Characterizing, Verifying, and Improving Software Resilience with Exception Contracts and Test Suites
Expected Problems
Unexpected Errors
Goal

Resilience : Capability of a system to heal itself in presence of unexpected errors

Our goal is to identify pieces of codes that are capable to handle unexpected errors.
Exception ⊂ Error

- No exception
  - A B C E

- MyException on B
  - A B D E

- OtherException on B
  - A B E
Expected errors in Test Suite?

- Test colors
  - Pink
  - Blue
  - White

```java
public String getProperty(String s) {
    String res = null;
    try{
        res = getPropertyFromFile(s);
    }catch(PropertyNotFoundException pnfe){
        return null;
    }
    return res;
}

@Test
public void testAbsentProperty() {
    assertNotNull(getProperty("@$%^*_\mu;?!"));
}
```
# Formal definition

<table>
<thead>
<tr>
<th></th>
<th>Exception thrown in app code</th>
<th>Exception caught in app code</th>
<th>Exception reaching the test code</th>
</tr>
</thead>
<tbody>
<tr>
<td>pink</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>White (expected)</td>
<td>&gt;0</td>
<td>&gt;0</td>
<td>0</td>
</tr>
<tr>
<td>Blue (expected)</td>
<td>&gt;0</td>
<td>&gt;=0</td>
<td>&gt;0</td>
</tr>
</tbody>
</table>
How to determine test cases' color?

```java
try{
    // developer code
} catch (RuntimeException re){
    // developer code
}
```

```java
try{
    Framework.learn("try-start");
    // developer code
    Framework.learn("try-end");
    // developer code
    catch (RuntimeException re){
        Framework.learn("catch-start", re);
        // developer code
        Framework.learn("catch-end");
    }
}
```
There are expected errors
Unexpected Errors

Simulating unexpected errors

Study behavior of code under unexpected errors
Why ?
The system state is correct
Short-Circuit Testing

White test

+ throw new X() =

13
The try-catch blocks that keep the test green are capable to handle unexpected errors.

We call them "fault-independent"
There are fault independent try-catches
Related Work

- Amplifying tests to validate exception handling code (Zhang and Elbaum, ICSE 2012)

- Using fault injection to increase software test coverage (Bieman and al., 1996)
Conclusion

• Definition of 3 new test case kinds (colors).

• Algorithm for identifying try-catch able to handle unexpected exceptions.

• Empirical discovery of fault-independent (92) try-catch blocks.