A Preliminary Study of Quantified, Typed Events

Robert Dyer\textsuperscript{1}, Mehdi Bagherzadeh\textsuperscript{1}, Hridesh Rajan\textsuperscript{1} and Yuanfang Cai\textsuperscript{2}

\textsuperscript{1}Iowa State University  
\{rdyer, mbagherz, hridesh\}@cs.iastate.edu  
\textsuperscript{2}Drexel University  
yfcai@cs.drexel.edu

March 16, 2010
Ptolemy: What, Why?

- Ptolemy\(^1\) adds quantified, typed events to OO languages
  - 1. Well-defined interfaces between base & crosscutting code
  - 2. Separate type-checking, modular reasoning
- Combines aspect-oriented (AO) and implicit invocation (II)
- Solves problems with AO and II:
  - AO: quantification failure, fragile pointcuts, limited context information
  - II: coupling of observers, no replacement of event code, no quantification

\(^1\)Rajan and Leavens - ECOOP’08
This Paper: Why, How, and What?

- Motivation: Why use Quantified, Typed Events?²
- Approach: MobileMedia case study³
- Evaluation: Change impact and Design value analysis
  - Software engineering metrics: makes implicit coupling in AO explicit and decreases change impact
  - NOV analysis: Ptolemy needs ITDs (so we added it)

²Rajan and Leavens - ECOOP’08
³Figueiredo et al - ICSE’08
Elements of drawing

- Points, Lines, etc
- All such elements are of type \texttt{FElement}

**Challenge:** Modularize display update policy

- Whenever an element of drawing changes —
- Update the display
Point and its two Events

class Point implements FElement {
    int x; int y;
    void setX(int x) {
        this.x = x;
    }
    ..
    void makeEqual(Point other) {
        if (!other.equals(this)) {
            other.x = this.x;
            other.y = this.y;
        }
    }
}

- Changing FElement is different for two cases.
- Actual abstract event inside makeEqual is the true branch.
aspect Update { 
    around (FElement fe) : 
    execution (Point.set*(..)) && this(fe) || 
        (execution (Point.make*(..)) && args(fe) 
        if (!fe.equals(this(fe)))) { 
            proceed(fe); 
            Display.update(); 
    } 
} 

- Enumeration required of two different joinpoints. 
- Had to use if pointcut to get to the real event. 
- Alternative is to refactor makeEqual (refactoring doesn’t always creates meaningful abstractions).
Ptolemy: Declaring Event Types

```java
void event FEChanged {
    FElement changedFE;
}
```

- Event type is an abstraction (design this first).
- Declares context available at the concrete events.
- Interface, so allows design by contract (DBC) methodology.
Explicit, declarative, typed event announcement.

Provides flexibility, e.g. see `makeEqual`.

```java
class Point implements FElement {
  int x; int y;
  void setX(int x) {
    announce FEChanged(this) {
      this.x = x;
    }
  }
  void makeEqual(Point other) {
    if(!other.equals(this)) {
      announce FEChanged(other) {
        other.x = this.x; other.y = this.y;
      }
    }
  }
}
```
Ptolemy: Binding to Events

class Update {
    when FEChanged do update;
    void update (FEChanged next) {
        invoke(next); //Like AspectJ proceed
        Display.update();
    }
    public Update() {
        register(this); //Allows dynamic deployment
    }
}
Research Questions

- How do these two designs compare?
- When do we see benefits of AO?
- When do we see benefits of Ptolemy?
Observed Benefits of Aspect-oriented Designs

- Static crosscutting features are very useful
  - Inter-type declarations (ITDs)
  - Declare Parents
  - Softened Exceptions
Inter-type declarations (ITDs)

- Had to be emulated in Ptolemy\(^4\)
- AspectJ:
  
  \[
  \text{public } T \text{ C.field;}
  \]

- Ptolemy emulation strategy:
  
  \[
  \text{static } \text{Hashtable fieldMap;}
  \]
  
  \[
  \text{public } \text{static } T \text{ getField(C);} \\
  \text{public } \text{static } \text{void } \text{setField(C, T);} \\
  \]

\(^4\)Ptolemy now supports AspectJ-style ITDs
Declare Parents

- Affects the type hierarchy
- Only used in revision 8
- Effects modeled similar to ITDs
Softened Exceptions

- Aspects handle certain exceptions
- Softened exceptions don’t need declared thrown in the base code
- Con: Ptolemy version still must declare those exceptions are thrown
- Pro: Ptolemy’s exception handling code is (un)pluggable
Softened Exceptions

AspectJ version:

```java
declare soft: RecordStoreEx :
  execution(public void
    ImageAccessor.addImageData(.));

public void addImageData(..) throws
  InvalidImageDataEx, PersistenceMechanismEx {

Without the aspect, the base code won’t compile!
```
Softened Exceptions

Ptolemy version:

```java
public void addImageData(..) throws
    InvalidImageDataEx, PersistenceMechanismEx, RecordStoreEx {
```

Even though RecordStoreEx isn’t thrown by the body, it still must be declared!
Observed Benefits of Quantified, Typed Events
- No Quantification Failure
- No Fragile Pointcuts
- Can easily advise other advice (due to symmetry)
Solves Quantification Failure Problem

- In revision 2, the AspectJ version had to expose a `while` loop (by refactoring of course)
- Similar problems in other revisions
- Ptolemy versions did not need to refactor those points to expose to the aspects
Example of Quantification Failure in AO

OO version:

```
.. while (is.read(b)) { .. } ..
```

AspectJ version:

```
.. internalReadImage(..);
..

private void internalReadImage(..) {
    while (is.read(b)) { .. }
}
```
No Quantification Failure

**OO version:**

```
.. 
while (is.read(b)) { .. } 
.. 
```

**Ptolemy version:**

```
.. 
announce ReadInternalImageAsByteArrayEvent() { 
    while (is.read(b)) { .. } 
} 
.. 
```
No Fragile Pointcuts

- Aspects implicitly match the base code
- Quantified, typed events make this coupling explicit
- Changes to the base code (e.g., renaming a method) can propagate to the aspects

- **AspectJ**: `execution(* DeletePhoto(..))`
- **Ptolemy**: `when DeletePhotoEvent do Handler`
- **What if** `DeletePhoto` is renamed to `RemovePhoto`?
Can easily advise other advice

- AspectJ allows you to advise all advice, not specific advice bodies

- Exception handling modularity was not maintained in later revisions for AspectJ versions

- Ex: revision 8, aspect lancs.midp.mobilephoto.alternative.music.MusicAspect
after() : addNewMediaToAlbum() {
    try {
        /* advice body */
    } catch (InvalidImageDataException e) {
        ..
    }
}

public void handler(AddMediaToAlbumEvent next) {
    announce AddNewMediaToAlbumHandlerEvent() {
        /* advice body */
    }
}
Overview of Change Impact AO vs. Ptolemy

- Ptolemy design limited change propagation
- Only had to change 13 event types in revision 7
- AO revisions required changing 50 pointcuts in revision 7
- No other Ptolemy revision required changing event types
- 28 pointcuts changed across 4 other AO revisions
Net Options Value Analysis AO vs. Ptolemy

Dyer, Bagherzadeh, Rajan and Cai

Preliminary Study of Quantified, Typed Events
Ptolemy: higher NOV values for R2-R6
AO: higher value in R6 (Ptolemy doesn’t have ITDs)
Revisions adding ITDs: R3-R8
Summary of Study and Future Work

- Explicit Coupling increases from OO → AO → Ptolemy
- Despite more coupling in Ptolemy, lower change impact
- Net options value increases from OO → AO → Ptolemy

Future Work:
- Repeat the study with Ptolemy + ITDs
- Compare to other AO interface features
Questions?

http://www.cs.iastate.edu/~ptolemy/
http://ptolemyj.sourceforge.net
<table>
<thead>
<tr>
<th>Release</th>
<th>Description</th>
<th>Type of Change</th>
</tr>
</thead>
<tbody>
<tr>
<td>R1</td>
<td>MobilePhoto core</td>
<td></td>
</tr>
<tr>
<td>R2</td>
<td>Exception handling included</td>
<td>Non-functional concern</td>
</tr>
<tr>
<td>R3</td>
<td>Added photo sorting</td>
<td>Optional feature</td>
</tr>
<tr>
<td></td>
<td>Added editing photo labels</td>
<td>Mandatory feature</td>
</tr>
<tr>
<td>R4</td>
<td>Added favorites</td>
<td>Optional feature</td>
</tr>
<tr>
<td>R5</td>
<td>Allow users to keep multiple copies of photos</td>
<td>Optional feature</td>
</tr>
<tr>
<td>R6</td>
<td>Added send photo to other users by SMS</td>
<td>Optional feature</td>
</tr>
<tr>
<td>R7</td>
<td>Photo management made into two alternatives: photo or music</td>
<td>One mandatory feature</td>
</tr>
<tr>
<td></td>
<td></td>
<td>into two alternatives</td>
</tr>
<tr>
<td>R8</td>
<td>Add video management</td>
<td>Alternative feature</td>
</tr>
<tr>
<td>Components</td>
<td>Added</td>
<td>Removed</td>
</tr>
<tr>
<td>------------</td>
<td>-------</td>
<td>---------</td>
</tr>
<tr>
<td>R2 R3 R4 R5 R6 R7 R8</td>
<td></td>
<td></td>
</tr>
<tr>
<td>OO</td>
<td>9</td>
<td>0</td>
</tr>
<tr>
<td>AO</td>
<td>12</td>
<td>0</td>
</tr>
<tr>
<td>PTL</td>
<td>13</td>
<td>0</td>
</tr>
<tr>
<td>OO</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>AO</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>PTL</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>OO</td>
<td>5</td>
<td>8</td>
</tr>
<tr>
<td>AO</td>
<td>5</td>
<td>10</td>
</tr>
<tr>
<td>PTL</td>
<td>11</td>
<td>8</td>
</tr>
<tr>
<td>Added</td>
<td>AO</td>
<td>43</td>
</tr>
<tr>
<td>Removed</td>
<td>AO</td>
<td>0</td>
</tr>
<tr>
<td>Changed</td>
<td>AO</td>
<td>0</td>
</tr>
<tr>
<td>Added</td>
<td>PTL</td>
<td>25</td>
</tr>
<tr>
<td>Removed</td>
<td>PTL</td>
<td>0</td>
</tr>
<tr>
<td>Changed</td>
<td>PTL</td>
<td>0</td>
</tr>
</tbody>
</table>

Event Types

<p>| Dyer, Bagherzadeh, Rajan and Cai | Preliminary Study of Quantified, Typed Events | 31 |</p>
<table>
<thead>
<tr>
<th></th>
<th>LCOO</th>
<th>R2</th>
<th>R3</th>
<th>R4</th>
<th>R5</th>
<th>R6</th>
<th>R7</th>
<th>R8</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Average</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>OO</td>
<td>6.38</td>
<td>10.08</td>
<td>10.88</td>
<td>8.83</td>
<td>10.24</td>
<td>12.04</td>
<td>11.80</td>
<td></td>
</tr>
<tr>
<td>AO</td>
<td>5.67</td>
<td>8.69</td>
<td>8.50</td>
<td>6.97</td>
<td>8.24</td>
<td>9.39</td>
<td>8.03</td>
<td></td>
</tr>
<tr>
<td>PTL</td>
<td>3.30</td>
<td>3.36</td>
<td>3.21</td>
<td>2.86</td>
<td>2.83</td>
<td>2.77</td>
<td>2.96</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Max</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>OO</td>
<td>64</td>
<td>70</td>
<td>71</td>
<td>73</td>
<td>96</td>
<td>113</td>
<td>114</td>
<td></td>
</tr>
<tr>
<td>AO</td>
<td>64</td>
<td>94</td>
<td>94</td>
<td>64</td>
<td>85</td>
<td>109</td>
<td>111</td>
<td></td>
</tr>
<tr>
<td>PTL</td>
<td>84</td>
<td>122</td>
<td>122</td>
<td>66</td>
<td>66</td>
<td>112</td>
<td>153</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Average</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>OO</td>
<td>1.46</td>
<td>2.00</td>
<td>2.36</td>
<td>3.17</td>
<td>3.30</td>
<td>3.54</td>
<td>3.96</td>
<td></td>
</tr>
<tr>
<td>AO</td>
<td>1.30</td>
<td>1.72</td>
<td>1.84</td>
<td>2.50</td>
<td>2.65</td>
<td>2.76</td>
<td>2.69</td>
<td></td>
</tr>
<tr>
<td>PTL</td>
<td>0.80</td>
<td>0.94</td>
<td>1.06</td>
<td>1.43</td>
<td>1.57</td>
<td>1.71</td>
<td>2.01</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Max</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>OO</td>
<td>9</td>
<td>13</td>
<td>13</td>
<td>11</td>
<td>11</td>
<td>15</td>
<td>20</td>
<td></td>
</tr>
<tr>
<td>AO</td>
<td>9</td>
<td>13</td>
<td>12</td>
<td>10</td>
<td>13</td>
<td>14</td>
<td>14</td>
<td></td>
</tr>
<tr>
<td>PTL</td>
<td>9</td>
<td>14</td>
<td>14</td>
<td>12</td>
<td>13</td>
<td>14</td>
<td>14</td>
<td></td>
</tr>
<tr>
<td>LOC</td>
<td>R2</td>
<td>R3</td>
<td>R4</td>
<td>R5</td>
<td>R6</td>
<td>R7</td>
<td>R8</td>
<td></td>
</tr>
<tr>
<td>-----</td>
<td>-----</td>
<td>-----</td>
<td>-----</td>
<td>-----</td>
<td>-----</td>
<td>-----</td>
<td>-----</td>
<td></td>
</tr>
<tr>
<td>OO</td>
<td>1159</td>
<td>1314</td>
<td>1363</td>
<td>1555</td>
<td>2051</td>
<td>2523</td>
<td>3016</td>
<td></td>
</tr>
<tr>
<td>AO</td>
<td>1276</td>
<td>1494</td>
<td>1613</td>
<td>1834</td>
<td>2364</td>
<td>3068</td>
<td>3806</td>
<td></td>
</tr>
<tr>
<td>PTL</td>
<td>1605</td>
<td>1923</td>
<td>2049</td>
<td>2374</td>
<td>2969</td>
<td>3655</td>
<td>4508</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>NOC</th>
<th>R2</th>
<th>R3</th>
<th>R4</th>
<th>R5</th>
<th>R6</th>
<th>R7</th>
<th>R8</th>
</tr>
</thead>
<tbody>
<tr>
<td>OO</td>
<td>24</td>
<td>25</td>
<td>25</td>
<td>30</td>
<td>37</td>
<td>46</td>
<td>51</td>
</tr>
<tr>
<td>AO</td>
<td>27</td>
<td>29</td>
<td>32</td>
<td>38</td>
<td>46</td>
<td>59</td>
<td>75</td>
</tr>
<tr>
<td>PTL</td>
<td>56</td>
<td>67</td>
<td>70</td>
<td>79</td>
<td>92</td>
<td>112</td>
<td>132</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>NOA</th>
<th>R2</th>
<th>R3</th>
<th>R4</th>
<th>R5</th>
<th>R6</th>
<th>R7</th>
<th>R8</th>
</tr>
</thead>
<tbody>
<tr>
<td>OO</td>
<td>62</td>
<td>71</td>
<td>74</td>
<td>75</td>
<td>106</td>
<td>132</td>
<td>165</td>
</tr>
<tr>
<td>AO</td>
<td>62</td>
<td>72</td>
<td>76</td>
<td>77</td>
<td>111</td>
<td>139</td>
<td>177</td>
</tr>
<tr>
<td>PTL</td>
<td>72</td>
<td>82</td>
<td>86</td>
<td>88</td>
<td>121</td>
<td>146</td>
<td>185</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>NOO</th>
<th>R2</th>
<th>R3</th>
<th>R4</th>
<th>R5</th>
<th>R6</th>
<th>R7</th>
<th>R8</th>
</tr>
</thead>
<tbody>
<tr>
<td>OO</td>
<td>124</td>
<td>140</td>
<td>143</td>
<td>160</td>
<td>200</td>
<td>239</td>
<td>271</td>
</tr>
<tr>
<td>AO</td>
<td>158</td>
<td>187</td>
<td>199</td>
<td>230</td>
<td>285</td>
<td>345</td>
<td>441</td>
</tr>
<tr>
<td>PTL</td>
<td>143</td>
<td>169</td>
<td>179</td>
<td>197</td>
<td>247</td>
<td>308</td>
<td>369</td>
</tr>
</tbody>
</table>
1 Eduardo Figueiredo, Nelio Cacho, Claudio Sant’Anna, Mario Monteiro, Uira Kulesza, Alessandro Garcia, Sergio Soares, Fabiano Ferrari, Safoora Khan, Fernando Castor Filho and Francisco Dantas. Evolving software product lines with aspects: an empirical study on design stability. In ICSE ’08.
