Motivation

Many metrics for separation of concerns have been proposed
• We call Concern Metrics

Concern Metrics rely on the projection of concerns onto software artifacts
• Source code in this study

Other Concern Metrics

Eaddy, Aho, and Murphy [4]
• Lines of Concern Code (LOCC)
• Concentration and Dedication
Lopez-Herrejon and Apel [5]
• Number of Features (NOF)
• Feature Crosscutting Degree (FCD)
NOCA, NCC, …

Quantifying Crosscutting

Some Concern Metrics

Who proposed them?
Sant’Anna [2]
• Concern Diffusion over Components (CDC)
• Concern Diffusion over Operations (CDO)
• Concern Diffusion over Lines of Code (CDLOC)
Ducasse, Girba, and Kuhn [3]
• Size, Touch, Spread, and Focus

Agenda

Metrics for Separation of Concerns
Study Settings
• Experiment Hypotheses
• The Participants
• Evaluation Procedures
Results and Key Findings
Ongoing and Future Work

On the Impact of Crosscutting Concern Projection on Code Measurement
Eduardo Figueiredo (UFMG), Alessandro Garcia (PUC-Rio), Marcelo Maia (UFU), Gabriel Ferreira (UFU), Camila Nunes (PUC-Rio) and Jon Whittle (Lancaster University)
23 March 2011

Source code of the

Evolving software product lines with aspects: an empirical study

MediaController class

if(label.equals("Password Given")){
...
}

otherConcernMetrics

if(password == null) {
...
}

quantifyingCrosscutting

Some Concern Metrics

• Concern Diffusion over Lines of Code (CDLOC)

Ongoing and Future Work

Study Settings

• Experiment Hypotheses
• The Participants
• Evaluation Procedures

Ongoing and Future Work

Metric for Separation of Concerns

• Concern Diffusion over Lines of Code (CDLOC)
• Features Crosscutting Degree (FCD)

Who proposed them?

Sant’Anna [2]
• Concern Diffusion over Components (CDC)
• Concern Diffusion over Operations (CDO)
• Concern Diffusion over Lines of Code (CDLOC)

Ducasse, Girba, and Kuhn [3]
• Size, Touch, Spread, and Focus
public boolean handleCommand(Command command) {
  String label = command.getLabel();
  if (label.equals("Select Album")) {
    String password;
    if(password == null) {
      password = getAlbumData().getPassword(getCurrentAlbumName());
      setCurrentAlbumName(getDisplay(midlet).getSelectedIndex());
    }
    else {
      if(pwScreen.getPassword().equals(password)) {
        password = getAlbumData().getPassword(getCurrentAlbumName());
        PasswordScreen pwScreen = (PasswordScreen) getCurrentScreen();
        return true;
      }
    }
    return true;
  }
  return false;
}

public class MediaController extends AbstractController {
  ...
  ...
  public void showMediaList(String albumName, boolean sort, boolean favorite) {
    ...
    super(midlet, albumData, albumListScreen);
  }
  ...
  ...
  public boolean handleCommand(Command command) {
    ...
    super(midlet, albumData, albumListScreen);
  }
  ...
  ...
  public class MediaController extends AbstractController {
    ...
    ...
    public void showMediaList(String albumName, boolean sort, boolean favorite) {
      ...
      super(midlet, albumData, albumListScreen);
    }
    ...
    ...
    public boolean handleCommand(Command command) {
      ...
      super(midlet, albumData, albumListScreen);
    }
    ...
    ...
    public class MediaController extends AbstractController {
      ...
      ...
      public void showMediaList(String albumName, boolean sort, boolean favorite) {
        ...
        super(midlet, albumData, albumListScreen);
      }
      ...
      ...
      public boolean handleCommand(Command command) {
        ...
        super(midlet, albumData, albumListScreen);
      }
      ...
      ...
    }
  }
  ...
  ...
  public class MediaController extends AbstractController {
    ...
    ...
    public void showMediaList(String albumName, boolean sort, boolean favorite) {
      ...
      super(midlet, albumData, albumListScreen);
    }
    ...
    ...
    public boolean handleCommand(Command command) {
      ...
      super(midlet, albumData, albumListScreen);
    }
    ...
    ...
  }
  ...
  ...
  public class MediaController extends AbstractController {
    ...
    ...
    public void showMediaList(String albumName, boolean sort, boolean favorite) {
      ...
      super(midlet, albumData, albumListScreen);
    }
    ...
    ...
    public boolean handleCommand(Command command) {
      ...
      super(midlet, albumData, albumListScreen);
    }
    ...
    ...
  }
}

Research Question / Hypothesis

Research Question (RQ1)
Can different developers identify the same code fragments for a concern?

Hypothesis (H1)
The projection of crosscutting concerns into source code does not depend on individual differences between developers

Measures for Security

Security

CDC = 1
CDO = 1
LOCC = 20

The Selected Concerns

We selected 10 concerns from two applications
- 6 concerns from Health Watcher [6]
- 4 concerns from MobileMedia

Eight crosscutting concerns
- Concurrency, Distribution, Persistence, Exception Handling (2x), Security, Sorting, and Favourites

Two non-crosscutting concerns
- Business and View (GUI)

Study Settings

The Selected Concerns
Participants and Institutions
Experimental Procedures
Participants per Replication

5 replications with 80 participants from 4 institutions

1st Replication (FRB - Health Watcher)
- 6 interns and young developers with less than 3 years experience in software development

2nd Replication (Lancs - Health Watcher)
- 13 undergraduate Computer Science students

3rd Replication (PUC-Rio - Health Watcher)
- 16 graduated Master and PhD students
- Organized in groups of two or three people

4th Replication (PUC-Rio - MobileMedia)
- 16 graduated Master and PhD students
- None of them participated in the 3rd replication

5th Replication (UFMG - MobileMedia)
- 32 undergraduate Computer Science students
- Organized in groups of two or three people

Experimental Procedures

1. Each replication was limited by 1.5 hour
   - Including a 15-minute training session
2. We give the participants
   - The source code of four classes
   - The description of the concerns
3. We asked them to project (by hand) the concerns onto the given code
4. After each experiment, we measure the rate of hits, false positives and false negatives

Measure of False Positives

<table>
<thead>
<tr>
<th>Concern Name</th>
<th>Hits</th>
<th>False Positives</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>FP%</td>
<td># False Positives</td>
</tr>
<tr>
<td></td>
<td></td>
<td># LOC not implementing concern</td>
</tr>
<tr>
<td></td>
<td></td>
<td># False Positives = 1</td>
</tr>
<tr>
<td></td>
<td></td>
<td># LOC not implementing concern = 8</td>
</tr>
<tr>
<td></td>
<td></td>
<td>FP% = 12.5%</td>
</tr>
</tbody>
</table>

Measure of False Negatives

<table>
<thead>
<tr>
<th>Concern Name</th>
<th>Hits</th>
<th>False Negatives</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>FN%</td>
<td># False Negatives</td>
</tr>
<tr>
<td></td>
<td></td>
<td># Concern LOC</td>
</tr>
<tr>
<td></td>
<td></td>
<td># False Negatives = 1</td>
</tr>
<tr>
<td></td>
<td></td>
<td># Concern LOC = 2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>FN% = 50%</td>
</tr>
</tbody>
</table>

Measure of Hits

<table>
<thead>
<tr>
<th>Concern Name</th>
<th>Hits</th>
<th>False Positives</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Hits%</td>
<td>Correctly Tagged</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Correctly Not Tagged</td>
</tr>
<tr>
<td></td>
<td></td>
<td># Lines of Code</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Correctly Tagged = 2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Correctly Not Tagged = 8</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Hits% = 100%</td>
</tr>
</tbody>
</table>
Results of the 1st Replication

Most participants scored more than 80% of Hits in 5 out of 6 concerns.

The Issue of Business

Table 1. Comparison of Business to other concerns (FRB)

<table>
<thead>
<tr>
<th></th>
<th>S1</th>
<th>S2</th>
<th>S3*</th>
<th>S4</th>
<th>S5</th>
<th>S6</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hits</td>
<td>60.5%</td>
<td>59.8%</td>
<td>55.2%</td>
<td>79.4%</td>
<td>45.5%</td>
<td>71.3%</td>
</tr>
<tr>
<td>False Positives</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>4.9%</td>
<td>0%</td>
<td>0%</td>
</tr>
<tr>
<td>False Negatives</td>
<td>25.1%</td>
<td>56.1%</td>
<td>62.4%</td>
<td>26.8%</td>
<td>76.1%</td>
<td>43.1%</td>
</tr>
<tr>
<td>Average Others</td>
<td>46.6%</td>
<td>93.4%</td>
<td>93.4%</td>
<td>95.2%</td>
<td>91.0%</td>
<td>94.5%</td>
</tr>
<tr>
<td>False Positives</td>
<td>9.7%</td>
<td>1.0%</td>
<td>2.5%</td>
<td>1.2%</td>
<td>2.9%</td>
<td>1.0%</td>
</tr>
<tr>
<td>False Negatives</td>
<td>44.1%</td>
<td>39.0%</td>
<td>35.6%</td>
<td>29.9%</td>
<td>50.8%</td>
<td>36.7%</td>
</tr>
</tbody>
</table>

Business is also an issue in the measurement of False Negatives.

Results of the 2nd Replication

Due to time constraints we remove Business from the 2nd study replication.

Results of the 3rd Replication

The participants were organized in groups of two or three people.

All participants scored more than 80% of Hits in the 5 remaining concerns.

Again, all participants scored more than 80% of Hits in all concerns.
Results of the 3rd Replication

The participants were organized in groups of two or three people.

![Measure of Hits (PUC-Rio)]

Groups of three participants do not perform as well as groups of two.

4th and 5th Replications

Confirmed previous observations:
All participants scored more than 80% of Hits for all concerns.

![Measure of Hits (PUC-Rio)]

General Observations

In general, participants score about the same for every concern:
- All participants scored more than 95% for Concurrency.
- Most subjects scored between 80% and 90% for Distribution.

Distribution seems the hardest crosscutting concern to be projected.

The measure of false positives was generally low:
- < 10% for all crosscutting concerns.

The measure of false negatives was generally high:
- > 20% for all crosscutting concerns.

Hypothesis (H1):
The projection of crosscutting concerns into source code does not depend on individual differences between developers.

Additional Finding (1):
Developers usually do not assign a line of code to a concern if they are unsure about it.

Additional Finding (2):
Developers usually miss out code fragments that are realizing a concern.

In general, subjects score more or less the same for every concern:
- All subjects scored more than 95% for Concurrency.
- Most subjects scored between 80% and 90% for Distribution.

Distribution seems the hardest crosscutting concern to be projected.

The measure of false positives was generally low:
- < 10% for all crosscutting concerns.

The measure of false negatives was generally high:
- > 20% for all crosscutting concerns.
The Impact on Concern Metrics

### Procedures

**Hypothesis**

Concern metrics can be precisely quantified regardless of who projected the crosscutting concern into the system.

1. Apply a set of metrics into concern projections from all participants
2. Verify the variance of a given metric across projections from different participants

### Average (and Reference Value)

<table>
<thead>
<tr>
<th>Concern</th>
<th>CDC</th>
<th>CDIO</th>
<th>UBDDL</th>
<th>LOC</th>
<th>NOCA</th>
</tr>
</thead>
<tbody>
<tr>
<td>Business</td>
<td>3 (30)</td>
<td>24 (12)</td>
<td>65 (265)</td>
<td>3 (7)</td>
<td></td>
</tr>
<tr>
<td>Concurrency</td>
<td>2 (2)</td>
<td>8 (10)</td>
<td>6.5</td>
<td>0.8 (1)</td>
<td></td>
</tr>
<tr>
<td>Distribution</td>
<td>2 (2)</td>
<td>21 (42)</td>
<td>0.5 (3)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>EH</td>
<td>12</td>
<td>25 (14)</td>
<td>0.4 (14)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Persistence</td>
<td>2 (2)</td>
<td>14 (6)</td>
<td>1.4 (2)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>View</td>
<td>1 (1)</td>
<td>10 (38)</td>
<td>0.4 (0)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

For some metrics, the average matches the reference value.

But, this is not the case for other metrics.

### Relative Error per Concern Metric

- CDLOC presents the largest relative error variation.
- CDC and NOCA present the smallest relative error variation.

### Study Constraints

The concern metrics were not automated:

- Hand-count measurement might be error prone.
- We have not taken any special care to select the subjects and institutions.
  - We consider random choices.
- The selected systems may not be representative of the industrial practice:
  - Two systems from different domains.
  - Heavily based on industry-strength technologies.

### Conclusions

This study aimed to quantify the impact of concern projections on measurement of concern properties. The selected crosscutting concerns could be projected with more than 80% precision.

The conservative behaviour makes developers miss code where concerns should be projected.

CDLOC can be considered unreliable since its values highly vary across projections.

CDC and NOCA metrics were found to be the most reliable ones.
Ongoing and Future Work

Additional concerns
- Our results are limited to 10 concern instances

Different systems and domains
- We rely on the source code of only two systems

Other research questions
- What recurring mistakes developers make when projecting crosscutting concerns?
- What kind of code fragments developers usually consider relevant to a crosscutting concern?

On the Impact of Crosscutting Concern Projection on Code Measurement

Eduardo Figueiredo (UFMG), Alessandro Garcia (PUC-Rio), Marcelo Maia (UFU), Gabriel Ferreira (UFU), Camila Nunes (PUC-Rio) and Jon Whittle (Lancaster University)

23 March 2011