Towards an Empirical Validation of Aspect-Oriented Coupling Measures

Marc Bartsch  
School of Systems Engineering  
University of Reading  
Reading, RG6 6AY, UK  
m.bartsch@reading.ac.uk

Rachel Harrison  
Stratton Edge Consulting  
Stratton Edge, School Hill  
Cirencester, GL7 2LS, UK  
rachel.harrison@strattonedge.com

Abstract

In this paper we report on ongoing research into the empirical validation of aspect-oriented coupling measures as indicators of maintainability of aspect-oriented software. We investigated the aspects of a medium-scale software package developed in industry and measured the maintenance effort that was spent on this software as the number of changed lines throughout the history of each aspect. This measure of maintenance effort was correlated with size and several of the coupling metrics. Our results seem to suggest that some coupling measures indeed can be considered as possible indicators of maintainability. However, our data also suggests that size is a better indicator of maintenance effort than any other metric we studied.

1 Introduction

In software maintenance, we would like to be able to relate internal software properties to external properties. Object-oriented metrics, for example, have been shown to indicate error-proneness [3] and thus can be considered as tools in the software development and maintenance process that have the potential to reduce development cost. These metrics can help to identify software components that are likely to lead to an increasing amount of cost. To date, there is a lack of similar research for aspect-oriented metrics including aspect-oriented coupling metrics.

Object-oriented coupling metrics may not be fit to be used for aspect-oriented software, since they neglect coupling dimensions which are unknown in object-oriented software. Also, since composition mechanisms are quite different in aspect-oriented software, coupling metrics which have been validated empirically with object-oriented software would at least need to be validated again for aspect-oriented software.

2 Related Work

The concept of coupling for structured design was first introduced by Stevens et al. as “the measure of the strength of association established by a connection from one module to another” [13]. A connection was defined as a reference to some label or address defined elsewhere. In order to avoid "ripple effects", where a change in one component results in errors in another component, a common design aim is to achieve low coupling between components, i.e. a low number of connections [13]. This early concept of coupling was applied to the object-oriented paradigm by Coad and Yourdon to account for inheritance [7].

Research on coupling in aspect-oriented systems has been carried out by Zhao [14]. He defined a coupling framework based on the earlier work by Briand et al. by specifying coupling dependencies between aspects and classes each of which focuses on a certain element of an aspect. Further research into aspect-oriented coupling frameworks has been carried out by Bartsch and Harrison [2] and more recently by Bartolomei et al. [1]. These approaches differ in the choice of framework criteria and level of abstraction.

Aspect-oriented coupling metrics have been defined and applied by Sant'Anna et al. [12], Garcia et al. [9] and Kulesza [10]. In their work, a coupling
metric Coupling Between Components (CBC) has been defined that consists of various coupling connections. Also Ceccato and Tonella suggested a suite of aspect-oriented coupling metrics [5]. In our work we focused on single coupling dimensions as metrics and a combination of all single dimensions in one metric similar to CBC.

Empirical validation of coupling metrics has so far mostly been carried out for object-oriented software. Briand et al., for example, validated object-oriented coupling metrics as indicators of error-proneness for objects [3].

### Metrics

In this study we investigated metrics for maintainability, coupling, size and age of an aspect. All metrics will be presented in this section.

#### 3.1 Maintainability

Maintainability is an external product attribute that we cannot measure directly. Instead, we need to measure direct attributes and hypothesise about their relationship with maintainability. A factor that indicates maintainability is the amount of effort that went into a component to apply changes.

We regard a component to be more maintainable, if few changes are requested for it. Vice versa, we regard a component to be less maintainable if more changes are requested for it during its lifetime. Therefore, in this study we used the number of changed lines of an aspect throughout its version history as a measure of maintainability. Such information can be derived from versioning tools such as CVS. For example, if an aspect consisted of the version history as shown in Figure 1, the maintainability metric would count all lines changes between version 1.1 and 1.2, then between 1.2 and 1.3 and then between all following successive version pairs. The value of this metric is the total sum of all counted line changes.

A change between two versions is defined as the removal, the addition or the modification of a line. This metric makes the underlying assumption that an aspect remains stable within one version, meaning that changes which have been applied to the aspect in one version will not be changed within that same version again.

#### 3.2 Coupling

Our approach to measuring coupling differs from coupling metrics found elsewhere [12, 14] in granularity. Briand et al. remark [4] that in order to combine several coupling dimensions in one single metric, each dimension needs ideally to be weighted, but this is difficult to do accurately. Therefore, we focus on single coupling dimensions as well as on combinations of different coupling dimensions. Bartsch and Harrison identified several different types of coupling in aspect-oriented software [2], such as dynamic crosscutting, static crosscutting, type identifier coupling, method invocation and pattern based coupling. We used a subset of these coupling dimensions in our work so far.

If an aspect is used in a coupling connection, a distinction is made between import and export coupling [4]. For example, with regard to the use of type identifiers, import coupling counts the number of components whose type identifiers are used in a given aspect. Export coupling on the other hand counts the number of components that use the type identifier of the given aspect. In both types of coupling the actual type of coupling connection is the same (use of type identifiers), only the counting rule is different. In this work, we consider import coupling only.

A coupling connection can be described as a relationship between a server component that exports a service and a client component that uses this service [4]. The service can be the type identifier of a class or a method which is invoked by a client. For each coupling dimension we defined two different metrics. The first metric counts a coupling connection between the same client and server component once only and the second metric counts the total number of coupling relationships including duplicate connections. Thus, we were able to compare these slightly different metrics. In total, we defined 8 coupling metrics as shown in Table 1.
<table>
<thead>
<tr>
<th>Metric</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Coupling on Dynamic Crosscutting (CDC)</td>
<td>For each aspect, CDC counts the number of other components which are crosscut dynamically. Duplicate connections are counted once only.</td>
</tr>
<tr>
<td>Coupling on Dynamic Crosscutting Counting Duplicates (CDCCD)</td>
<td>For each aspect, CDC counts the number of other components which are crosscut dynamically. Duplicate connections are counted once only.</td>
</tr>
<tr>
<td>Coupling on Static Crosscutting (CSC)</td>
<td>For each aspect, CSC counts the number of other components which are crosscut statically. Duplicate connections are counted once only.</td>
</tr>
<tr>
<td>Coupling on Static Crosscutting Counting Duplicates (CSCCD)</td>
<td>For each aspect, CSC counts the number of other components which are crosscut statically. Duplicate connections are counted individually.</td>
</tr>
<tr>
<td>Coupling on Type Identifier (CTI)</td>
<td>For each aspect, CTI counts the number of other type identifiers used. Duplicate connections are counted once only.</td>
</tr>
<tr>
<td>Coupling on Type Identifier Counting Duplicates (CTICD)</td>
<td>For each aspect, CTI counts the number of other type identifiers used. Duplicate connections are counted individually.</td>
</tr>
<tr>
<td>ALL</td>
<td>This metrics combines CDC, CSC and TI, i.e. all metrics that count a coupling connection once.</td>
</tr>
<tr>
<td>ALLCD</td>
<td>This metrics combines CDCCD, CSCCD and TICD, i.e. all metrics that count coupling connections individually.</td>
</tr>
</tbody>
</table>

Table 1. Coupling metrics

In Table 1 the two ALL metrics combine all three coupling dimensions in one metric. Again, the difference between the two ALL versions refer to the treatment of duplicate coupling connections. ALL counts connections once while ALLCD counts connections individually. These metrics allow us to find out whether more coupling dimensions may have a stronger correlation with maintenance regardless of the problem of assigning weights to each coupling dimension.

### 3.3 Size and Age

In order to check for size we measured non-commented lines of code of each aspect [8]. In addition to studying the correlation between coupling metrics to maintainability, we will also study the correlation between maintainability and size.

The aspects that we investigated have different initial check-in dates (see section 3) with the effect that some aspects may not be part of the software long enough to undergo maintenance activities. Thus it might be the case that aspects which have been checked in only recently, tend to exhibit fewer changes than aspects that have existed for much longer. In order to check for this we will also look at the correlation between maintainability and the number of days that an aspect resided in the software. More specifically, we determined the latest date that an aspect had checked into the repository as a reference date. Then we calculated the number of days for each aspect from its initial check in date to the reference date. We also correlated this measure of "age" of an aspect with maintenance effort.

### 4 Results

The software under investigation is an open-source troubleshooting agent for Java applications developed in a commercial environment. It is written in AspectJ and contains 385 classes, 64 aspects and has a size of 24,791 NCLOC. The Java classes account for 20,746 NCLOC and the aspects for 4,045 NCLOC. Overall, aspects account for roughly 15% of all components and of the size of the application. The initial check-in date and the age of the aspects varies substantially. Table 2 gives an overview of the distribution of the age of all aspects.

<table>
<thead>
<tr>
<th>Distribution</th>
<th>Age</th>
</tr>
</thead>
<tbody>
<tr>
<td>31%</td>
<td>280 days</td>
</tr>
<tr>
<td>21%</td>
<td>between 200 and 280 days</td>
</tr>
<tr>
<td>9%</td>
<td>between 100 and 200 days</td>
</tr>
<tr>
<td>39%</td>
<td>between 0 and 100 days</td>
</tr>
</tbody>
</table>

Table 2. Distribution of the ages of the aspects

More than 50% of all aspects are older than 200 days, but almost 40% of all aspects are only less than 100 days in the repository, minimizing the opportunity for changes to these aspects.

The study was carried out as follows: first, we obtained the complete CVS repository of the software from the authors. Second, we used TortoiseCVS and WinMerge to go through the entire change history of each aspect file and derived the measure of maintainability. Third, we measured the size of each aspect file in NCLOC, the age of each aspect in days and obtained all coupling measurements. In order to measure coupling we created a plug-in for the Eclipse platform that uses the AjAST infrastructure which is

---

2 WinMerge: http://winmerge.org
part of the AspectJ Development Tools (AJDT)\(^3\). Having obtained all data sets, we derived correlations between maintainability and all other metrics. Table 3 shows all calculated values. Since we did not find a normal distribution for any data set, we use the non-parametric Spearman rank correlation index. We used the SPSS\(^4\) statistical package for all statistical values.

<table>
<thead>
<tr>
<th>Metric</th>
<th>p-value</th>
<th>Spearman rank correlation coefficient</th>
</tr>
</thead>
<tbody>
<tr>
<td>Size</td>
<td>0.000</td>
<td>0.505</td>
</tr>
<tr>
<td>ALL</td>
<td>0.001</td>
<td>0.409</td>
</tr>
<tr>
<td>ALLCD</td>
<td>0.001</td>
<td>0.400</td>
</tr>
<tr>
<td>TI</td>
<td>0.002</td>
<td>0.381</td>
</tr>
<tr>
<td>TICD</td>
<td>0.002</td>
<td>0.377</td>
</tr>
<tr>
<td>CDC</td>
<td>0.002</td>
<td>0.372</td>
</tr>
<tr>
<td>CDCCD</td>
<td>0.004</td>
<td>0.354</td>
</tr>
<tr>
<td>AGE</td>
<td>0.014</td>
<td>0.305</td>
</tr>
<tr>
<td>CSCCD</td>
<td>0.847</td>
<td>0.025</td>
</tr>
<tr>
<td>CSC</td>
<td>0.854</td>
<td>0.024</td>
</tr>
</tbody>
</table>

Table 3. Correlations

Column one of table 3 refers to the name of each metric and the second column lists the p-value. The p-value expresses the probability that two data sets have no linear relationship. Assuming a significance level of 5%, all correlations that exhibit a p-value less than 0.05 are said to correlate significantly at the 5% level, i.e. the probability of detecting such a linear relationship by mistake is 5% or less. The last column shows the correlation coefficient values and expresses the degree and nature of the correlation. The table is sorted in ascending order of the p-values.

The table shows that all correlations are positive and that all metrics but CSCCD and CSC correlate significantly at the 5% level. Size correlates highest with maintenance effort affirming that larger files potentially need more effort to modify. The next highest correlations are ALL and ALLCD i.e. the combination of all coupling dimensions in one metric. Of all coupling metrics that consist of single coupling dimensions, both type identifier metrics correlate equally high with CDC and CDCCD correlating only slightly weaker. A further weaker correlation can be identified for AGE. Almost no correlation exists for both static crosscutting metrics, CSC and CSCCD. It seems that there is no linear relationship at all between static crosscutting and maintenance effort. This result is unexpected and reasons for this need to be determined in a follow-up analysis. Overall, the results show that maintenance effort increases as coupling increases, which implies that coupling should be kept to a minimum for ease of maintenance.

Interestingly, both metrics that combine all three coupling dimension correlate higher than each single coupling metric. Also, there is hardly any difference between the metrics that only differ in respect of their counting rule. It seems that counting a coupling connection once leads in most cases to a higher or equally high correlation compared with the respective metric that count coupling connections individually. Again, this is a result that needs further investigation since it means that such counting the frequency of coupling may not be necessary.

There is a weak correlation between maintenance effort and age which indicates that there is a weak functional relationship between the two. It seems that the age of an aspect is not the main reason to explain the maintenance effort of an aspect. The main result is that even though some coupling metrics could be shown to correlate with maintainability, size is a better indicator of maintenance effort than any other metric that we studied.

5 Threat to Validity

Potential threats to the validity of a study as described in this paper are a major concern. We will discuss threats to internal and external validity. Internal validity describes the extent to which the observed result is caused by the study conditions [6, 11]. In our case we must ask whether any observed effect was caused by the measurement approach only. External validity is the extent to which the results of a study can be applied to and across different persons, settings and times [6, 11].

5.1 Internal Validity

Changeability was measured as the number of line changes throughout the version history of each aspect. This task could not be automated since the decision about whether there is a line change cannot be easily made without an analysis of the actual source code. The problem is that the modification of a single line from one version to the next sometimes is not a modification but the removal and addition of a line and thus counts as two changes instead of one. This decision was made for each single case manually and is of course subjective.

Another issue that influences the outcome of the coupling metrics is the time of measuring. In this study we measured coupling with all latest versions and thus with all aspects available. The change effort requiring a decision should be made on a regular basis, i.e. once a week or bi-weekly. It seems to be more valuable to measure changeability with the version of the aspect that was used in the previous week or two.

\(^3\) AJDT: http://www.eclipse.org/ajdt

\(^4\) SPSS: http://www.spss.com
was determined starting with each initial version to each final version. Thus, the coupling that was measured is the coupling that exists after all maintenance effort has been invested. The coupling at the beginning of the software development would be expected to be quite different.

5.2 External Validity

In this study, we used only a subset of all coupling dimensions as identified in [2]. For example, this study does not include method invocation or coupling due to the use of type patterns in varying degrees. Thus the overall conclusion of this study has to be treated with caution. We need to find how much influence the missing coupling dimensions have on the ALL metrics, i.e., those metrics that combine all single coupling dimensions.

Also, this study only covers a single software package which has been produced by a specific development team. Other teams might have different programming rules or conventions which might lead to different result.

6 Conclusions and Future Work

In this paper we reported on ongoing research into the empirical validation of aspect-oriented coupling metrics. We investigated a medium-size software package written in industry in AspectJ and studied the correlation between maintenance effort and several metrics among which were 8 coupling metrics. The data shows that although weak correlations can be found for 6 coupling metrics, size is a better indicator of maintenance effort than any other metric we studied. Two coupling metrics that involve static crosscutting showed virtually no correlation at all. These results need further investigation, particularly the question of why static crosscutting failed to be correlated with effort. The best candidate for an indicator of change effort apart from size was the combination of all coupling dimensions. More research should be carried out to determine whether a certain combination of coupling metrics might yield better results. Such an investigation will be future work as well as the addition of further coupling dimensions which have been neglected so far. One reason for this neglect is the lack of tool support to easily determine coupling relationships in AspectJ.

The lack of tool support is also a problem for future investigation. In particular the problem of determining maintenance effort from a version history in an automated manner needs to be solved to scale these kinds of investigations to larger software projects. Certainly, more industrial data needs to be investigated before more definitive conclusions can be drawn about the suitability of aspect-oriented coupling metrics as indicators of maintenance effort.

Acknowledgements

We would like to sincerely thank the authors of the software package investigated in this study for providing us with the entire CVS repository and their support. We would also like to thank Matt Chapman for his support on the Eclipse AJDT mailing list.

References


