

Mining framework concepts from application code

Steven She



Generative Software Developement Lab

August 19, 2009

Outline

- 1 Introduction
- 2 Mining for Framework Concepts
- 3 Conclusions

Problem - Java Applet

```
import java.applet.Applet;
```

```
public class HelloApplet extends Applet {
```

```
}
```

Problem - Java Applet

```
import javax.swing.JApplet;
```

```
public class HelloApplet extends JApplet {
```

```
}
```

Problem - Java Applet

```
import javax.swing.JApplet;

public class HelloApplet extends JApplet {

    public void init() {
        showStatus("Hello, world!");
    }

}
```

Problem - Java Applet

```
import javax.swing.JApplet;
import java.awt.event.KeyListener;

public class HelloApplet extends JApplet {

    KeyListener keyList = new KeyListener() {
        public void keyTyped(KeyEvent e) { /* Do something */ }
        public void keyPressed(KeyEvent e) {}
        public void keyReleased(KeyEvent e) {}
    }

    public void init() {
        showStatus("Hello, world!");
    }

}
```

Problem - Java Applet

```
import javax.swing.JApplet;
import java.awt.event.KeyListener;

public class HelloApplet extends JApplet {

    KeyListener keyList = new KeyListener() {
        public void keyTyped(KeyEvent e) { /* Do something */ }
        public void keyPressed(KeyEvent e) {}
        public void keyReleased(KeyEvent e) {}
    }

    public void init() {
        showStatus("Hello, world!");
        addKeyListener(keyList);
    }

}
```

Problem - Java Applet

```
import javax.swing.JApplet;
import java.awt.event.KeyListener;

public class HelloApplet extends JApplet {

    KeyListener keyList = new KeyListener() {
        public void keyTyped(KeyEvent e) { /* Do something */ }
        public void keyPressed(KeyEvent e) {}
        public void keyReleased(KeyEvent e) {}
    }

    public void init() {
        showStatus("Hello, world!");
        addKeyListener(keyList);
    }

    removeKeyListener(keyList);

}
```


Problem - Java Applet

```
import javax.swing.JApplet;
import java.awt.event.KeyListener;

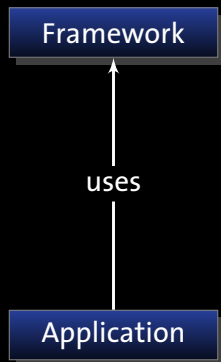
public class HelloApplet extends JApplet {

    KeyListener keyList = new KeyListener() {
        public void keyTyped(KeyEvent e) { /* Do something */ }
        public void keyPressed(KeyEvent e) {}
        public void keyReleased(KeyEvent e) {}
    }

    public void init() {
        showStatus("Hello, world!");
        addKeyListener(keyList);
    }

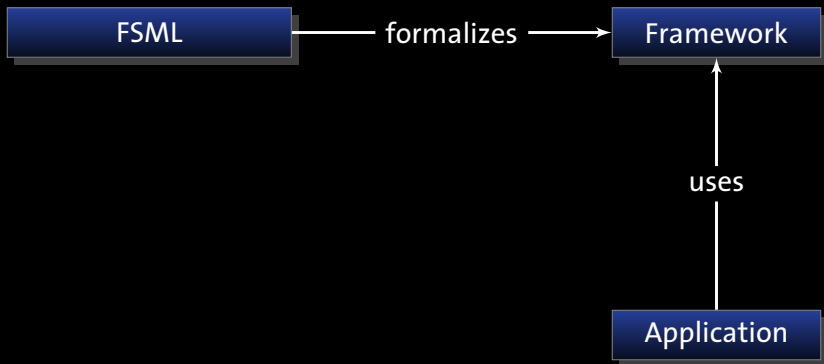
    public void destroy() {
        removeKeyListener(keyList);
    }
}
```

Framework-Specific Modelling Languages



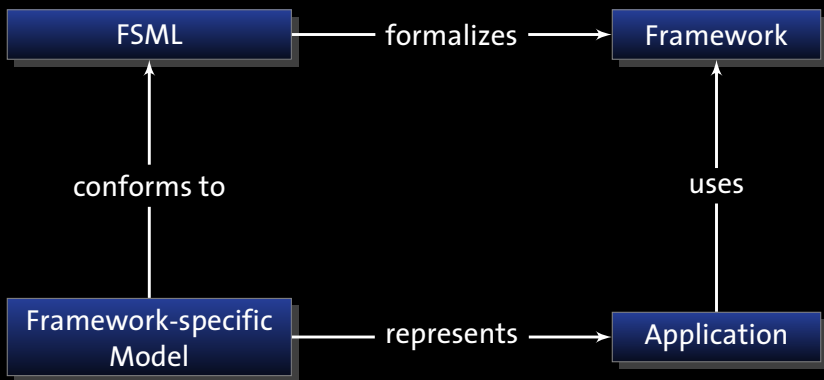
^o M. Antkiewicz, T. Tonelli Bartolomei, K. Czarnecki

Framework-Specific Modelling Languages



^o M. Antkiewicz, T. Tonelli Bartolomei, K. Czarnecki

Framework-Specific Modelling Languages



^o M. Antkiewicz, T. Tonelli Bartolomei, K. Czarnecki

Applet FSML

Applet FSML

[1..1] name (String)

![1..1] extendsApplet

[0..1] extendsJApplet

[0..*] registersKeyListener

!⟨1 - 1⟩

[0..1] asThis

...

[0..1] asAField

[1..1] listenerField (String)

[1..1] typedKeyListener

[1..1] initialized

[1..1] deregisters

[0..*] showsStatus

[1..1] message (String)

class name

extends Applet

extends JApplet

calls addKeyListener(...)

parameter is a field

field name

typed KeyListener(...)

assigned an object

calls removeKeyListener(...)

calls showsStatus(...)

parameter value

Applet FSML

Applet FSML

[1..1] name (String)

![1..1] extendsApplet

[0..1] extendsJApplet

[0..*] registersKeyListener

!⟨1 - 1⟩

[0..1] asThis

...

[0..1] asAField

[1..1] listenerField (String)

[1..1] typedKeyListener

[1..1] initialized

[1..1] deregisters

[0..*] showsStatus

[1..1] message (String)

class name

extends Applet

extends JApplet

calls addKeyListener(...)

parameter is a field

field name

typed KeyListener(...)

assigned an object

calls removeKeyListener(...)

calls showsStatus(...)

parameter value

Forward & Reverse

Model

Code

Applet Instance

```
name ← ``HelloApplet"
extendsApplet ← ✓
  extendsJApplet ← x
registersKeyListener ← x
  ⟨group⟩
  asThis ← x
  ...
  asAField ← x
    listenerField (String) ← x
    typedKeyListener ← x
    initialized ← x
    deregisters ← x
showsStatus ← ✓
  message ← ``Hello, world!"
```

Forward & Reverse

Model

Applet Instance

```

name ← ``HelloApplet"
extendsApplet ← ✓
  extendsJApplet ← x
registersKeyListener ← x
  ⟨group⟩
  asThis ← x
  ...
  asAField ← x
    listenerField (String) ← x
    typedKeyListener ← x
    initialized ← x
    deregisters ← x
showsStatus ← ✓
  message ← ``Hello, world!"

```

Code

```

public class HelloApplet extends Applet {

    public void init() {
        showStatus("Hello, world!");
    }

}

```


Forward & Reverse

Model

Applet Instance

```

name ← ``HelloApplet"
extendsApplet ← ✓
  extendsJApplet ← x
registersKeyListener ← x
  ⟨group⟩
  asThis ← x
  ...
  asAField ← ✓
  listenerField (String) ← ...
  typedKeyListener ← ✓
  initialized ← x
  deregisters ← x
showsStatus ← ✓
  message ← ``Hello, world!"

```

Code

```

public class HelloApplet extends Applet {

    KeyListener keyList

    public void init() {
        showStatus("Hello, world!");
        addKeyListener(keyList);
    }

}

```

Forward & Reverse

Model

Applet Instance

```

name ← ``HelloApplet"
extendsApplet ← ✓
  extendsJApplet ← x
registersKeyListener ← x
  ⟨group⟩
  asThis ← x
  ...
  asAField ← ✓
    listenerField (String) ← ...
    typedKeyListener ← ✓
    initialized ← ✓
    deregisters ← ✓
showsStatus ← ✓
  message ← ``Hello, world!"

```

Code

```

public class HelloApplet extends Applet {

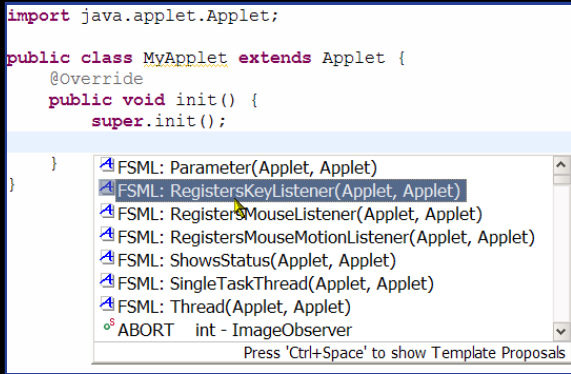
    KeyListener keyList = new KeyListener() { ... };

    public void init() {
        showStatus("Hello, world!");
        addKeyListener(keyList);
    }

    public void destroy() {
        removeKeyListener(keyList);
    }
}

```

Tool Support: Forward Eng.

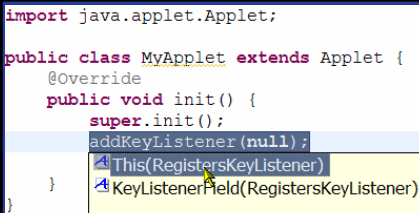


^o H. M. Lee. Model-guided Code Assistance for Framework Application Development.

Tool Support: Forward Eng. (cont.)

```
import java.applet.Applet;

public class MyApplet extends Applet {
    @Override
    public void init() {
        super.init();
        addKeyListener(null);
    }
}
```



^o H. M. Lee. Model-guided Code Assistance for Framework Application Development.

Tool Support: Forward Eng. (cont.)

```
import java.applet.Applet;
import java.awt.event.KeyListener;
import java.awt.event.KeyEvent;

public class MyApplet extends Applet implements KeyListener {
    @Override
    public void init() {
        super.init();
        addKeyListener(this);
    }

    public void destroy() {
        super.destroy();
        removeKeyListener(this);
    }

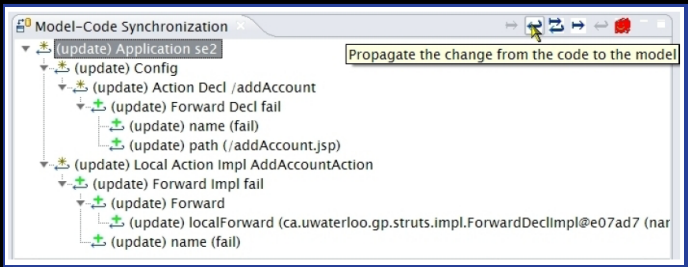
    public void keyTyped(KeyEvent keyEvent) {
    }

    public void keyPressed(KeyEvent keyEvent) {
    }

    public void keyReleased(KeyEvent keyEvent) {
    }
}
```

^o H. M. Lee. Model-guided Code Assistance for Framework Application Development.

Tool Support: Reverse Engineering



^o H. M. Lee. Model-guided Code Assistance for Framework Application Development.

Constructing FSMs

FSMs are great!

But, how are they constructed?

- Creator must have an in-depth understanding of the framework.
- Examine tutorials and documentation.
- Examine boiler-plate code and example applications.

...currently an entirely **manual** process.

Constructing FSMs

FSMs are great!

But, how are they constructed?

- Creator must have an in-depth understanding of the framework.
- Examine tutorials and documentation.
- Examine boiler-plate code and example applications.

...currently an entirely **manual** process.

Constructing FSMs

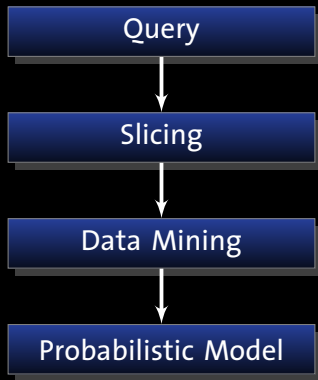
FSMs are great!

But, how are they constructed?

- Creator must have an in-depth understanding of the framework.
- Examine tutorials and documentation.
- Examine boiler-plate code and example applications.

...currently an entirely **manual** process.

Mining for Framework Concepts



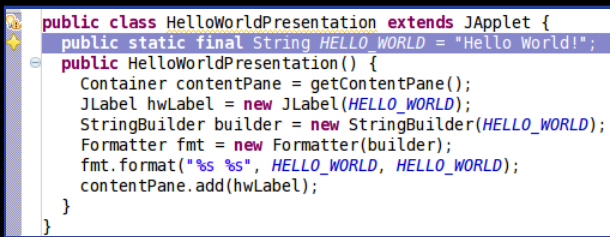
Provide hints for finding concept statements.

Find related statements.

Find usage patterns among multiple examples.

Output a probabilistic model describing usage.

Input Query

A screenshot of a Java code editor window. The code defines a class named HelloWorldPresentation that extends JApplet. It includes a static final String HELLO_WORLD with the value "Hello World!". The constructor initializes a JPanel, adds a JLabel with the HELLO_WORLD text, and formats the output. The code is as follows:

```
public class HelloWorldPresentation extends JApplet {  
    public static final String HELLO_WORLD = "Hello World!";  
    public HelloWorldPresentation() {  
        Container contentPane = getContentPane();  
        JLabel hwLabel = new JLabel(HELLO_WORLD);  
        StringBuilder builder = new StringBuilder(HELLO_WORLD);  
        Formatter fmt = new Formatter(builder);  
        fmt.format("%s %s", HELLO_WORLD, HELLO_WORLD);  
        contentPane.add(hwLabel);  
    }  
}
```

User selects lines of code in a framework application.

Concept Slicing

Reduce an example to a minimal set of statements that retains the specified behaviour.



Source code
with Query

Concept Slicing

Reduce an example to a minimal set of statements that retains the specified behaviour.



Source code
with Query



Program Slicer

Concept Slicing

Reduce an example to a minimal set of statements that retains the specified behaviour.



Source code
with Query



Program Slicer



Relevant
Statements

Concept Slicing

```
public class HelloApplet extends JApplet {  
    KeyListener keyList = new KeyListener() { ... }  
    Image smiley;  
  
    public void init() {  
        smiley = getImage(getCodeBase(), "images/smiley.gif");  
        showStatus("Hello, world!");  
        addKeyListener(keyList);  
        play(getCodeBase(), "audio/woohoo.wav");  
    }  
  
    public void destroy() {  
        removeKeyListener(keyList);  
    }  
}
```

Concept Slicing

```
public class HelloApplet extends JApplet {  
    KeyListener keyList = new KeyListener() { ... }  
    Image smiley;  
  
    public void init() {  
        smiley = getImage(getCodeBase(), "images/smiley.gif");  
        showStatus("Hello, world!");  
        addKeyListener(keyList);  
        play(getCodeBase(), "audio/woohoo.wav");  
    }  
  
    public void destroy() {  
        removeKeyListener(keyList);  
    }  
}
```


Sliced Statements

Example	Location	Statement
HelloApplet	Class	<i>field</i> :keyList (KeyListener)
HelloApplet	initializer	<i>assigns</i> :keyList = new KeyListener()
HelloApplet	init()	<i>calls</i> :addKeyListener(keyList)
HelloApplet	destroy()	<i>calls</i> :removeKeyListener(keyList)

Finding Related Usages

The screenshot shows an IDE window with a Java file named `Test.java`. The code defines a class `Test` that extends `JApplet` and implements the `init()` method. The `init()` method contains several lines of code, including creating an `ImageIcon`, creating three `JLabel` objects (`l1`, `l2`, and `l3`), and adding them to the content pane. The line `add(l2);` is highlighted.

Below the code editor, there is a table showing the locations where the selected code is used. The table has two columns: **Feature** and **Location**.

Feature	Location
HelloWorldAssignment.java	Call: cp2.add(hw)
Test.java	Call: add(l2)
Test.java	Call: add(l1)
HelloWorld2.java	Call: getContentPane().add(label)
HelloWorldPresentation.java	Call: contentPane.add(hwLabel)
HelloWorld3.java	Call: add(new JLabel("Hello World"))
Test.java	Call: add(l3)

Reverse-Engineering the Model

★ EVENTS₁

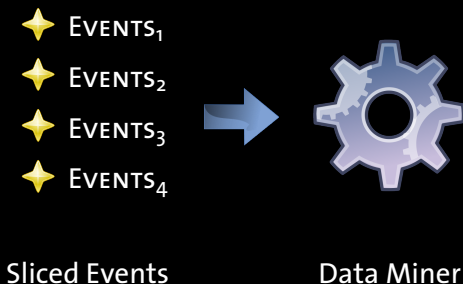
★ EVENTS₂

★ EVENTS₃

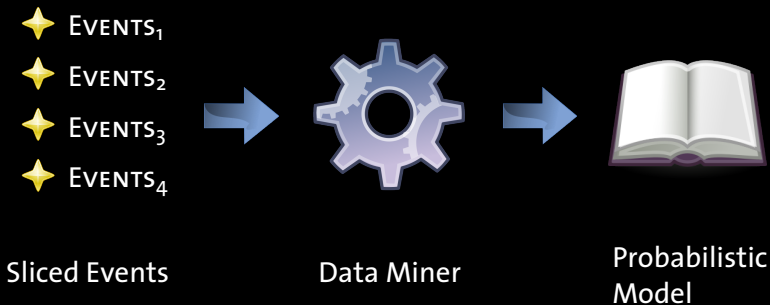
★ EVENTS₄

Sliced Events

Reverse-Engineering the Model



Reverse-Engineering the Model



Feature Model Mining

Mine for probabilistic expressions from slices gathered from multiple applications.

^oK. Czarnecki, A. Wasowski, S. She.

Feature Model Mining

Mine for probabilistic expressions from slices gathered from multiple applications.

Association Rule	conf
KeyListener $\Rightarrow$ asThis $\vee$ asAField	100%

^oK. Czarnecki, A. Wasowski, S. She.

Feature Model Mining

Mine for probabilistic expressions from slices gathered from multiple applications.

Association Rule	conf
KeyListener $\Rightarrow$ asThis $\vee$ asAField	100%
asAField $\Rightarrow \neg$ asThis	100%

^oK. Czarnecki, A. Wasowski, S. She.

Feature Model Mining

Mine for probabilistic expressions from slices gathered from multiple applications.

Association Rule	conf
KeyListener $\Rightarrow$ asThis $\vee$ asAField	100%
asAField $\Rightarrow \neg$ asThis	100%
KeyListener $\Rightarrow$ asAField	70%
KeyListener $\Rightarrow$ asThis	30%

^oK. Czarnecki, A. Wasowski, S. She.

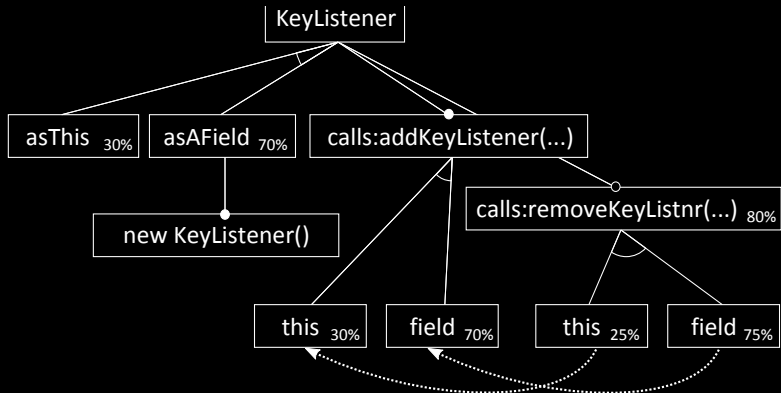
Feature Model Mining

Mine for probabilistic expressions from slices gathered from multiple applications.

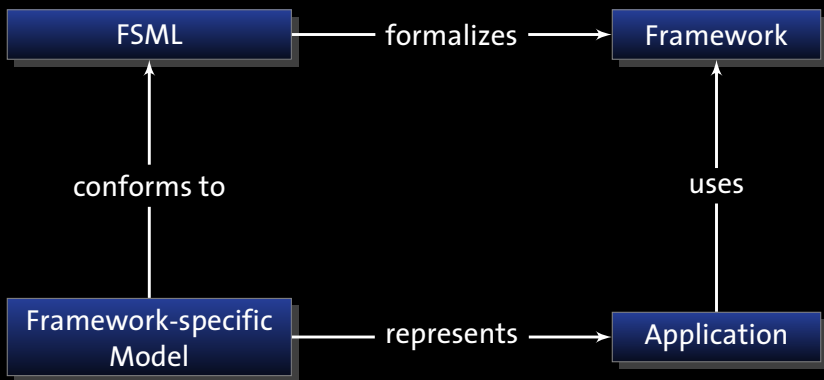
Association Rule	conf
KeyListener $\Rightarrow$ asThis $\vee$ asAField	100%
asAField $\Rightarrow \neg$ asThis	100%
KeyListener $\Rightarrow$ asAField	70%
KeyListener $\Rightarrow$ asThis	30%
asAField $\Leftrightarrow$ new KeyListener()	100%

^oK. Czarnecki, A. Wasowski, S. She.

Mined Feature Model

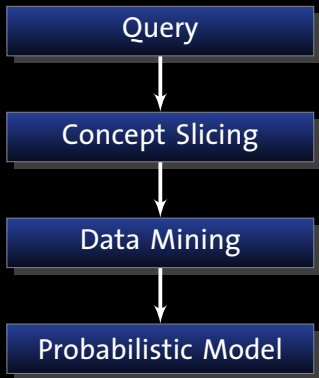


Framework-Specific Modelling Languages



^o M. Antkiewicz, T. Tonelli Bartolomei, K. Czarnecki

Conclusions



- Constructs a **recipe** from a set of examples.
- **Slicing** finds related statements given a query.
- **Data mining** finds patterns in sliced statements to form a probabilistic model.