Marea
A Tool for Breaking Dependency Cycles Between Packages

Master Project
Bledar Aga

Software Composition Group
University of Bern
Fall 2014
Dependency Cycles
Problems

- Acyclic Design Principle
- Maintenance and Testing costs
- Modularity and Reuse
Dependency Types

- Reference
- Inheritance
- Invocation

Refactoring Methods

- Move Class (MC)
- Move Method (MM)
- Using design patterns:
  - Abstract Server Pattern (ASP)
  - Dependency Injection (DI)
ASPS Example

- Control
  - Light is not initialised

- Components
  - Button
    - light: Light
    - press()
  - Light
    - turnOnOff()
ASP Example

control

<<interface>>

Switchable

turnOnOff()

Button

light:Switchable

press()

components

Light

turnOnOff()
ASP + DI Example

Control

Switchable
<<interface>>
turnOnOff()

Button
light:Switchable
setLight(light:Switchable)
press()

Light
turnOnOff()

components

Injection of light by setter method

Switchable l = new Light();
Button b = new Button();
b.setLight(l);

Inject
injection
## Applicability of Refactoring

<table>
<thead>
<tr>
<th></th>
<th>Move Class</th>
<th>Move Method</th>
<th>ASP</th>
<th>ASP+DI</th>
</tr>
</thead>
<tbody>
<tr>
<td>Inheritance</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Class Variable</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Initialised Class Variable</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Local Variable</td>
<td>Yes</td>
<td>Yes (Limited)</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Initialised Local Variable</td>
<td>Yes</td>
<td>Yes (Limited)</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Parameter</td>
<td>Yes</td>
<td>Yes (Limited)</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Return Type</td>
<td>Yes</td>
<td>Yes (Limited)</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Invocation</td>
<td>Yes</td>
<td>Yes (Limited)</td>
<td>Yes (Limited)</td>
<td>No</td>
</tr>
</tbody>
</table>
## Existing Tools

<table>
<thead>
<tr>
<th></th>
<th>Detection</th>
<th>Removal</th>
</tr>
</thead>
<tbody>
<tr>
<td>JooJ</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Jepends</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Stan</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Pasta</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Lattix</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Structure101</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>ECOO</td>
<td>Yes</td>
<td>Yes</td>
</tr>
</tbody>
</table>
Our Solution

1. Analysis
   - **logical dep.**

2. Simulation
   - **decision tree**

3. Define Refactoring Chain
   - **decision trees**

4. Apply Refactoring
Our Solution

- Detect cycles
- Propose dependency to break
Cycle Detection

Project

SCC

A

B

C

D

E

Single Cycles

A

B

E

A

B

C

D

C

D
Dependencies

Sort Logical Dependencies by: % Shared Dependencies
Our Solution

- Simulate refactorings
- Propose best refactoring sequence
Decisional Tree

\[ P = w_c \times \frac{1}{\text{# cycles} + 1} + w_d \times \frac{1}{\text{depth}} + w_i \times \frac{(1 - I_{from}) + (1 - I_{to})}{2} + w_a \times \frac{(A_{from}) + (A_{to})}{2} \]
Our Solution

- Choice of the best path
- Repeat until no more cycles
DEMO
Case study su GWT Calendar

7 cycles 5 refactoring actions 0 cycles
4 iterations

University of Bern, SCG
Case study su JHotDraw

stability + abstractness

stability
Integration with Dicto
Future Work

- User interface interaction
- Export the decisional trees in a file
- Automate the refactoring decision integrating with Eclipse
Summary

 Dependency cycles difficult topic

 Without a proper tool support engineers cannot solve the problem

 Marea assistes engineers in resolving dependency cycles

 Smalltalkhub: BledarAga/Marea