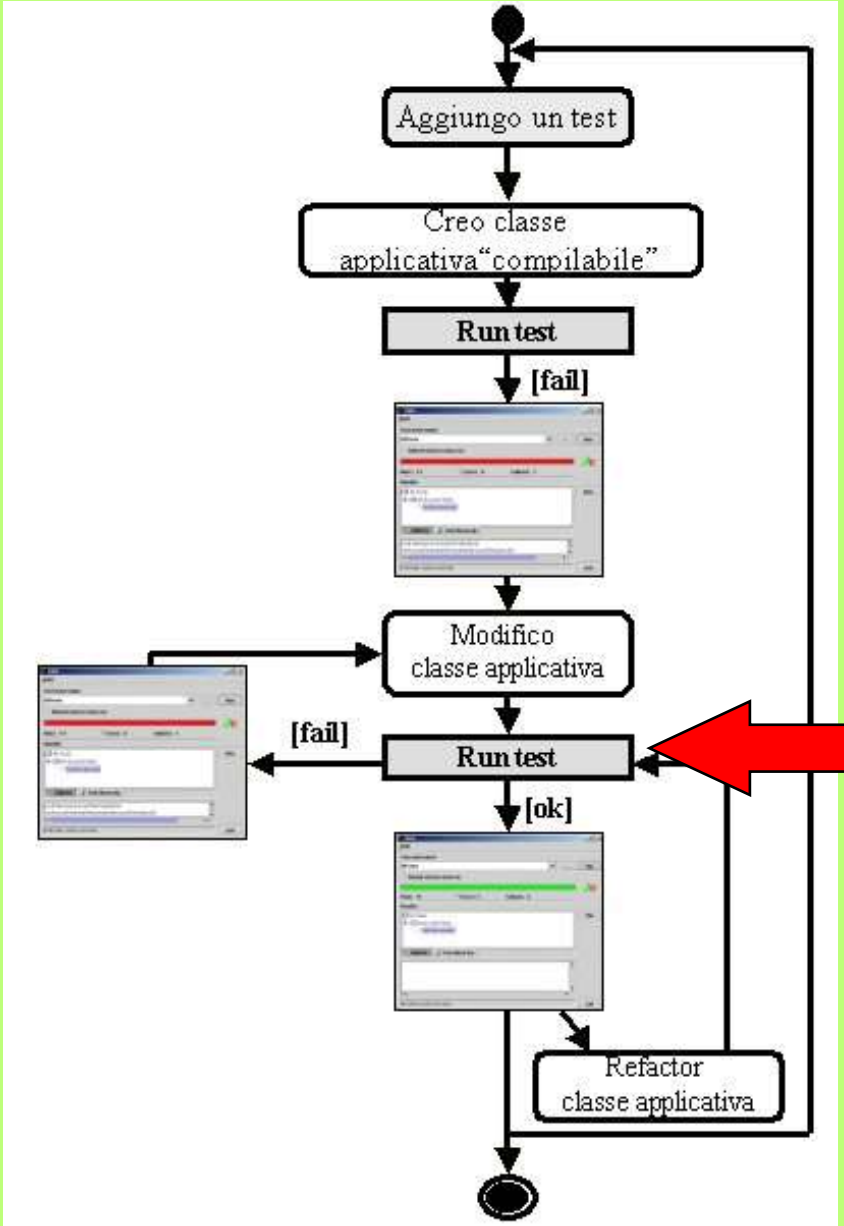



*"I can refactoring the program without anxiety. I have the testcases ..."*

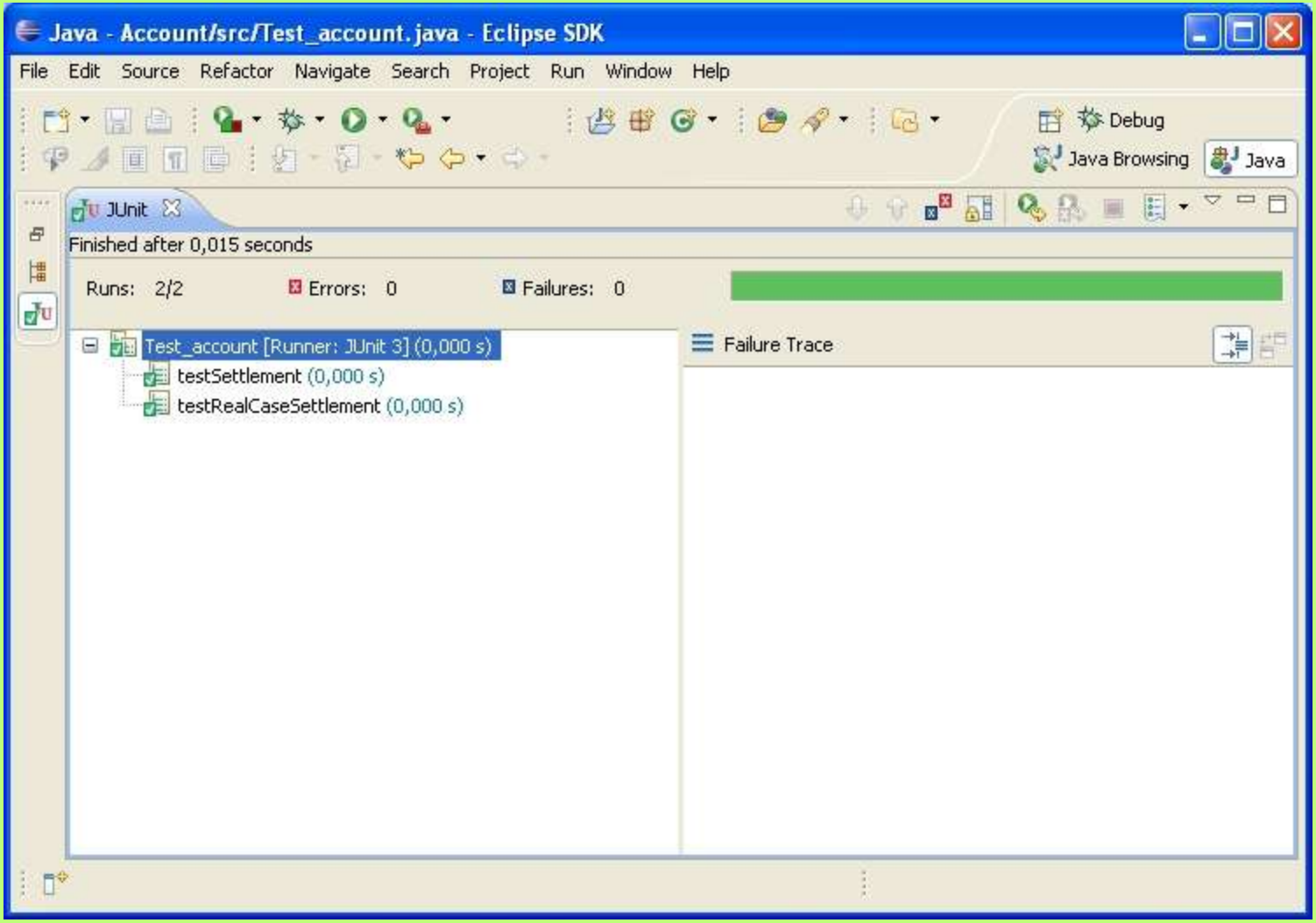
```
public class CAccount{
  List<Integer> account= new
     LinkedList<Integer>();
  public void deposit(int value){
    account.add(value);
   }
  public void draw(int value){
    account.add(value);
  }
  public int settlement() {
      int result = 0;
      for(int val: account) {
            result = result + val;
       }
  return result;
  }
}
```

# Refactor
## (Java generics)

```
public class Test_account extends TestCase {

public void testSettlement() {
      …..
}

public void testRealCaseSettlement() {
  CAccount c = new CAccount();
  for (int i=0; i <10 ; i++)
         c.deposit(1);
  c.draw(-10);
  assertTrue(c.settlement() == 0);
 }
}
```


Aggiungo un test
Creo classe applicativa"compilabile"
Run test
[fail]
Modifico classe applicativa
[fail]
Run test
[ok]
Refactor classe applicativa

# Run JUnit (5)


Java - Account/src/Test_account.java - Eclipse SDK
File Edit Source Refactor Navigate Search Project Run Window Help
Debug
Java Browsing
Java
JUnit
Finished after 0,015 seconds
Runs: 2/2
Errors: 0
Failures: 0
Test_account [Runner: JUnit 3] (0,000 s)
testSettlement (0,000 s)
testRealCaseSettlement (0,000 s)
Failure Trace

# Does Research Support the Effectiveness of Test Driven Development?

# Empirical studies

**Long list …**

| | |
|---|---|
| Madeyski [56] | CE |
| George, Williams [8] | QCE |
| Bhat, Nagappan [17] | CS |
| Erdogmus [3] | CE |
| Flor, Schneider [32] | QCE |
| Gupta, Jalote [57] | CE |
| Huang, Holcombe [4] | CE |
| Janzen, Saiedian [55] | QCE,CS |
| Janzen, Saiedian [24] | CE |
| Crispin [54] | CS |
| Geras et al. [16] | QCE |
| Pančur et al. [7] | CE |
| Siniaalto [21] | CS |
| Mueller, Hagner [5] | QCE |
| Madeyski [58] | CE |

…

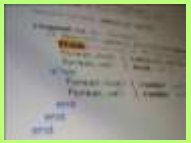
**CE = Controlled Exp**
**QCE = Quasi-Controlled Exp**
**CS = Case Study**

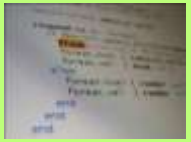
**Industrial Context**
**With professionals (toy example)**
**Academic Context**

# Experiment

TLD

implement





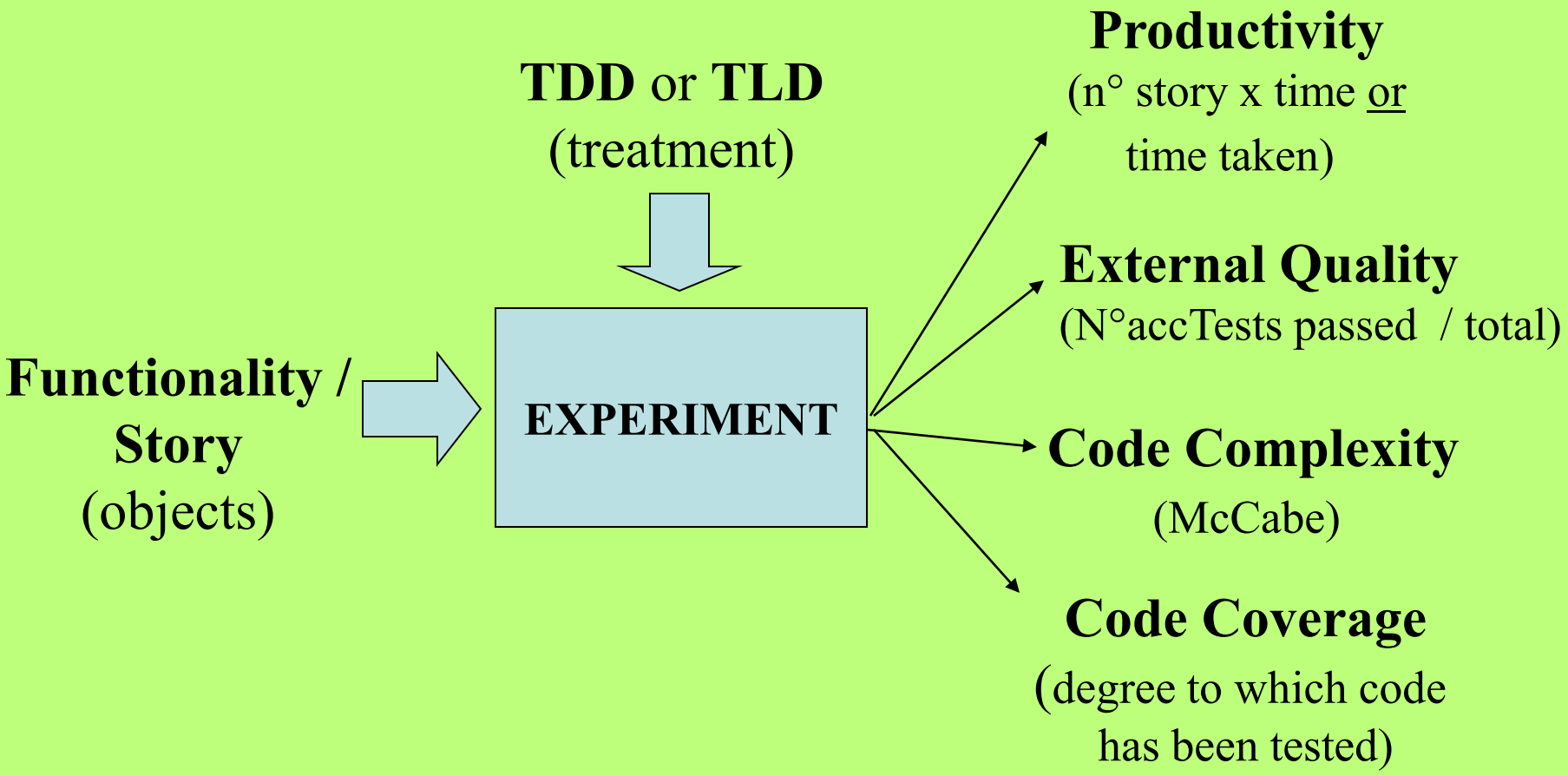
TDD

implement

?

2 Groups

# Variables


TDD or TLD
(treatment)
Functionality / Story
(objects)
EXPERIMENT
Productivity
(n° story x time or time taken)
External Quality
(N°accTests passed / total)
Code Complexity
(McCabe)
Code Coverage
(degree to which code has been tested)

Information and Software Tecnology 2011

Impact of test-driven development on productivity, code and tests: a controlled experiment

Matjaž Pančur[1] and Mojca Ciglarič

# Effects of TDD

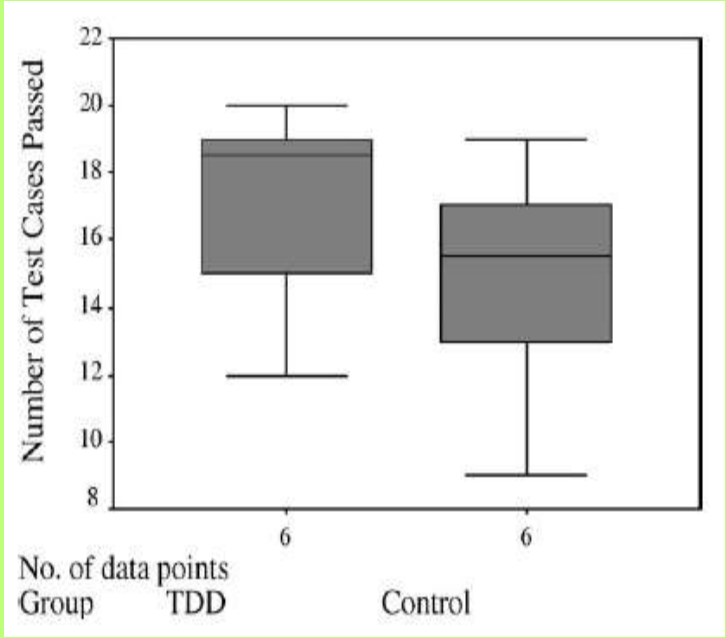
| | Study type | Productivity | External Quality | Complexity | Code Coverage |
|---|---|---|---|---|---|
| Madeyski [56] | CE | < | < | | > |
| George, Williams [8] | QCE | < | > | | |
| Bhat, Nagappan [17] | CS | < | | | > |
| Erdogmus [3] | CE | > | o | | |
| Flor, Schneider [32] | QCE | > | | | < |
| Gupta, Jalote [57] | CE | > | > | | |
| Huang, Holcombe [4] | CE | > | o | | |
| Janzen, Saiedian [55] | QCE,CS | | | > | |
| Janzen, Saiedian [24] | CE | | > | | |
| Crispin [54] | CS | | > | | |
| Geras et al. [16] | QCE | | o | | |
| Pančur et al. [7] | CE | | < | | < |
| Siniaalto [21] | CS | | | | > |
| Mueller, Hagner [5] | QCE | | | | < |
| Madeyski [58] | CE | | < | | |
| | | 4-3 | 4-3 | 1-0 | 3-3 |

15

(>) positive (<) negative (o) null

# George, Williams [8]

- Result:
  - TDD devs passed 18% more tests
  - TDD devs took 16% more time
- Threats to validity
  - Results are not statistically significant
  - Sample very small (24 subjects)
  - Pair programming
  - Application very small
    - 200 LOC
  - Different experience devs




**With professionals (toy example)**

# Conclusion

- Several supposed benefits of TDD
- Currently research **seems not support TDD** effectiveness
  - Overall result is not clear-cut
- But more replications are needed!
  - "Building Knowledge through Families of Experiments"
    - Victor R. Basili et alt., IEEE Trans. Software Eng. 25(4): 456-473 (1999)
- But a "serious" systematic review is needed!
  - Meta-analysis

# Literature (1)


3. H. Erdogmus, M. Morisio, M. Torchiano, On the Effectiveness of the Test-First Approach to Programming, IEEE Trans. Software Eng. 31(3), pp. 226-237 (2005)
4. L. Huang, M. Holcombe, Empirical investigation towards the effectiveness of Test First programming, Information and Software Technology 51 (2009), pp. 182–194.
5. M. M. Mueller, O. Hagner, Experiment about test-first programming, IEE Procedings-Software 149(5), 2002, pp. 131–136.
7. M. Pančur, M. Ciglarič, M. Trampuš, T. Vidmar, Towards empirical evaluation of test-driven development in a university environment, in EUROCON '03, Proceedings of the International Conference on Computer as a Tool, 2003, pp. 83–86.
8. B. George, L. Williams, A structured experiment on test-driven development, Information and Software Technology, vol. 46, Elsevier, 2004, pp. 337-342.
16. A. Geras, M. R. Smith, J. Miller, A Prototype Empirical Evaluation of Test Driven Development, in: IEEE METRICS'2004: Proceedings of the 10th IEEE International Software Metrics Symposium, IEEE Computer Society, 2004, pp. 405–416.
17. T. Bhat, N. Nagappan, Evaluating the efficacy of test-driven development: industrial case studies, in Proceedings of the 2006 ACM/IEEE international Symposium on Empirical Software Engineering (Rio de Janeiro, Brazil, September 21 – 22, 2006), ISESE '06, ACM, New York, NY, pp. 356-363.
21. M. Siniaalto, P. Abrahamsson, A Comparative Case Study on the Impact of Test-Driven Development on Program Design and Test Coverage, in ESEM '07: Proceedings of the First International Symposium on Empirical Software Engineering and Measurement, IEEE Computer Society, 2007, pp. 275-284.

# Literature (2)


24. D. Janzen, H. Saiedian, On the Influence of Test-Driven Development on Software Design, CSEET, 19th Conference on Software Engineering Education & Training (CSEET'06), 2006, pp.141-148.
32. T. Flohr, T. Schneider, Lessons Learned from an XP Experiment with Students:Test-First Needs More Teachings, in J. Münch and M. Vierimaa (Eds.): PROFES 2006, LNCS 4034, pp. 305 – 318.
54. L. Crispin, Driving Software Quality: How Test-Driven Development Impacts Software Quality, IEEE Software 23(6) 2006, pp. 70-71.
55. D. S. Janzen, H. Saiedian, Does Test-Driven Development Really Improve Software Design Quality?, IEEE Software, 25(2) 2008, pp. 77-84.
56. Madeyski, L.: Test-Driven Development: An Empirical Evaluation of Agile Practice. Springer, London, UK 2010.
57. A. Gupta, P. Jalote: An experimental evaluation of the effectiveness and efficiency of the test driven development. In: ESEM'07: International Symposium on Empirical Software Engineering and Measurement, pp. 285-294. IEEE Computer Society, Washington, DC, USA (2007).
58. L. Madeyski: Preliminary Analysis of the Effects of Pair Programming and Test-Driven Development on the External Code Quality. In: K. Zielinski, T. Szmuc (eds.): Software Engineering: Evolution and Emerging Technologies, IOS Press, 2005.

# Questions?