Quantifying the Encapsulation of Implemented Software Architectures

@ericbouwers
@avandeursen
@jstvssr
an *implemented* software architecture
What is encapsulation?

“When applied correctly, the process of encapsulation ensures that the design decisions that are likely to change are localized”

Encapsulation == local changes

Component 1

Change-set A

Component 2

Change-set B

Local

Non local
Historical encapsulation

System 1:

Local change-set
Non-local change-set

System A:

Local change-set
Non-local change-set

0.7
0.4
Problem?

You need to have the change-sets!
Solution: find related snapshot metric!

System 1, 2, ... N:

Correlation?

# components: 7
# dependencies: 8

....

....
What to correlate?

# of components vs. Time

Change-set series 1

Change-set series 2
Experimental design

Select metrics → Select systems

Metric 1 → Determine stable periods → Calculate historic encapsulation → Correlate

Metric 2 → Determine stable periods → Calculate historic encapsulation → Correlate

Metric N → Determine stable periods → Calculate historic encapsulation → Correlate
Selected metrics

Ratio of Cohesive Interactions (RCI)
Cumulative Component dependency (CCD)
Average CCD (ACD)
Normalized CCD (NCD)
Cyclic Dependency Index (CDI)
Inbound Code (IBC)
Outbound Code (OBC)
Internal Code (IC)
Number of Binary Dependencies (NBD)

Component Balance (CB)
Module Size Uniformity (MSUI)
Number of Components (NC)
Selected systems

- Apache Ant
- Apache Beehive Project
- JasperReports
- Lucene
- FindBugs
- Subversion
- Top-level package = component

1+ year development
Number of periods per metric

- RCI
- CCD
- ACD
- NCD
- CDI
- IBC
- OBC
- IC
- NBD
- CB
- MSUI
- NC

Median length: W months
## What are normal values?

<table>
<thead>
<tr>
<th>Metric</th>
<th>Median length (months)</th>
<th>Median ratio of local change</th>
</tr>
</thead>
<tbody>
<tr>
<td>RCI</td>
<td>4.0</td>
<td>0.84</td>
</tr>
<tr>
<td>CCD</td>
<td>6.0</td>
<td>0.84</td>
</tr>
<tr>
<td>ACD</td>
<td>3.0</td>
<td>0.85</td>
</tr>
<tr>
<td>NCD</td>
<td>4.5</td>
<td>0.84</td>
</tr>
<tr>
<td>CDI</td>
<td>6.0</td>
<td>0.84</td>
</tr>
<tr>
<td>IBC</td>
<td>3.0</td>
<td>0.86</td>
</tr>
<tr>
<td>OBC</td>
<td>3.0</td>
<td>0.86</td>
</tr>
<tr>
<td>IC</td>
<td>2.0</td>
<td>0.86</td>
</tr>
<tr>
<td>NBD</td>
<td>3.0</td>
<td>0.84</td>
</tr>
<tr>
<td>CB</td>
<td>3.0</td>
<td>0.86</td>
</tr>
<tr>
<td>MSUI</td>
<td>3.0</td>
<td>0.84</td>
</tr>
<tr>
<td>NC</td>
<td>6.0</td>
<td>0.83</td>
</tr>
</tbody>
</table>

The correlation results

<table>
<thead>
<tr>
<th>Metric</th>
<th>Correlation</th>
<th>P-value (corrected)</th>
</tr>
</thead>
<tbody>
<tr>
<td>RCI</td>
<td>0.16</td>
<td>11.3</td>
</tr>
<tr>
<td>CCD</td>
<td>-0.27</td>
<td>0.13</td>
</tr>
<tr>
<td>ACD</td>
<td>-0.26</td>
<td>0.04</td>
</tr>
<tr>
<td>NCD</td>
<td>-0.19</td>
<td>0.59</td>
</tr>
<tr>
<td>CDI</td>
<td>0.32</td>
<td>11.94</td>
</tr>
<tr>
<td>IBC</td>
<td>-0.30</td>
<td>&lt; 0.01</td>
</tr>
<tr>
<td>OBC</td>
<td>-0.31</td>
<td>&lt; 0.01</td>
</tr>
<tr>
<td>IC</td>
<td>0.47</td>
<td>&lt; 0.01</td>
</tr>
<tr>
<td>NBD</td>
<td>-0.22</td>
<td>0.14</td>
</tr>
<tr>
<td>CB</td>
<td>0.29</td>
<td>0.05</td>
</tr>
<tr>
<td>MSUI</td>
<td>-0.08</td>
<td>2.42</td>
</tr>
<tr>
<td>NC</td>
<td>-0.26</td>
<td>0.27</td>
</tr>
</tbody>
</table>
Dependency profiles

- **Inbound**: provides interface
- **Internal**: no dependencies
- **Outbound**: requires interface
- **Transit**: mixed inbound/outbound code

More internal code is related to more local change

E. Bouwers, et.al. *Dependency profiles for software architecture evaluations.* ICSM 2011
“The percentage of internal code can serve as an indicator for the success of encapsulation of an implemented software architecture.”

We @ International Conference of Software Maintenance and Evolution 2014
The threats

- Local change = encapsulation?
- When is a metric stable?
- Top-level packages = components?
- Is there really no relationship with the other metrics?
What did we do with the results?

In "eating-your-own-dogfood-news", the new component independence metric helped us find a remnant of old design in the that was subsequently refactored, resulting in a +0.1 maintainability and a +0.85 component independence
“Keep implementation details internal”