

Test Driven Design Challenges for Faster Product Development

Jason Fraser: **fraser.jason@gmail.com**
Baldev S. Mattu: **bsmhsn@hotmail.com**

Center for Systems Integration
Department of Computer Science and Engineering
Florida Atlantic University

Outline

- Introduction
- Methods
- Challenges/Experiences
- Conclusions
- Future Work
- References

Introduction

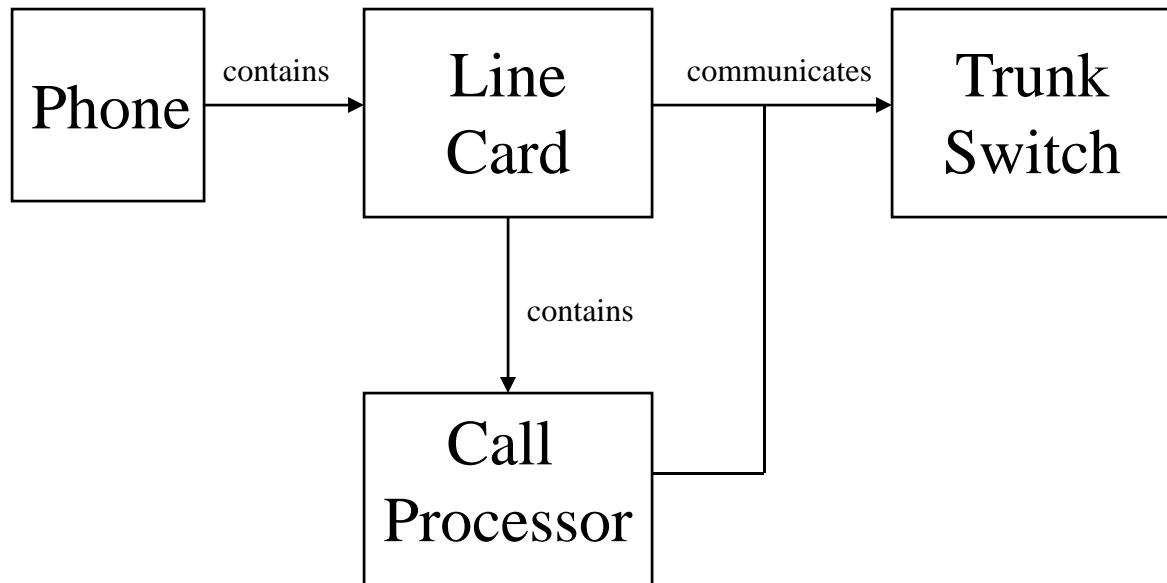
- **Test Driven Design (TDD)**
 - What is it?
- **The Advantages**
 - Reduced debug time
 - Faster code development
 - Increased fulfillment of customer requirements
 - No hardware required at the beginning of development

Introduction (cont.)

- Despite these advantages, why has TDD not been embraced?
- Are there problems with TDD, or difficulties in implementing TDD?

Methods

- Implementation of PBX system



Methods (cont.)

- Requirements
 - System requirement: PBX with 10 outside lines...etc.
- Use cases
 - Our use cases are broken into user stories
- For each coding cycle:
 - One user story
 - Unit test for that story
 - Business logic (functional) code

Test first? (example)

- How do we write the test before we have any functional code to test?
- Simplest example: On-hook and Off-hook user stories.

User Story

- User takes extension off-hook and receives a dial tone.

JUnit Test

```
package pbx;
import junit.framework.TestCase;

public class OffHookTest extends TestCase {

    public OffHookTest(String name) {
        super(name);
    }
    public void testOffHook() {
        CallProcessor cp = new CallProcessor();
        SimulatedLineCard_2 slc = new SimulatedLineCard_2(cp);
        assertTrue(slc.isDialToneOn() == false);
        slc.simulateOffHook();
        assertTrue(slc.isDialToneOn());
    }
}
```

Java - OnHookTest.java - Eclipse SDK

File Edit Source Refactor Navigate Search Project Run Window Help

Package Explorer JUnit x

Finished after 0.14 seconds

Runs: 1/1 Errors: 0 Failures: 1

Failures Hierarchy

- testOnHook - pbx.OnHookTest

```
/* Added 11/04/06 by Jason Fraser
package pbx;
import junit.framework.TestCase;

public class OnHookTest extends TestCase {

    public OnHookTest(String name) {
        super(name);
    }

    public void testOnHook() {
        CallProcessor cp = new CallProcessor();
        SimulatedLineCard_2 slc = new SimulatedLineCard_2(cp);
        assertTrue(slc.isDialToneOn() == true);
        slc.simulateOffHook();
        assertTrue(slc.isDialToneOn());
        slc.simulateOnHook();
        assertTrue(slc.isDialToneOn() == false);
    }
}
```

Failure Trace

- junit.framework.AssertionFailedError: null
- at pbx.OnHookTest.testOnHook(OnHookTest.java:15)
- at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
- at sun.reflect.NativeMethodAccessorImpl.invoke(Unknown Source)
- at sun.reflect.DelegatingMethodAccessorImpl.invoke(Unknown Source)

Problems Javadoc Declaration Console Plug-in Dependencies Search

start C:\Documents and Se... C:\Eclipse3_1\ eclipse Microsoft PowerPoint ... Java - OnHookTest.j... 11:32 AM

Develop the code

CallProcessor:

```
package pbx;
import java.lang.String;
//CallProcessor.java - Version 2
public class CallProcessor {
    public CallProcessor() {
    }
    public void offHook(LineCard lc) {
        lc.dialToneOn();
    }
}
```

LineCard:

```
package pbx;
//LineCard.java
public interface LineCard {
    public void dialToneOn();
}
```

Simulated LineCard:

```
package pbx;
import java.lang.String;

//SimulatedLineCard_2.java
public class SimulatedLineCard_2 implements LineCard {
    boolean dialToneOn = false;

    CallProcessor callProcessor;
    SimulatedTrunkSwitch ts = new SimulatedTrunkSwitch();
    public SimulatedLineCard_2(CallProcessor callProcessor) {
        this.callProcessor = callProcessor;
    }
    public void dialToneOn() {
        dialToneOn = true;
    }
    public boolean isDialToneOn() {
        return dialToneOn;
    }
    public void simulateOffHook(){
        callProcessor.offHook(this);
    }
}
```

Java - OnHookTest.java - Eclipse SDK

File Edit Source Refactor Navigate Search Project Run Window Help

Package Explorer JUnit x

Finished after 0.047 seconds

Runs: 1/1 Errors: 0 Failures: 0

Failures Hierarchy

Failure Trace

```
CallProcessor.java CallProcessorTest... OnHookTest.java TrunkSwitch.java
```

/* Added 11/04/06 by Jason Fraser

```
package pbx;
import junit.framework.TestCase;

public class OnHookTest extends TestCase {

    public OnHookTest(String name) {
        super(name);
    }

    public void testOnHook() {
        CallProcessor cp = new CallProcessor();
        SimulatedLineCard_2 slc = new SimulatedLineCard_2(cp);
        assertTrue(slc.isDialToneOn() == false);
        slc.simulateOffHook();
        assertTrue(slc.isDialToneOn());
        slc.simulateOnHook();
        assertTrue(slc.isDialToneOn() == false);
    }
}
```

pbx
import declare
junit.fra
OnHookTest
OnHookT
testOnH

Problems Javadoc Declaration Console Plug-in Dependencies Search

User Story

- Extension is placed on-hook, and phone is reset.
- What does this mean?
 - Domain knowledge needed
- Follow same test and code cycle

Unit Test

```
package pbx;
import junit.framework.TestCase;

public class OnHookTest extends TestCase {

    public OnHookTest(String name) {
        super(name);
    }

    public void testOnHook() {
        CallProcessor cp = new CallProcessor();
        SimulatedLineCard_2 slc = new SimulatedLineCard_2(cp);
        assertTrue(slc.isDialToneOn() == false);
        slc.simulateOffHook();
        assertTrue(slc.isDialToneOn());
        slc.simulateOnHook();
        assertTrue(slc.isDialToneOn() == false);
    }
}
```

Develop the code

CallProcessor:

```
package pbx;
import java.lang.String;
//CallProcessor.java - Version 2
public class CallProcessor {
    public CallProcessor() {
    }
    public void offHook(LineCard lc) {
        lc.dialToneOn();
    }
    public void onHook(LineCard lc) {
        lc.dialToneOff();
    }
}
```

LineCard:

```
package pbx;
```

```
//LineCard.java
```

```
public interface LineCard {
```

```
    public void dialToneOn();
```

```
    public void dialToneOff();
```

```
}
```

Simulated LineCard:

```
package pbx;
import java.lang.String;

//SimulatedLineCard_2.java
public class SimulatedLineCard_2 implements LineCard
{
    boolean dialToneOn = false;
    CallProcessor callProcessor;
    SimulatedTrunkSwitch ts = new SimulatedTrunkSwitch();
    public SimulatedLineCard_2(CallProcessor callProcessor) {
        this.callProcessor = callProcessor;
    }
    public void dialToneOn() {
        dialToneOn = true;
    }
    public void dialToneOff() {
        dialToneOn = false;
    }
    public boolean isDialToneOn() {
        return dialToneOn;
    }
    public void simulateOffHook(){
        callProcessor.offHook(this);
    }
    public void simulateOnHook(){
        callProcessor.onHook(this);
    }
}
```

- Other user stories are fulfilled in the same way one at a time:
 - Write the test,
 - Run the test,
 - Write the code,
 - Run the test,
 - Fix the code,
 - Run the test, etc.

Challenges/Experiences

- Interpretation of user stories
 - Need domain knowledge
- Writing ‘correct’ unit tests
 - How is this any different from traditional development?
- Ordering of implementation
 - Taken care of by team and customer collaboration
 - Requirements are reviewed multiple times.

Challenges/Experiences (cont.)

- Inferring missing user stories/use cases
 - Need continued involvement of domain experts
- Duplicate user stories
 - Team involvement, automation, and refactoring
- Designing from the user stories
 - Again, domain experts needed and how is this different from traditional development?

Conclusions

- Design architecture before user story generation
- Involve domain expert and user in all stages of product development
- Give it time; switching to TDD takes a shift in thinking from the whole team

Future Work

- Develop use cases for our PBX system that have real-time constraints
- Develop use cases that have concurrency requirements, and take them through the TDD cycle

References

- [1] Grenning, J., *Extreme Programming and Embedded Software Development*
<http://www.objectmentor.com/resources/articles/EmbeddedXp.pdf>
- [2] Mattu, B., Shankar, R., *Test-Driven Design Methodology for Component Based Systems* (draft version to be submitted to 1st Annual IEEE Systems Conference)