Chapter - 5

Cognitive Weighted Coupling on Advice Execution
CHAPTER - 5

COGNITIVE WEIGHTED COUPLING ON ADVICE EXECUTION

5.1 Introduction

Software engineering is the study and application of engineering to the design, development, and maintenance of software. Software metric is a measure of some property of a piece of software or its specifications. Metrics attempt to measure a particular aspect of a software system. There are several approaches to estimate the complexity of software, but none of them has been accepted as a true measure of the complexity of a class.

In computing, aspect-oriented programming is a programming paradigm that aims to increase modularity (a grouping of related code) by allowing the separation of cross-cutting concerns. AOP forms a basis for aspect-oriented software development. Out of the available AOP languages, AspectJ is the most popular and widely used in research areas. AspectJ is a simple general purpose extension to Java that provides, through the definition of new constructors, and supporting for modular implementation of crosscutting concerns. AspectJ has been successfully used to cleanly modularize implementations of crosscutting concerns such as synchronization, consistency checking, protocol management and others.

The aspect is the modular unit of crosscutting implementation. Each aspect encapsulates functionality that crosscuts other classes in an AspectJ program. A central concept in the composition of an aspect with other classes is called a joinpoint, which is a well-defined point in the execution of a program, such as a call to a method, access to an attribute, an object initialization, an exception handler etc.
Joinpoint in AspectJ reduces complexity. Joinpoint is desired by the developers due to several positive quality attributes like reliability, maintainability and portability are associated with it. Therefore, there is a constant search for better, more accurate and easy to compute joinpoint complexity metric in the software industry. So, there is a need for cognitive weighted Coupling on Advice Execution (CAE) for the Aspect level measurement. Hence the primary goal of the study is to define a Cognitive Weighted Coupling on Advice Execution (CWCAE) metric to measure the Complexity of various types of Joinpoints.

5.2 Motivation

The researchers have proposed several metrics for AOP systems. One of the metrics proposed by Ceccato et al. [Cec, 04] and Kotrappa Sirbi et al. [Kot, 10] is CAE. Coupling on Advice Execution (CAE) is some aspects containing advice possibly triggered by the execution of operations in a given module. However, such kind of coupling is not available in Object Oriented (OO) systems.

Bartsch and Harrison [Bar, 06] revealed the joinpoints which cause advice to be executed. AspectJ supports more types of joinpoints that can also cause the execution of advice, such as object initialization joinpoints, exception handler joinpoints, call joinpoints and advice execution joinpoints. A valid measure of coupling on advice execution needs to count all of these joinpoint coupling mechanisms.

The Weighted Joinpoints (WJP) metric was proposed by Parthipan, Senthil Velan, Chitra Babu [Par, 14] in which the WJP is calculated by taking the sum of cognitive weights of types of joinpoints. The cognitive weight assigned to the identified designators is based on its cognitive complexity. The drawback of the WJP
metric is that the metric did not prove their metric according to the statistical approach. No theoretical validation was done by the prominent properties.

5.3 Formulation of the CWCAE Metric

5.3.1 Existing Metric

Coupling on Advice Execution (CAE) is some aspects containing advices possibly triggered by the execution of operations in a given module. If the behavior of an operation can be altered by aspect advice, due to a pointcut intercepting it, there is an (implicit) dependence of the operation from the advice. Thus, the given module is coupled with the aspect containing the advice, and a change of the latter might impact the former. However, such kind of coupling is not found in Object Oriented (OO) systems.

5.3.2 Proposed Metric

The work formally defines the proposed metric called Cognitive Weighted CAE (CWCAE). The mathematical equation of the metric is based on the existing metric called Coupling on Advice Execution suggested by Ceccato et al. [Cec, 04] and Kotrappa Sirbi et al. [Kot, 10]. Joinpoints are one of the significant factors which will affect the complexity of the aspect, and it is clear that the use of different data type of joinpoints increases the complexity of the programs. There is no specific measure exists to calculate the complexity due to joinpoint. Hence, a new metric CWCAE has been proposed for AOP system with joinpoint. AspectJ supports more types of joinpoints that can also cause the execution of advice, such as object initialization joinpoints, exception handler joinpoints, call joinpoints and advice
execution joinpoints. The proposed metric called Cognitive Weighted CAE (CWCAE) considers the cognitive complexity of the different types of joinpoints. The overall assessment framework is given in Figure 5.1.

\[
\text{Comp}_{\text{CAE}}(j) = \sum_{j \in J} \text{comp}(j)
\]

5.4 Experimental Design

In this section, an experiment is conducted to assign cognitive weight to the various types of joinpoint. A comprehensive test has been conducted for a group of students to find out the time taken to understand the complexity of aspect-oriented program concerning different types of joinpoint. The group consisting of 60 students were selected who had sufficient exposure in analysing the aspect-oriented programs, as they had undergone courses in AspectJ language.

The time taken by students to comprehend the programs was recorded after the completion of each program. The time taken for comprehension of all these programs was noted and the mean time was calculated. Five different programs were
administered in each case, and entirely 30 different timings were recorded. Average
time was calculated for each program from the individual time taken by students
which are shown in Table 5.1.

Table 5.1 Categorized mean comprehension time

<table>
<thead>
<tr>
<th>Programs</th>
<th>Average Comprehension Time (In Minutes)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Method Call (MC) / Constructor Call (CC)</td>
</tr>
<tr>
<td>P1</td>
<td>15</td>
</tr>
<tr>
<td>P2</td>
<td>14</td>
</tr>
<tr>
<td>P3</td>
<td>14</td>
</tr>
<tr>
<td>P4</td>
<td>12</td>
</tr>
<tr>
<td>P5</td>
<td>13</td>
</tr>
<tr>
<td>Mean value</td>
<td>13.6</td>
</tr>
</tbody>
</table>

The cognitive weights for different types of joinpoint are calibrated in this
section. To find the cognitive weight factor for each of the joinpoint, cognitive-based
psychological experiments are conducted for five different groups of students to find
out the time and effort taken to understand the complexity of different types of
joinpoint. The students are given sufficient exposure to AspectJ programming and
especially in understanding various degrees of joinpoint within an aspect. Around 60
students, who have scored 65% and above marks in Semester examination, are
selected for the comprehension test. They are supplied 30 different programs P1 to P5
for each type of joinpoint. The time taken by each student to understand the program
and time taken is recorded automatically by the computer after the completion of each program. This process is repeated for each group of students.

These program comprehension tests are registered to be accurate, and the comprehension timings are registered automatically by the computer in minutes. The average time taken to comprehend each program from P1 to P5 by each group is calculated, to get 30 different CMTs. These values are tabulated in Table 5.1. These programs are based on Aspect-Oriented Programming. The mean time is also calculated for each category of the programs.

5.5 Cognitive Weighted Coupling on Advice Execution

The proposed metric called Cognitive Weighted Coupling on Advice Execution (CWCAE), which considers the cognitive complexity of the different types of joinpoints such as object initialization joinpoint, exception handler joinpoint, call joinpoint and advice execution joinpoint. The existing CAE metric proposed by Ceccato et al. [Cec, 04] and Kotrappa Sirbi et al. [Kot, 10] counts the number of aspects containing advice possibly triggered by the execution of methods or advice. This metric did not consider the various types of joinpoint. CWCAE can be calculated by using the Equation as follows,

\[
    CWCAE = ((MC*CW_{MC}) + (ME*CW_{ME}) + (CC*CW_{CC}) + (CE*CW_{CE}) + (CI*CW_{CI}) + (FR*CW_{FR}) + (FW*CW_{FW}) + (EH*CW_{EH})) \quad \ldots \quad (5.1)
\]

where,

MC  Method Call
ME  Method Execution
CC  Constructor Call
CE  Constructor Execution
CI  Class Initialization
FR  Field Read Access
FW  Field Write Access
EH  Exception Handler execution

$C_{CC}$  Cognitive Weight of Constructor Call
$C_{CE}$  Cognitive Weight of Constructor Execution
$C_{CI}$  Cognitive Weight of Class Initialization
$C_{FR}$  Cognitive Weight of Field Read Access
$C_{FW}$  Cognitive Weight of Field Write Access
$C_{EH}$  Cognitive Weight of Exception Handler execution

5.6 Statistical Analysis

The CAE metric is selected for AO software for conducting statistical analysis. This metric is used to find the complexity of various types of advice using Cognitive Approach. The relationship among the joinpoints is evaluated and analyzed statistically. For each joinpoint, mean was selected as a measure of correlation between other joinpoints. Table 5.2 illustrates the Pearson correlation between comprehension time of different joinpoints and the corresponding graph is shown in Figure 5.2.
Table 5.2 Correlation between Comprehension time of different joinpoints

<table>
<thead>
<tr>
<th>Joinpoints</th>
<th>MC/CC</th>
<th>ME/CE</th>
<th>FR</th>
<th>FW</th>
<th>CI</th>
<th>EH</th>
</tr>
</thead>
<tbody>
<tr>
<td>MC/CC</td>
<td>1</td>
<td>0.8077</td>
<td>0.9231</td>
<td>0.6940</td>
<td>0.5718</td>
<td>0.3468</td>
</tr>
<tr>
<td>ME/CE</td>
<td>0.8077</td>
<td>1</td>
<td>0.5385</td>
<td>0.6940</td>
<td>0.0673</td>
<td>0.7473</td>
</tr>
<tr>
<td>FR</td>
<td>0.9231</td>
<td>0.5385</td>
<td>1</td>
<td>0.4627</td>
<td>0.7736</td>
<td>0.0534</td>
</tr>
<tr>
<td>FW</td>
<td>0.6940</td>
<td>0.6940</td>
<td>0.4627</td>
<td>1</td>
<td>0.0759</td>
<td>0.2408</td>
</tr>
<tr>
<td>CI</td>
<td>0.5718</td>
<td>0.0673</td>
<td>0.7736</td>
<td>0.0759</td>
<td>1</td>
<td>-0.0934</td>
</tr>
<tr>
<td>EH</td>
<td>0.3468</td>
<td>0.7473</td>
<td>0.0534</td>
<td>0.2408</td>
<td>-0.0934</td>
<td>1</td>
</tr>
</tbody>
</table>

| Mean Correlation Value | 0.7239 | 0.6425 | 0.6252 | 0.5279 | 0.3992 | 0.3825 |

Figure 5.2 Mean Correlation Time of different types of joinpoint

The Cognitive Weight of each type of joinpoint is calibrated in Table 5.3. The weight value is calculated based on the mean time and mean correlation time, to get
appropriate weight value. Average mean value of each type of joinpoint is divided by the corresponding mean correlation time.

By normalizing these values, it is found \( CW_{MC} = 1 \), \( CW_{ME} = 1 \), \( CW_{CC} = 1.4 \), \( CW_{CE} = 1.4 \), \( CW_{CF} = 3 \), \( CW_{FR} = 3.7 \), \( CW_{FW} = 2.3 \), \( CW_{EH} = 3.7 \). These values represent the difficulty of understanding various joinpoint types. The finalize weight values are given in Table 5.3.

**Table 5.3 Cognitive Weight Value of each type of Joinpoint**

<table>
<thead>
<tr>
<th>Joinpoint</th>
<th>Cognitive Weight Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>( CW_{MC} )</td>
<td>1</td>
</tr>
<tr>
<td>( CW_{CC} )</td>
<td>1</td>
</tr>
<tr>
<td>( CW_{ME} )</td>
<td>1.4</td>
</tr>
<tr>
<td>( CW_{CE} )</td>
<td>1.4</td>
</tr>
<tr>
<td>( CW_{FR} )</td>
<td>1.9</td>
</tr>
<tr>
<td>( CW_{FW} )</td>
<td>2.3</td>
</tr>
<tr>
<td>( CW_{CI} )</td>
<td>3</td>
</tr>
<tr>
<td>( CW_{EH} )</td>
<td>3.7</td>
</tr>
</tbody>
</table>

where,

\[
CW_{MC} = \text{Cognitive Weight of Method Call} \\
CW_{ME} = \text{Cognitive Weight of Method Execution} \\
CW_{CC} = \text{Cognitive Weight of Constructor Call} \\
CW_{CE} = \text{Cognitive Weight of Method Execution} \\
CW_{CI} = \text{Cognitive Weight of Class Initialization} \\
CW_{FR} = \text{Cognitive Weight of Field Read Access} \\
CW_{FW} = \text{Cognitive Weight of Field Write Access} \\
CW_{EH} = \text{Cognitive Weight of Exception Handler execution}
\]
5.7 Theoretical Validation

Fenton’s properties are used for theoretical validation of CWCAE, as it has been proposed in the evaluation framework [Fen, 97]. Those properties were used for the data collection process and are described as follows:

- **Accuracy**: The higher the difference between the actual data and measured data and the lower is the accuracy and vice-versa. It implies that the metric value for CAE is lower than the proposed CWCAE value.
- **Replicability**: Means that the analysis can be done at different times by different people using the same setting. Data are taken from rural and urban PG students at a different time.
- **Correctness**: According to the metrics definition data was collected.
- **Precision**: Data is expressed by some decimal places. Less number of decimal places shows a lower accuracy. If the decimal place of the data is high, it shows a higher accuracy.
- **Consistency**: It counts the differences in the metric values when collected using different tools by different people.

The following section explains how CWCAE is calculated using a case study.

5.8 Illustration

The proposed CWCAE metric given by Eq 5.1 is evaluated with both Java and AspectJ programs given in Appendix B. From the program, the existing and proposed metric value is calculated as.

\[
\text{CAE} = \sum_{X=0}^{8} \text{cmpx(ad)}
\]
x=6 so,
CMPX(CAE) = 6

CWCAE

CWCAE= ((MC*CW_{MC}) + (CE*CW_{CE}) + (FR*CW_{FR}) + (FW*CW_{FW}) + (CI*CW_{CI}) + 

(EH*CW_{EH}))

CWCAE= ((1*1) + (1*1.4) + (1*1.9) + (1*2.3) + (1*3) + (1*3.7))

CWCAE= 1 + 1.4 + 1.9 +2. 3 +3 +3.7 = 13.5

The joinpoint complexity metric value for the example program are shown in the Table 5.4.

<table>
<thead>
<tr>
<th>Program#</th>
<th>CAE</th>
<th>CWCAE</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>6</td>
<td>13.5</td>
</tr>
</tbody>
</table>

5.9 Comparative Study

Ceccato et al. [Cec, 04] and Kotrappa Sirbi et al. [Kot, 10] proposed a metric called CAE and made a comparative study. CAE defines a total number of the aspects containing advice possibly triggered by the execution of methods or advice. The current CWCAE metric is one step ahead of existing CAE metric because it includes the complexity that arises due to the various types of Joinpoints. Another advantage of CWCAE metric is that it takes cognitive weights into consideration and data collection satisfies the Fenton [Fen, 97] properties. To compare the proposed metric, a comprehension test was conducted for graduate students. Sixty students participated in the test; the students were given five different programs in AspectJ for the
comprehension test. The test was to find out the output of the given programs. The time taken to complete the test in minutes is recorded. The average time taken by all the students is calculated. In the following Table 5.5, a comparison has been proved with CAE, CWCAE of the comprehension test result. The corresponding graph is shown in Figure 5.3.

Table 5.5 Complexity metric values and mean comprehension time

<table>
<thead>
<tr>
<th>Program#</th>
<th>Existing Metric Value (CAE)</th>
<th>Proposed Metric Value (CWCAE)</th>
<th>Mean Comprehension Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>7</td>
<td>14</td>
<td>16</td>
</tr>
<tr>
<td>2</td>
<td>6</td>
<td>17</td>
<td>18</td>
</tr>
<tr>
<td>3</td>
<td>11</td>
<td>19.5</td>
<td>21</td>
</tr>
<tr>
<td>4</td>
<td>6</td>
<td>20.5</td>
<td>22.8</td>
</tr>
<tr>
<td>5</td>
<td>8</td>
<td>23.5</td>
<td>24</td>
</tr>
</tbody>
</table>

![Figure 5.3 Complexity metric values Vs. mean comprehension time](image-url)
The CAE complexity of the class is calculated by computing Method Call (MC), Method Execution (ME), Constructor Call (CC), Constructor Execution (CE), Class Initialization (CI), Field Read Access (FR), Field Write Access (FW) and Exception Handler Execution (EH). The weight of each type of joinpoint is calculated by using cognitive weights and weighing factor of the type of the joinpoint similar to that suggested by Wang et al. [Wan, 03]. It is found that the resulting value of CWCAE is larger than the CAE. This is because, in CAE, the weight of each advice is assumed to be one. However, including cognitive weights for calculation of the CWCAE is more realistic because it considers different types of Joinpoint. The results are shown in Table 5.5. A correlation analysis was performed between CAE Vs. Comprehension Time with $r = 0.221981$ and CWCAE Vs. Comprehension time with $r = 0.980778$. CWCAE is more positively correlated than CAE. It is observed that CWCAE is a better indicator of the complexity of the classes with various types of joinpoint in Advice Execution.

5.10 Conclusion

A CWCAE metric for measuring the class level complexity has been formulated. The complexity of the class includes the Advice Execution complexity of the class. CWCAE includes the cognitive complexity due to different types of Joinpoint. CWCAE has proved that the complexity of the class getting affected, is based on the cognitive weights of the various types of Joinpoint. The assigned cognitive weight of the various types of Joinpoint is validated using the comprehension test and found that the cognitive load to understand the EH > CI > FW > FW > FR > CE, ME > CC, MC. The metric is evaluated through statistical analysis, case study and a comparative.
study, and proved to be a better indicator of the class level complexity. Newer metrics may also be proposed and validated for assessing the cognitive complexity of another type of joinpoint. Having discussed the proposed CWCAE metric that is used to measure the complexity of AO software concerning the cognitive complexity, the following chapter attempts to define and validate the Cognitive Weighted Pointcut per Aspect (CWPA) metric.