Clover Demo: Test Coverage in Action

Sara Sprenkle
CISC879
March 15, 2007
Review: When Have You Tested Enough?

- Time? It’s been a couple hours/days/…
- Number of test cases executed? A lot!
- I asked my brother and he’s really smart and he says that it’s enough
Review: When Have You Tested Enough?

- Time? It’s been a couple hours/days/…
- Number of test cases executed? A lot!
- I asked my brother and he’s really smart and he says that it’s enough

Need something more systematic: Coverage Criteria
Review: Uses of Coverage Criteria
Review: Uses of Coverage Criteria

- “Stopping” rule → sufficient testing
  - Avoid unnecessary, redundant tests
Review: Uses of Coverage Criteria

- “Stopping” rule → sufficient testing
  - Avoid unnecessary, redundant tests
- Measure test quality
  - Dependability estimate
  - Confidence in estimate
Review: Uses of Coverage Criteria

- “Stopping” rule $\rightarrow$ sufficient testing
  - Avoid unnecessary, redundant tests

- Measure test quality
  - Dependability estimate
  - Confidence in estimate

- Specify test cases
  - Describe additional test cases needed
Adequacy Criteria

[Zhu 97]

Data Flow

Highest Expense

Mutation Testing

Lowest

Control Flow
Adequacy Criteria

[Zhu 97]

Clover measures

Data Flow

Highest Expense

Mutation Testing

Lowest
Statement Coverage

- Cover all **statements** in the program

```java
public String exampleMethod(int num) {
    String string = null;
    if (num < 10) {
        string = "" + condition;
    }
    return string.trim();
}
```
Statement Coverage

- Cover all **statements** in the program

Test Suite:

num=5

```java
public String exampleMethod(int num) {
    String string = null;
    if (num < 10) {
        string = "" + condition;
    }
    return string.trim();
}
```
Statement Coverage

- Cover all **statements** in the program

Test Suite:

num=5

```java
public String exampleMethod(int num) {
    String string = null;
    if (num < 10) {
        string = "" + condition;
    }
    return string.trim();
}
```
Statement Coverage

- Cover all **statements** in the program

Test Suite:

```java
public String exampleMethod(int num) {
    String string = null;
    if (num < 10) {
        string = "" + condition;
    }
    return string.trim();
}
```
Statement Coverage

- Cover all **statements** in the program

Test Suite:

```
num=5
```

```
public String exampleMethod(int num) {
    String string = null;
    if (num < 10) {
        string = "" + condition;
    }
    return string.trim();
}
```
Statement Coverage

- Cover all **statements** in the program

Test Suite:

```
num=5
```

```
public String exampleMethod(int num) {
    String string = null;
    if (num < 10) {
        string = "" + condition;
    }
    return string.trim();
}
```
public String exampleMethod(int num) {
    String string = null;
    if (num < 10) {
        string = "" + condition;
    }
    return string.trim();
}
public String exampleMethod(int num) {
    String string = null;
    if (num < 10) {
        string = "" + condition;
    }
    return string.trim();
}
What Went Wrong?

- Test suite had 100% statement coverage but missed a branch/edge

- Try covering all edges in program’s flow
  - Also covers all nodes
  - Called Branch Coverage

```java
exampleMethod(int num)

String string = null;

if (num < 10)
    string = "" + condition;
    string.trim();
```

- Implicit false Branch
- Called Branch Coverage
Branch Coverage

- Cover all branches in the program

Test Suite:
- num=5,
- num=10

CISC 879: Clover Demo
Sara Sprenkle
Branch Coverage

- Cover all **branches** in the program

Test Suite:
- num=5
- num=10

```java
demoMethod(int num)
    String string = null;
    if (num < 10)
        string.trim();
```

false Branch

```
string = "" + condition;
string.trim();
```
Branch Coverage

- Cover all **branches** in the program

**Test Suite:**
- num=5,
- num=10

```java
exampleMethod(int num)
String string = null;
if (num < 10)
  string = "" + condition;
  string.trim();
```

**Implicit false Branch**
Branch Coverage

- Cover all **branches** in the program

Test Suite:
- num=5
- num=10

```java
exampleMethod(int num)

String string = null;
if (num < 10)
    string.trim();
```

- Cover all branches in the program

Test Suite:
- num=5
- num=10

```java
exampleMethod(int num)

String string = null;
if (num < 10)
    string = "" + condition;
    string.trim();
```

- Cover all branches in the program

Test Suite:
- num=5
- num=10

```java
exampleMethod(int num)

String string = null;
if (num < 10)
    string = "" + condition;
    string.trim();
```
Branch Coverage

- Cover all **branches** in the program

Test Suite:
- num=5,
- num=10

```java
exampleMethod(int num)

String string = null;
if (num < 10)
  string.trim();

if (num < 10)
  String string = null;

string = "" + condition;

string.trim();
```

- **true**
- **false** Branch

Implicit Branch

CISC 879: Clover Demo
Sara Sprenkle
Example 2

```java
public int exampleMethod(int a) {
    String str = “d”;
    if (a < 7) {
        a *= 2;
        str += “riv”;
    } else {
        str = “co” + str;
    }

    if (a > 10) {
        str += “ing”;
    } else {
        str += “es”;
    }

    return str.substring(6);
}
```
Example 2

public int exampleMethod(int a) {
    String str = “d”;
    if ( a < 7 ) {
        a *= 2;
        str += “riv”;
    } else {
        str = “co” + str;
    }
    if( a > 10 ) {
        str += “ing”;
    } else {
        str += “es”;
    }
    return str.substring(6);
}

Branch Coverage

Test Suite:
  a=3,
a=30

```java
def exampleMethod(int a):
    str = "d"
    if (a < 7):
        a *= 2
        str += "riv"
    if (a > 10):
        str += "ing"
        if (a > 10):
            str += "es"
    return str.substring(6)
```
Branch Coverage

Test Suite:
- a=3
- a=30
- a=6

```
String str = "d";
if (a < 7)
    return str.substring(6);
    a *= 2;
    str += "riv";
if (a > 10)
    str += "ing";
    str += "es";
return str.substring(6);
```
Branch Coverage

Test Suite:
   a=3, a=30
Branch Coverage

Test Suite:
- a=3, str="cod"
- a=30, str="coding"

```
exampleMethod(int a)

String str = "d";

if( a < 7 )
    return str.substring(6);

a *= 2;
str += "riv";
str = "co" + str;
if( a > 10 )
    str += "ing";
    str += "es";
return str.substring(6);
```
Branch Coverage

Test Suite:  
a=3,  
a=30

```java
exampleMethod(int a)

String str = “d”;

if( a < 7 )
    a *= 2;
    str += “riv”; 
else
    str = “co” + str;

if( a > 10 )
    str += “ing”;
    str += “es”;
else
    return str.substring(6);
```
What Went Wrong?

- Test suite had 100% branch (and statement) coverage but missed a path
- Try to cover all paths in program’s flow
  - Also gets all branches, nodes
  - Called Path Coverage
  - Not generally practical

```java
exampleMethod(int a)

String str = “d”;

if( a < 7 )
  a *= 2;
  str += “riv”;  // 3
  str = “co” + str;  // 4
  if( a > 10 )
    str += “ing”;
    str += “es”;  // 6, 7
    return str.substring(6);  // 8
else
  str += “es”;  // 5
```

CISC 879: Clover Demo
Sara Sprenkle
Code Coverage Tools

- Coverage is used in practice
- You don’t need to figure out coverage manually
- Some available tools to calculate coverage
  - For Java: Clover, JCoverage, Emma, …
  - For C/C++: BullseyeCoverage, CoverageMeter, …
  - For C#: NClover, …
- And many more …
Cenqua’s Clover

- Tool to measure code coverage
- Web site: http://www.cenqua.com/clover/
- Code coverage used to
  - Measure quality of test suite
  - Improve test suite
  - Determine when to stop testing
A Little History of Clover

- Initially developed as an internal tool
  - Couldn’t find a reasonably priced coverage tool that also
    - Performed well in a continuous integration environment
    - Performed well with large J2EE applications
- Released as a side project in May 2002
- Within 12 months had eclipsed Cenqua’s traditional services business
- Name comes from shortened version of “Cover Lover”
How Clover Works

- Instruments source code w/ code to report coverage
How Clover Works

- Instruments source code w/ code to report coverage
How Clover Works

- Instruments source code w/ code to report coverage

```java
public void method() {
    _cl_method1_cnt++;
    x=y+z;
    _cl_stmt1_cnt++;
    ...
}
...
```

Approximation of Instrumentation
How Clover Works

- Compiles instrumented code

Source code

Instrumented Source code

Doesn’t actually output these

- Compiles instrumented code
How Clover Works

- Compiles instrumented code

Source code → Instrumented Source code → Compiled Instrumented Classes

 Doesn’t actually output these

CISC 879: Clover Demo  Sara Sprenkle
How Clover Works

- Records coverage as tests are executed

Compiled Instrumented Classes
How Clover Works

- Records coverage as tests are executed
How Clover Works

- Records coverage as tests are executed
How Clover Works

- Records coverage as tests are executed
How Clover Works

- Generate coverage report
  - Various formats: XML, HTML, PDF, Plain Text, Swing
How I Use Clover
My Approach to Web Application Testing

User-session-based testing

Test Case

Web Application

Output

Users access

Beta Web Application (v.0.9) Deployment

Record accesses

User sessions

CISC 879: Clover Demo
Sara Sprenkle
My Approach to Web Application Testing

User-session-based testing

Test Case

Web Application

Output

User access

Beta Web Application (v.0.9)

Deployment

Record accesses

User sessions

Test v.1.0 with collected data

Web Application (v.1.0)
My Approach to Web Application Testing

- **Test cases:** real user accesses
  - Test code users frequently access
  - Likely redundant
Evaluating Reduced Test Suites

- Reduce original test suite
- Compare coverage of reduced test suites

Generate coverage, fault detection reports
How I Use Clover

Web Application code
How I Use Clover

Web Application code

Instrumented Using Ant Tasks/Targets code

Compiled Instrumented Classes
How I Use Clover

Web Application code

Instrumented code

Compiled Instrumented Classes

Web Application Server code

Ant copies instrumented classes to correct location
How I Use Clover

- Execute test suites

Web Application code → Instrumented code

Using **Ant** Tasks/Targets

Compiled Instrumented Classes

**Ant** copies instrumented classes to correct location

Web Application Server

CISC 879: Clover Demo
Sara Sprenkle
How I Use Clover

- Execute test suites
- Generate coverage reports

Web Application code

Instrumented Using Ant Tasks/Targets code

Compiled Instrumented Classes

Ant copies instrumented classes to correct location

Web Application Server code

CISC 879: Clover Demo
Sara Sprenkle
How I Use Clover

- Example: Course Project Manager (CPM)
How I Use Clover

Example: Course Project Manager (CPM)

Choose a group and demo to grade

- Demo: PA4
- Group: bmoyers

- Grade:
- Grade 2:

Comments:

Assign Grade  Reset

Main Menu  Log Out
## How I Use Clover

### Demo sign up for PA1

**Information about the demo**

Click a radio button below to select a group demo timeslot.

**NOTE:** You may want to refresh this page if it has been awhile since it was first loaded. Other groups may have already signed up for these slots.

<table>
<thead>
<tr>
<th>Time</th>
<th>Tuesday, September 5</th>
<th>Wednesday, September 6</th>
<th>Thursday, September 7</th>
<th>Friday, September 8</th>
</tr>
</thead>
<tbody>
<tr>
<td>9:00 a.m.</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9:30 a.m.</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10:00 a.m.</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10:30 a.m.</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11:00 a.m.</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11:30 a.m.</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

[PAGE]
Clover Coverage Reports

- HTML report
- XML report
Clover Features

- Fast, accurate coverage measurement
- Directives to choose which code to instrument
  - E.g., can exclude certain methods
- Integrated with Apache Ant, Maven
- Multiple report formats
  - Historical reporting too
More Clover Features

- Plugins for IDEs
  - Integrate testing/coverage into development
  - Eclipse, IntelliJ IDEA, Eclipse, NetBeans, JBuilder
- Can measure coverage for distributed applications
- Can use interactively
  - Look at coverage during testing process
Future of Clover: Clover 2.0

- Includes more than just coverage reports
- Results of test cases (pass/fail)
- Which tests hit which code
- Analyzes coverage results
  - Where to focus testing