Measuring the Coupling of Imperative Computer Programs

by

Sadiq Hassan Al-Nasser

A Thesis Presented to the

FACULTY OF THE COLLEGE OF GRADUATE STUDIES
KING FAHD UNIVERSITY OF PETROLEUM & MINERALS
DHAHRAN, SAUDI ARABIA

In Partial Fulfillment of the Requirements for the Degree of

MASTER OF SCIENCE

In

COMPUTER SCIENCE

June, 1997
INFORMATION TO USERS

This manuscript has been reproduced from the microfilm master. UMI films the text directly from the original or copy submitted. Thus, some thesis and dissertation copies are in typewriter face, while others may be from any type of computer printer.

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleedthrough, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send UMI a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

Oversize materials (e.g., maps, drawings, charts) are reproduced by sectioning the original, beginning at the upper left-hand corner and continuing from left to right in equal sections with small overlaps. Each original is also photographed in one exposure and is included in reduced form at the back of the book.

Photographs included in the original manuscript have been reproduced xerographically in this copy. Higher quality 6" x 9" black and white photographic prints are available for any photographs or illustrations appearing in this copy for an additional charge. Contact UMI directly to order.

UMI
A Bell & Howell Information Company
300 North Zeeb Road, Ann Arbor MI 48106-1346 USA
313/761-4700 800/521-0600
MEASURING THE COUPLING OF IMPERATIVE
COMPUTER PROGRAMS

BY

SADIQ HASSAN AL-NASSER
A Thesis Presented to the
FACULTY OF THE COLLEGE OF GRADUATE STUDIES
KING FAHD UNIVERSITY OF PETROLEUM & MINERALS
DHAHRAHN, SAUDI ARABIA

In Partial Fulfillment of the
Requirements for the Degree of

MASTER OF SCIENCE
In

COMPUTER SCIENCE

JUNE, 1997
KING FAHD UNIVERSITY OF PETROLEUM AND MINERALS  
DHAHRAN, SAUDI ARABIA  
COLLEGE OF GRADUATE STUDIES

This thesis, written by

SADEG HASSAN AHMED AL-NASSER

under the direction of his Thesis Advisor and approved by his Thesis Committee, has been 
presented to and accepted by the Dean of the College of Graduate Studies, in partial 
fulfillment of the requirements for the degree of

MASTERS OF SCIENCE IN COMPUTER SCIENCE.

Thesis Committee

Dr. Jarallah S. AlGhamdi (Chairman)

Dr. Muhammed S. Al-Mulhem (Member)

Muhammed Shafique (Member)

Department Chairman

Dean, College of Graduate Studies

6-8-97

Date
Dedicated to

my

parents,

wife,

brothers,

and

sisters

whose patience and prayer

let to this accomplishment.
Acknowledgment

I would like to acknowledge King Fahd University of Petroleum and Minerals for all the support extended during this research.

I would like to extend my appreciation to my thesis committee chairman, Dr. Jarallah S. AlGhamdi for his continuous advice, guidance and cooperation. I feel grateful to my thesis committee members, Dr. Muhammed S. Al-Mulhem and Dr. Muhammed Shafique for their useful suggestions and cooperation.

Thanks for all my colleges for their suggestions during my thesis preparation and for all who encourages me.

A good appreciation to my dead mother who lives in my heart, father, brothers and sisters for their encouragement and support, especially my brothers Redha for reviewing the thesis, Abdul-Ghany for his cooperation, Ahmed for reviewing the Arabic abstract and my brother in law Yousef for writing it.

Finally, I would like to express my sincere thanks and gratitude to my wife for her continuous assistance, cooperation, patience, motivation and sacrifices.
Contents

Acknowledgment .................................. 1
List of Tables ................................ VI
List of Figures .................................. xii
Abstract (English) ................................ xiv
Abstract (Arabic) ................................ xv

1 Introduction .................................. 1
  1.1 Overview of Module Coupling and Cohesion .. 1
    1.1.1 Coupling Classifications .............. 2
      1.1.1.1 Content Coupling .................. 2
      1.1.1.2 Common Coupling ................. 3
      1.1.1.3 Control Coupling .................. 3
      1.1.1.4 Stamp Coupling .................... 4
      1.1.1.5 Data Coupling ..................... 4
    1.1.2 Cohesion Classifications ............ 5
      1.1.2.1 Coincidental Cohesion .......... 5
      1.1.2.2 Logical Cohesion ................. 5
      1.1.2.3 Temporal Cohesion ............... 6
      1.1.2.4 Procedural Cohesion ............. 6
      1.1.2.5 Communication Cohesion .......... 7
      1.1.2.6 Information Cohesion .......... 7
      1.1.2.7 Functional Cohesion ............ 8
  1.2 Software Complexity Measurements ....... 8
    1.2.1 Software Complexity ................ 9
      1.2.2 Measurement ........................ 10
      1.2.3 Measurement Scales ............... 11
        1.2.3.1 Nominal Scale ................. 12
        1.2.3.2 Ordinal Scale ................. 12
        1.2.3.3 Interval Scale ............... 12
        1.2.3.4 Ratio Scale .................. 13
        1.2.3.5 Absolute Scale ............... 13
  1.3 Software Metric Classifications ......... 14
  1.4 Problem Definition ...................... 15
  1.5 Overview of the Solution ............... 16
  1.6 Thesis Organization .................... 18
5.2.3.3 Fenton and Melton Method ........ 98
5.2.3.4 Offut, Harrold and Kotle Method . 98
5.2.3.5 Chen and Lu Method ............... 98
5.2.3.6 Analysis .................................. 99

5.2.4 Module IV .................................. 100
5.2.4.1 New Method ............................... 100
5.2.4.2 Dhama Method ............................ 101
5.2.4.3 Fenton and Melton Method ............ 102
5.2.4.4 Offut, Harrold and Kotle Method . 102
5.2.4.5 Chen and Lu Method .................... 102
5.2.4.6 Analysis .................................. 103

5.2.5 Module V .................................. 104
5.2.5.1 New Method ............................... 105
5.2.5.2 Dhama Method ............................ 106
5.2.5.3 Fenton and Melton Method ............ 107
5.2.5.4 Offut, Harrold and Kotle Method . 107
5.2.5.5 Chen and Lu Method .................... 108
5.2.5.6 Analysis .................................. 108

5.2.6 Module VI .................................. 109
5.2.6.1 New Method ............................... 110
5.2.6.2 Dhama Method ............................ 112
5.2.6.3 Fenton and Melton Method ............ 112
5.2.6.4 Offut, Harrold and Kotle Method . 113
5.2.6.5 Chen and Lu Method .................... 113
5.2.6.6 Analysis .................................. 114

5.2.7 Analysis Summary ......................... 115

5.3 Effect of Change in Coupling Values .......... 116
5.3.1 Remove a Procedure from Module II .......... 118
5.3.1.1 New Method ............................... 118
5.3.1.2 Dhama Method ............................ 119
5.3.1.3 Fenton and Melton Method ............ 120
5.3.1.4 Offut, Harrold and Kotle Method . 120
5.3.1.5 Chen and Lu Method .................... 120
5.3.1.6 Analysis .................................. 121

5.3.2 Combine Two Procedures in Module III ........ 121
5.3.2.1 New Method ............................... 122
5.3.2.2 Dhama Method ............................ 123
5.3.2.3 Fenton and Melton Method ............ 124
5.3.2.4 Offut, Harrold and Kotle Method . 124
5.3.2.5 Chen and Lu Method .................... 124
5.3.2.6 Analysis .................................. 125

5.3.3 Replace Leap in Module IV ................. 126
5.3.3.1 New Method ............................... 126
5.3.3.2 Dhama Method ............................ 127
5.3.3.3 Fenton and Melton Method ............ 128
5.3.3.4 Offut, Harrold and Kotle Method . 128
5.3.3.5 Chen and Lu Method .................... 128
5.3.3.6 Analysis .................................. 129

5.3.4 Increase Element Weight in Module V .......... 130
5.3.4.1 New Method ............................... 130
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.3.4.2</td>
<td>Dhama Method</td>
<td>132</td>
</tr>
<tr>
<td>5.3.4.3</td>
<td>Fenton and Melton Method</td>
<td>132</td>
</tr>
<tr>
<td>5.3.4.4</td>
<td>Offut, Harrold and Kotle Method</td>
<td>133</td>
</tr>
<tr>
<td>5.3.4.5</td>
<td>Chen and Lu Method</td>
<td>133</td>
</tr>
<tr>
<td>5.3.4.6</td>
<td>Analysis</td>
<td>134</td>
</tr>
<tr>
<td>5.3.5</td>
<td>Reduce Element Weight in Module VI</td>
<td>135</td>
</tr>
<tr>
<td>5.3.5.1</td>
<td>New Method</td>
<td>136</td>
</tr>
<tr>
<td>5.3.5.2</td>
<td>Dhama Method</td>
<td>138</td>
</tr>
<tr>
<td>5.3.5.3</td>
<td>Fenton and Melton Method</td>
<td>138</td>
</tr>
<tr>
<td>5.3.5.4</td>
<td>Offut, Harrold and Kotle Method</td>
<td>139</td>
</tr>
<tr>
<td>5.3.5.5</td>
<td>Chen and Lu Method</td>
<td>139</td>
</tr>
<tr>
<td>5.3.5.6</td>
<td>Analysis</td>
<td>140</td>
</tr>
<tr>
<td>5.3.6</td>
<td>Use Parameters in Module VI</td>
<td>140</td>
</tr>
<tr>
<td>5.3.6.1</td>
<td>New Method</td>
<td>141</td>
</tr>
<tr>
<td>5.3.6.2</td>
<td>Dhama Method</td>
<td>143</td>
</tr>
<tr>
<td>5.3.6.3</td>
<td>Fenton and Melton Method</td>
<td>143</td>
</tr>
<tr>
<td>5.3.6.4</td>
<td>Offut, Harrold and Kotle Method</td>
<td>144</td>
</tr>
<tr>
<td>5.3.6.5</td>
<td>Chen and Lu Method</td>
<td>144</td>
</tr>
<tr>
<td>5.3.6.6</td>
<td>Analysis</td>
<td>145</td>
</tr>
<tr>
<td>5.3.7</td>
<td>Analysis Summary</td>
<td>145</td>
</tr>
<tr>
<td>6</td>
<td>Conclusion</td>
<td>148</td>
</tr>
<tr>
<td>6.1</td>
<td>Accomplishments</td>
<td>148</td>
</tr>
<tr>
<td>6.2</td>
<td>Contributions</td>
<td>150</td>
</tr>
<tr>
<td>6.3</td>
<td>Future Works</td>
<td>150</td>
</tr>
<tr>
<td>Appendix A</td>
<td></td>
<td>153</td>
</tr>
<tr>
<td>References</td>
<td></td>
<td>177</td>
</tr>
</tbody>
</table>
List of Tables

1 Fenton and Melton Modified Definitions for Myers Coupling Levels .......................... 32
2 Offut, Harrold and Kotele Modified Definitions for Myers Coupling Levels .................. 36

3 Comparison between the Four Recent Coupling Measures among the Eight Identified Coupling Factors 57
4 Element Type Weight Comparison between Three Methods and the Proposed Method ............. 60
5 The Combination of Myers Three Coupling Factors and their Corresponding Proposed Weights .... 61
6 Element Scope Weight Comparisons between Dhama and the Proposed Method .................. 62
7 Element Access Purpose Weight Comparisons between Dhama and the Proposed Method .......... 62
8 Element Type Weight Comparisons between Dhama and the Proposed Method .................. 62

9 The Description Matrix with Weights Flags Assigned from Source Code with Non-relevance Variables Marked with 'x' .......................................................... 77
10 The Description Matrix with Coupling Weights Assigned Separated by Commas ................. 78
11 The Final Generated Description Matrix ................................................................. 78
12 Dhama Method Weights Calculation Summary ......................................................... 80
13 Procedure by Procedure Coupling Values Using Dhama Method .................................. 80
14 Fenton and Melton Modified Definitions for Myers Coupling Levels .......................... 81
15 Procedure Pair Coupling Weights Using Fenton and Melton Method ............................. 82
16 Offut, Harrold and Kotele Modified Definitions for Myers Coupling Levels .................. 83
17 Procedure Pair Coupling Weights Using Offut, Harrold and Kotele Method ..................... 85
18 Procedure Coupling Weights Using Chen and Lu Method ........................................ 86
19 Procedure by Procedure Coupling Values Using Chen and Lu Method .......................... 87
20 Module I Definition Matrix ....................................................................................... 88
80. Comparisons of New and Recent Measures on Capturing Relation on Modules. 115
81. Module II Definition Matrix After Removal of Main Procedure 118
82. Module II Procedure to Procedure Coupling Matrix after Removal of Main Procedure. 118
83. Module II Procedure to Procedure Normalized Coupling Matrix After Removal of Main Procedure. 119
84. Module II Procedures Overall Coupling After Removal of Main Procedure 119
85. Module II Procedures Coupling Weights Using Dhama Method 119
86. Module II Procedure to Procedure Coupling Weight Using Dhama Method After Removal of Main Procedure 119
87. Module II Procedure to Procedure Coupling Weight Using Fenton and Melton Method After Removal of Main Procedure 120
88. Module II Procedure to Procedure Coupling Weight Using Offut, Harrold and Kotle Method 120
89. Module II Procedures Coupling Weight Using Chen and Lu Method After Removal of Main Procedure 120
90. Module II Procedure to Procedure Coupling Weight Using Chen and Lu Method After Removal of Main Procedure 120
91. Module III Definition Matrix After Combining AVCALC and SUMCAL into new AVCALC Procedure 122
92. Module III Procedure to Procedure Coupling Matrix After Combining AVCALC and SUMCAL into new AVCALC Procedure 122
93. Module III Procedure to Procedure Normalized Coupling Matrix After Combining AVCALC and SUMCAL into new AVCALC Procedure 123
94. Module III Procedures Overall Coupling After Combining AVCALC and SUMCAL into new AVCALC Procedure 123
95. Module III Procedures Coupling Weights Using Dhama Method After Combining AVCALC and SUMCAL into new AVCALC Procedure 123
96. Module III Procedure to Procedure Coupling Weights Using Dhama Method After Combining AVCALC and SUMCAL into new AVCALC Procedure 123
97. Module III Procedure to Procedure Coupling Weights Using Fenton and Melton Method After Combining AVCALC and SUMCAL into new AVCALC Procedure 124
98. Module III Procedure to Procedure Coupling Weights Using Offut, Harrold and Kotle Method After Combining AVCALC and SUMCAL into new AVCALC Procedure 124
99. Module III Procedures Coupling Weights Using Chen and Lu Method After Combining AVCALC and SUMCAL into new AVCALC Procedure 124
100 Module III Procedure to Procedure Coupling Weight
Using Chen and Lu Method After Combining AVCALC
and SUMCAL into new AVCALC Procedure .......... 125
101 Module IV Definition Matrix After Removing One Leap
Procedure and Let VALDAY Procedure Use the Other
LEAP Procedure ..................................... 126
102 Module IV Procedure to Procedure Coupling Matrix
After Removing One Leap Procedure and Let VALDAY
Procedure Use the Other LEAP Procedure .......... 126
103 Module IV Procedure to Procedure Normalized Coupling
Matrix After Removing One Leap Procedure and Let
VALDAY Procedure Use the Other LEAP Procedure ... 127
104 Module IV Procedures Overall Coupling After Removing
One Leap Procedure and Let VALDAY Procedure Use
the Other LEAP Procedure .......................... 127
105 Module IV Procedures Coupling Weights Using Dhama
Method After Removing One Leap Procedure and Let
VALDAY Procedure Use the Other LEAP Procedure ... 127
106 Module IV Procedure to Procedure Coupling Weights
Using Dhama Method After Removing One Leap
Procedure and Let VALDAY Procedure Use the
Other LEAP procedure .............................. 127
107 Module IV Procedure to Procedure Coupling Weights
Using Fenton and Melton Method After Removing One
Leap Procedure and Let VALDAY Procedure Use the
Other LEAP Procedure .............................. 128
108 Module IV Procedure to Procedure Coupling Weights
Using Offut, Harrold and Kotle Method After
Removing One Leap Procedure and Let VALDAY
Procedure Use the Other LEAP Procedure .......... 128
109 Module IV Procedures Coupling Weights Using Chen
and Lu Method After Removing One Leap Procedure
and Let VALDAY Procedure Use the Other LEAP
Procedure ............................................. 128
110 Module IV Procedure to Procedure Coupling Weight
Using Chen and Lu Method After Removing One
Leap Procedure and Let VALDAY Procedure Use the
Other LEAP Procedure .............................. 129
111 Module V Description Matrix After Increasing the
weight of WON in Main Procedure From 3 to 6 ...... 130
112 Module V Procedure to Procedure Coupling Matrix
After Increasing the weight of WON in Main
Procedure From 3 to 6 .............................. 131
113 Module V Procedure to Procedure Normalized
Coupling Matrix After Increasing the weight of
WON in Main Procedure From 3 to 6 .............. 131
114 Module V Procedures Overall Coupling After
Increasing the weight of WON in Main procedure
From 3 to 6 ........................................ 131
115 Module V Procedures Coupling Weights Using Dhama Method After Increasing the weight of WON in Main Procedure From 3 to 6 .......... 132
116 Module V Procedure to Procedure Coupling Weights Using Dhama Method After Increasing the weight of WON in Main Procedure From 3 to 6 .......... 132
117 Module V Procedure to Procedure Coupling Weights Using Fenton and Melton Method After Increasing the weight of WON in Main Procedure From 3 to 6 .. 132
118 Module V Procedure to Procedure Coupling Weights Using Offut, Harrold and Kotle Method After Increasing the weight of WON in Main Procedure From 3 to 6 ............... 133
119 Module V Procedure Coupling Weights Using Chen and Lu Method After Increasing the weight of WON in Main Procedure From 3 to 6 ...................... 133
120 Module V Procedure by Procedure Coupling Values Using Chen and Lu Method After Increasing the weight of WON in Main Procedure From 3 to 6 .... 134
121 Module VI Definition Matrix After Decreasing the weight of outstat in write_stat Procedure From 60 to 20 ....................... 136
122 Module VI Procedure to Procedure Coupling Matrix After Decreasing the weight of outstat in write_stat Procedure From 60 to 20 ............ 137
123 Module VI Procedure to Procedure Normalized Coupling Matrix After Decreasing the weight of outstat in write_stat Procedure From 60 to 20 ... 137
124 Module VI Procedures Overall Coupling After Decreasing the weight of outstat in write_stat Procedure From 60 to 20 .................. 137
125 Module VI Procedures Coupling Weights Using Dhama Method After Decreasing the weight of outstat in write_stat Procedure From 60 to 20 ............. 138
126 Module VI Procedure to Procedure Coupling Weights Using Dhama Method After Decreasing the weight of outstat in write_stat Procedure From 60 to 20......... 138
127 Module VI Procedure to Procedure Coupling Weights Using Fenton and Melton Method After Decreasing the weight of outstat in write_stat Procedure From 60 to 20 ............. 138
128 Module VI Procedure to Procedure Coupling Weights Using Offut, Harrold and Kotle Method After Decreasing the weight of outstat in write_stat Procedure From 60 to 20 ....... 139
129 Module VI Procedures Coupling Values Using Chen and Lu Method After Decreasing the weight of outstat in write_stat Procedure From 60 to 20......... 139
## List of Figures

1. **Calculating Definition Matrix from Source Code Procedure** .................................................. 75
2. **Module I Source Code** ........................................................................................................ 153
3. **Module II Source Code** ...................................................................................................... 155
4. **Module III Source Code** ..................................................................................................... 158
5. **Module IV Source Code** ..................................................................................................... 161
6. **Module V Source Code** ...................................................................................................... 167
7. **Module VI Source Code** ..................................................................................................... 171
THESIS ABSTRACT

Name: SADEG HASSAN AHMED AL-NASSER
Title: MEASURING THE COUPLING OF IMPERATIVE COMPUTER PROGRAMS
Degree: MASTER OF SCIENCE
Major Field: INFORMATION & COMPUTER SCIENCE
Date of Degree: JUNE 1997

Measuring the coupling of computer programs is a way to determine the quality of a developed system. Coupling is the level of interactions between the system components. In this thesis, the notion of coupling is introduced, the logical layers and existing measures are surveyed, and a new coupling measure is developed. The new coupling measure determines the elements that affect the coupling between two modules and assigns weights in proportion to their effect on coupling. These weights are fed into a model that automatically calculates the coupling between the system modules.

The new method captures most of the coupling factors. The new method assignment is shown to be of a ratio scale. This method has been evaluated experimentally on six modules from five different programming languages in two integral parts. First, the new method is tested on capturing the observed relation on the module procedures coupling. It satisfies all the relations among the six modules. Second, the new method is tested on capturing the expected changes when six atomic operations are conducted on the corresponding modules. It satisfies five of these expected changes. Finally, the new method outperforms four recent methods on the two parts of the validation.

Keywords: Coupling, Coupling Levels, Cohesion, Cohesion Levels, Software Complexity, Software Quality, Software Measurement, Measurement Scales, Software Metrics, Software Metrics Classification, Coupling Metrics, Metric Development Model Based Methodology, Coupling Model, Coupling Factors, The New Coupling Measure, Software Metrics Validation, Theoretical Validation, Experimental Validation.

King Fahd University of Petroleum and Minerals, Dhahran.
June 1997
خلاصة الرسالة

اسم الطالب الكامل : صادق حسن أحمد الناصر
عنوان الدراسة : قياس الترابط بين إجراءات برامج الحاسب الأولوية
التخصص : معلومات وعلوم الحاسب الآلي
تاريخ الشهادة : يونية 1997م

إن قياس الترابط بين أجزاء برامج الحاسب تعد طريقة من طرق تحديد نوعية النظم المطورة. الترابط هو مستوى التفاعل بين أجزاء النظام. هذه الرسالة تطرق إلى مفهوم الترابط وتشرح مستوى وطرق قياسه المستخدمة. وتطرح طريقة جديدة لقياسها. هذه الطريقة الجديدة تمدد العناصر المؤثرة في الترابط بين وحدتين برامجتين وخصص وزها طبقاً لما ذكرها. هذه الأوزان تُوزَّد بموجز

حسب الترابط بين أجزاء الوحدات البرمجية تلقائياً.

إذا الطريقة الجديدة تحتوي على معظم مؤثرات الترابط. كما إن تخصصها تثبت أنه مدرج نسبياً.

بالإضافة إلى أنها قيمت تجريبياً على ستة وحدات برامجية من خمس لغات حاسوبية مختلفة في حزمتين متكاملتين. أولاً، حسب الطريقة المذكورة على احتمال العلاقات المألوفة في الترابط بين إجراءات الوحدات وثانيًا، حسب الطريقة ذاتها على احتمال التغييرات المتوقعة عندما تُنفذ ست عمليات أولية على الوحدات المعنية. وثانيًا، أُخذ فمن من هذه التغييرات المتوقعة. في النهاية، إن الطريقة المطورة أثبتت أربع طرق جديدة في حزم التقييم.

هفواتي : الترابط، مستويات الترابط، التماسك، مستويات التماسك، تقدير البرامج، جودة البرامج، قياس البرامج، مفاهيم القياس، طرق قياس البرامج، تصنيف طرق قياس البرامج، طرق قياس الارتباط، طريقة الأخذية لتطوير المعايير، نموذج قياس الارتباط عنصر الارتباط، الطريقة الجديدة لقياس الارتباط، تقدير طرق القياس، التقييم النظري، التقييم التجريبي.

درجة الماجستير في العلوم
جامعة الملك فهد للبترول ومواد العازل
الظهران، المملكة العربية السعودية
يونية 1997م
Chapter 1

Introduction

A software development is a costly and time-consuming process. To be able to control the process, we measure some attributes of the developed software. Coupling is one attribute that measures the level of interactions between the system components. In this thesis, coupling and cohesion attributes are introduced, their existing measures are surveyed and a new coupling measure is presented. The new coupling measure calculates weight of each element in all procedures of the system. The new measure feeds these weights to a model that automatically calculates the coupling between the procedures.

1.1 Overview of Module Coupling and Cohesion

A software system may be designed or implemented monolithically or compositely to accomplish some tasks [scha90]. A monolithic system consists of a single monolithic block of code, i.e., a single module. A composite software system is designed to consist of a number of inter-related
modules. A module is a sequence of program statements, bounded by boundary elements, having an aggregate identifier. Two important module attributes influence the system design: module coupling and cohesion [Dham95, Offu93, Scha90, Your79]. Coupling is the degree of interactions between modules, while cohesion is the level of interactions within a module [Your79]. Cohesion may also be defined as the coupling between the elements of a module within that module. A large single module is hard to maintain, understand, extend or enhance, test, and use in the development of other systems [Mart85, Scha90]. On the other hand, dividing the system into many smaller modules that contain one single statement will generate a complex system which is hard to manage. The decomposition should be performed in such a way to maximize cohesion and minimize coupling in order to achieve a well-designed software system [Offu93, Your79].

1.1.1 Coupling Classifications

Myers defined five classes of coupling ordered from worst to best [Mart85, Scha90].

1.1.1.1 Content Coupling

Two modules are content-coupled if one directly references the content of the other, for example:

Example 1:

*Module 1 modifies a statement in Module 2.*

Example 2:
Module 1 refers to a local data of Module 2 by specifying a numerical displacement within Module 2.

1.1.1.2 Common Coupling

Two modules are common-coupled if both modules access to global data, for example:

Example 1:

Using Common in FORTRAN Language.

Example 2:

Declaring variables to take global Scope in Block Structured Languages.

1.1.1.3 Control Coupling

Two modules are control-coupled if one module controls the logic of the other by passing an element of control to it, for example:

Procedure Control
begin

Flag = 1;

Call Set_Input_File (Flag, File1, File2)

Flag = 0;

Call Set_Input_File (Flag, File1, File2)
end

Procedure Set_Input_File(Flag, File1, File2)

begin

if Flag = 0 then

/* read from file 1 */

else

/* read from File 2 */

end

1.1.1.4 Stamp Coupling

Two modules are stamp-coupled if one module passes a composite data structure to the other but the called module only operates on parts of the composite data structure, for example:

Calculate Employee Monthly Salary (Employee_Record).

1.1.1.5 Data Coupling

Two modules are data-coupled if all of the modules parameters are homogenous data items. This means that a parameter is either a simple parameter or a data structure of which all of its elements are used in the called module, for example:

Calculate Employee Monthly Salary (Weekly_Wage).
1.1.2 Cohesion Classifications

Myers defined seven classes of cohesion ordered by a subjective ranking from lowest desired to highest desired [Mart85, Scha90, Your79].

1.1.2.1 Coincidental Cohesion

Definition: A module has a coincidental cohesion if:

a. The module performs multiple completely unrelated functions, for example:

   **Example 1:**

   *Read a string of characters and compute the sum of two numbers.*

   **Example 2:**

   *Print a character string and compute the average of three numbers.*

b. The module function can not be defined, i.e., the module must be described in terms of its internal logic but not its function, for example:

   *Print Next Character, and Multiply the third parameter by 6.*

1.1.2.2 Logical Cohesion

Definition: A module has a logical cohesion if the module performs a series of related functions, one of which is selected by a calling procedure, for example:
Example 1:

\texttt{New\_Operation(function\_code, parameter1, parameter2);}

\texttt{if (function\_code = 5) then}

\texttt{return 1;}

\texttt{else}

\texttt{return parameter1 + parameter2;}

Example 2:

\textit{A module that performs all of the program input, or all of the program output.}

1.1.2.3 Temporal Cohesion

Definition: A module has a temporal cohesion if it performs a series of functions that are related in time, for example:

Example 1:

\textit{Open Old Employee File and New Employee File.}

Example 2:

\textit{Read First Old Employee Record and First New Employee Record.}

1.1.2.4 Procedural Cohesion

A module has a procedural cohesion if:
The module performs a series of functions that are related by the procedure followed by the product, for example:

*Read Employee Record, Increase his salary by 0.1 and Update The Salary File.*

1.1.2.5 Communication Cohesion

A module has a communication cohesion if the module performs a series of functions that are related by the procedure followed by the product all of which operate on the same data, for example:

Example 1:

*Calculate the Employee New Salary and Send it to the Printer.*

Example 2:

*Update Employee Record and Copy it to a flat File.*

1.1.2.6 Information Cohesion

A module has an information cohesion if the module performs a series of related functions, each with its own entry point and independent code, and all work on the same data structure, for example:

*Procedure Integer_Stack*

begin

/* Declaration of an Integer Stack array and element */
Procedure Push(E: integer; var S: Stack)

begin

/* body of Push procedure */

end

Procedure Pop(E: integer; var S: Stack)

begin

/* body of Pop Procedure */

end

/* body of Integer_Stack procedure */

end

1.1.2.7 Functional Cohesion

A module has a functional cohesion if the module performs only one function or achieves a single goal, for example:

Compute The Area of a Triangle.

1.2 Software Complexity Measurements

In this section, software complexity is defined, measurement concepts are introduced and software metrics are classified.
1.2.1 Software Complexity

Software Complexity is the difficulty to maintain, change and understand software [Zuse90]. There are three factors that determine the probability of human errors in any information processing [Your79]:

1. Information needed to understand the system correctly;
2. Information Accessibility; and
3. Information Structure.

These factors can be treated in inter-modular interfaces as follows:
The programmer needs to understand the module function fully by understanding the role of arguments in the module. A simple rule is that the complexity of the module increases as the number of arguments increases. For example, a module with 100 parameters is more difficult to understand than a module with 3 parameters. The developer perceives the interface information to be more complex when it is represented in a nested fashion versus linear fashion. As an example, nested if structure is more complex than a single one level if structure.

Finally, the system analyst needs to understand correctly how the module items are being accessed and what they mean. The four attributes that affect accessibility are [Your79]:

1. Direct/Indirect Access: The interface information is more complex when accessed indirectly by the program. This means simple data items are less complex than a
structured data item carrying the same information.

2. Local/Global Access: The interface information is more complex if it is represented remotely.

3. Standard/Unstandard Access: The interface information is more complex when presented in an unexpected manner rather than standard. As an instance, repressing two points in a standard way \((X_0, Y_0)\) and \((X_1, Y_1)\) is less complex than representing them in \((X_0, X_1)\) and \((Y_0, Y_1)\).

4. Obvious/Obscure Nature: The interface information is less complex when its nature is obvious than obscure. As an example, computing the sum of two numbers with two arguments usually implies that the result will be stored in one of them while commuting the same function with three parameters usually implies that the first two are the input and the last is the output.

1.2.2 Measurement

Measurement is the process of assigning numbers or symbols to the attributes of entities of the world that describe the entities and the attributes according to clearly defined rules [Fent90, Fent94a, Fent94b]. An entity is either an object like a person or an event such as repairing the car. An attribute is a property or a feature of an entity. The assignment of numbers that describe the attribute usually is imprecise since it depends on view points. When we fix a viewpoint, we define a model for our measurement that gets a precise meaning. For example, the height of a
human may have different interpretations. A model for it might be the shortest length from the bottom of the feet to the neck. Clearly, such an assignment must preserve the empirical and initiative observations about the attributes and entities. So, taller humans, for instance, must be assigned larger numbers. These numbers, however, may be different depending on the unit of measurement. Generally, there are two types of measurements: direct and indirect. The direct measurement of an attribute depends only on the measurement of this attribute. The indirect measurement of an attribute may depend on the measurement of one or more attributes [Fent94b]. The particular measurement of an attribute is called a measure or a metric. These two terms are used interchangeably in the literature but they have in other science disciplines precise meanings. A measure is a mapping that assigns measurement values to the attributes of objects. On the other hand, a metric is defined as a criterion to determine the difference between two entities [Zuse90].

1.2.3 Measurement Scales

There are five types of measurement scales: Nominal, Ordinal, Interval, Ratio and Absolute out of which the first four are the most applicable to software measurements [Fent90, Fent94a, Fent94b, Hend96, Shep93, Zuse90].
1.2.3.1 Nominal Scale

It is the simplest and the least restricted. It has only one empirical operation which is the determination of equality. It is symmetric, transitive and reflexive operation. Examples of such a scale are: the football player numbers, the car numbers, and any one to one mapping between object and numbers. Mode, frequencies, and median are non-parametric statistics that are allowed in this scale.

1.2.3.2 Ordinal Scale

It introduces rank ordering by allowing less than and greater than operations. Instances of this scale are hardness (1 - 7), air equality and intelligence tests. Although in this scale it is clear that less ordering is less than high ordering, the difference between the ordering is not measured. The allowed transformation is a monotonic measuring function. The allowed statistical operations are the remaining non-parametric statistics which are median, percentile, kendall τ, Spearman r, and Kendall W in addition to those permitted by nominal scale.

1.2.3.3 Interval Scale

It introduces the possibility of comparing (difference) interval empirically and ratios of any intervals. Time of calendar, temperature measurements in Fahrenheit or Centigrade, and intelligent tests in standard scores are examples of this scale. The limitation of this scale is the definition of
absolute zero. This is why we can not say a temperature of 20 degrees is twice as hot as a
temperature of 10 degrees. However, we can state that the difference between 10 - 5 interval and
20 - 15 interval is 5. Linear transformation with a slope greater than zero is the allowable
transformation in this scale. Mean, standard deviation, Pearson product-moment correlation and
multiple product-moment correlation are parametric statistical tests that are allowed as well as the
statistics which are meaningful to an ordinal scale.

1.2.3.4 Ratio Scale

It solves the definition of absolute zero by letting it to be implied, i.e., it gives a meaning to the
zero value. Mass, time interval, length, and absolute temperature (Kelvin) are instances of ratio
scale. This scale is general. It allows comparing (difference) interval and ratio empirically.
Linear transformation without intercept and with a slope greater than zero is the allowable
transformation in this scale. This scale allows the remaining parametric statistical tests which are
geometric mean and coefficient of variations beside those allowed by interval scale.

1.2.3.5 Absolute Scale

It is the highest level among all scales. It is used for simple counts as lines of codes, number of
alphabetic characters in a word and number of days in a year. Identity which transforms the object
to itself is the only allowed transformation of this scale. Since absolute scale represents counts,
they permit the usual descriptive statistics to be applied like means and standard deviation in
addition to those that are permitted by the ratio scale.

1.3 Software Metric Classifications

There are many software measures in the literature [Cote88, Harr82, Li87, Navl87, Rodr87, Shep88, Shep93]. These may be classified into two primary types: static and dynamic [Li87]. Static measures are computed by static analysis of the source program. Dynamic measures are computed while the program executes. The static measures in turn are divided into three types [Li87]:

1. Volume: Measures the size of software product like counting the lines of codes, statements, I/O formats, procedures and statement per procedure.

2. Data Organization: Measures the usage, visibility and interactions of data. For example measuring the span of a variable and data bindings.

3. Control Organizations: Measures the usage of control structures. For instance, measuring cyclomatic complexity (number of independent logical paths), number of subroutine calls and the number of intersections of a pair of control (Knot Count).

Another classification is to classify software metrics into code or design metrics. Code metrics are the ones derived from the source code. Design metrics are derived from the software structure during the design phase or later. Design metrics are classified into three groups [Harr93, Shep88, Zuse90]:

1. Inter-modular metrics: metrics that measure the inter-relationships between modules.

2. Intra-module metrics: metrics that measure the inter-relationships in a module.

3. Inter- and Intra-modular metrics: metrics that measure the internal and external relationships of a module or a pair of modules.

Software metrics may also be classified according to one of the following classifications [Zuse90]:

1. Automated versus manual measures

2. Quality versus cost measures

3. Product versus process measures

4. Subjective versus objective measures

1.4 Problem Definition

The purpose of this work is to:

1. Develop a new coupling measure that includes all of the cited coupling factors.

2. Use the new measure in a general model that automatically calculates the coupling of subprograms to each other.

3. Validate the new measure in two integral parts

   • Empirical Validation: Test if the new measure captures the observed relations between the coupling of subprograms on sample systems.

   • Atomic Modifications Validation: Test if the new measure preserves the expected
1.5 Overview of the Solution

In order to achieve the objectives specified above, I will proceed as follows:

1. Analyze the existing coupling measures to state their strengths and shortcomings, identify the coupling factors on which they are based and collect all of these factors to be included in the new measure.

2. Determine the system components of which the coupling need to be measured and the elements with which the components interact. The components are subprograms, i.e. procedures or functions, in traditional higher level languages and the elements are the variables, i.e. globals, parameters or returned items.

3. Assign the variable a number according to its contribution to coupling. This assignment should take into considerations all of the identified coupling factors. The listing of these assigned number for each variable among all subprograms is called the definition matrix.

4. Calculate the subprogram to subprogram coupling using a model that takes the definition matrix as its input. The listing of the coupling of all subprograms to each other is called the coupling matrix. This coupling matrix is used by the model to define the following coupling measures:
   - Another subprogram to subprogram coupling
   - Subprogram coupling
• A set of subprograms coupling and
• System coupling

5. Validate and Compare the new coupling measure to four recent coupling measures using six systems from five different languages in two integral parts

• Empirical Validation: The objective of this part is to test the new coupling measure and the four recent measures on preserving the observed relation among the system subprograms coupling on the six selected systems. This is performed in the following steps:
  • Identify the system observed relations among its subprograms coupling
  • Calculate the system subprograms coupling along with the system coupling for all the measures to be compared.
  • Test if the new measure and the four recent measures satisfy the observed relations or not with its associated reason.
  • Summarize the results and find out the compared measures performance.

• Atomic Modifications Validation: The objective of this part is to test the new coupling measure and the four recent measures on capturing the subprograms coupling values changes expected when six atomic operations are conducted on the corresponding system. This is performed in the following steps:
  • List the atomic operations with the corresponding expected changes in coupling values.
  • Calculate the system subprograms coupling along with the system coupling for all the measures for each atomic operation.
• Test if the new measure and the four recent measures satisfy the expected changes or not with its associated reason.

• Summarize the results and find out the compared measures performance.

1.6 Thesis Organization

Chapter 2 presents a literature survey that covers coupling parameters, benefits of measuring coupling and a survey of coupling metrics. Chapter 3 discusses a theoretical model for measuring interaction between objects and surveys our coupling measurements. Chapter 4 shows the new method weight assignments is a ratio scale. Chapter 5 covers the implementation of our metric on six modules and the validation of our metric. The conclusion and future work is discussed in Chapter 6.
Chapter 2

Coupling Metrics

Coupling is an inter-module metric because coupling measures the interdependencies between modules. Four dimensions of module coupling are identified [Lohs84]:

1. The interface size
2. The role of information being passed (Data or Control)
3. The interface passing type (Parameter or Global)
4. The interface item types (Simple or Structured)

Troy and Zweben [Troy81] identified the following attributes that may affect module coupling:

- The maximum, average or total number of interface items between modules
- The number of modules accessing a common interface item.
- The number of modules accessing control interactions with or without OK/Fail flag to indicate the successful or unsuccessful computing of the module.
- The number of simple interface items, structured interface items or both simple and structured interface items used in the current module up to the main module.
Troy and Zweben analyzed the above attributes and concluded coupling is reduced when the following three attributes are reduced:

1. The total number of interface items used.
2. The number of common interface items.
3. The number of control interface items.

Both of Lohse and Troy and Zweben agree that the size of interface items affect coupling. Although they both agree that global, control, structure interface items affect coupling, Lohase took into consideration the factors categories while Troy and Zweben took into considerations the number of modules that access the attributes.

Several coupling measures have been proposed. The following coupling measures would be discussed:

3. Hutches and Basili (1985)
4. Fenton and Melton (1990)
5. Selby and Basili (1991)
7. Chen and Lu (1993)

The above 10 metrics were chosen because they:

1. are static and code metrics that span 15 years.
2. can be automated by using computer programs to calculate them.
3. have been validated on large systems.
4. point out the major coupling attributes and their measurements.

2.1 Henry and Kafura Information Flow Metric

The information flow proposed by Henry and Kafura [Henr81a] is the widely accepted and validated measure. The information flow is a digraph in which a node represents either a procedure or a global data structure and an edge represents either control flow (procedure calls) or a data flow (data structure retrieval or update). Three different flow types have been classified:

1. Local flow, occurs from module M1 to Module M2 if M1 calls M2.
2. Global flow, occurs from module M1 to Module M2 if M1 deposits information into a data structure that M2 retrieves.
3. Indirect local flow, occurs from module M1 to Module M2 when Module M calls both M1 and M2 passing an output value from M1 to M2 or when M2 calls a module M1 which
returns a value used by M2.

For a given procedure, two important attributes have been identified:

1. Fan-in: Is the number of local flows [Henr84], the number of local flows and the retrieved data structures [Henr81a], or the number of local flows and the updated data structure [Henr81b] terminating at the procedure node.

2. Fan-out: Is the number of local flows [Henr84], the number of local flows and the updated data structures [Henr81a], or the number of local flows and the retrieved data structure [Henr81b] emanating from the procedure node.

A module has been defined as the procedures which either directly update or retrieve a global data structure.

The information flow metric of a procedure is defined as the product of the procedure complexity (measured in Lines of Codes) and the square of the product of its fan-in and fan-out.

Procedure information flow = procedure complexity * (fan in * fan out)^2

Many variants of information flow metric were proposed for specific usage [Card88, Kite90, Pick95, Shep89].

The strength or coupling from a module A to another module B was defined as: the sum of the number of procedures exporting information from module A and the number of procedures importing information into module B multiplied by the number of information paths [Henr81a].
The information flow metric is widely accepted measure. Its source is language independent since it relies on a graphic design [Henr84]. It is easily computable from the data flow generated by the compiler by procedure by procedure analysis [Henr81a, Henr84]. It can be used in both the design and implementation [Henr84, Henr81a]. It has significant correlation (r = 0.63) with maintainability [Pick95, Shep89]. Henry and Kafura [Henr81a] found a high correlation (r = 0.95) between the information flow and the number of program changes taken as approximation to program errors. The complexity information obtained by the information flow metric appears to be independent of cyclomatic complexity and software science [Henr81b, Henr84]. It correlates badly with the cyclomatic complexity (r = 0.38) and software science (r = 0.35) while the latter correlates highly (r = 0.84) to each other. Shepperd and Ince listed the empirical validation conducted on information flow [Shep94].

The information flow was used for many purposes [Henr81a, Navl87, Shep88]:

1. Lack of cohesion: Identify procedures performing more than one function. It arises when a procedure reveals a large number of interconnections due to a high fan-in and fan-out.

2. Stress points: Identify the procedures with a high information flow through it.

3. Inadequate Refinement: Identify the procedures with a huge size that needs to be split or with a missing level of abstraction due to a high fan-in or fan-out.

The information flow coupling identifies two of coupling categories [Henr81a]: content coupling and common coupling. Content coupling is identified by a direct reference between the modules
which is equivalent to direct local flow. Common coupling is identified by the sharing of a global data structure which is equivalent to global flow.

However, there are a number of statistical and theoretical problems with information flow. Some of the statistical problems are:

- Henry and Kafura treated the number of modifications as an approximation to errors [Henr81a]. This is not really true because some of modifications are enhancements to the system [Shep94].

- They claim that information flow outperformed the cyclomatic complexity [Henr81a]. This implies that a more extractable metric is more effective than the information flow metric [Shep94].

- They assumed normally distributed data [Shep89].

- They omitted some of the four most complex modules from their analysis [Shep88, Shep89].

- They applied logs and square roots in transforming the data. This causes a loss in a lot of data resulting in a week ordering. It also decreases the number of data points from 165 to 8 resulting in harder to obtain statistical significance and less reliable trends [Shep94].

Moreover, a number of theoretical problems have been identified [Shep89, Shep94]:

- The information flow metric does not have a precise definitions of global data and procedures.
• The information flow indirect local flows can only be detected by the module internal analysis which most unlikely to be available at the design level.

• The information flow indirect local flows definition restricts the levels of system structure to two. However, allowing more than two levels will yield to overcounting, since all modules are linked together.

• The information flow equation will force a single zero term to propagate through and result in an overall measure of zero.

• The information flow considers all flows to be of uniform complexities while the information may be a simple Boolean or a complex structure such as a record or an array.

• The information flow has arbitrarily chosen the power of two with the only justification that the complexity is thought to be a non-linear function.

• The information flow is not consistent because it uses different methods for counting information flows via globals and flows via parameters. The parameters count was based on the number of local flows while the global count was based on the number of accesses of the global data structure.

• The information flow uses the length as a measure of Intra-module complexity. This measure has no empirical evidence and its availability will be late and not at the design phase.

Despite the information flow problems, it was a good trial and encouraged others to develop coupling metrics.
2.2 Lohse and Zweben Metric

Lohse and Zweben [Lohs84] investigated the effect of parameter and global interfaces in the system coupling. They conducted three experiments on student software. They used the programming effort $E$ of the software science proposed by Halstead as a coupling measure. The definition of the programming effort depends on the following quantities among others:

- $n_1$: number of unique operators
- $n_2$: number of unique operands
- $N_1$: total number of operators
- $N_2$: total number of operands
- $n = n_1 + n_2$: vocabulary size
- $N = N_1 + N_2$: program length
- $V = N \log_2 n$: program volume
- $V^*$: potential volume (Volume of the smallest representation of the algorithm)
- $E = V^2 / V^*$: programming effort.

- Lohse and Zweben proposed a coupling measure $C$ of a system $S$ consisting of $m$ modules ($M_1, M_2, ..., M_m$) to be the difference between the effort of the system and the cumulative effort of the constituting modules. Mathematically, $C = E(S) - \sum_{i=1}^{m} E(M_i)$.

Lohse and Zweben analyzed this measure on a system structure consisting of one main
module \( M_0 \) that calls \( m \) modules \( M_1, M_2, \ldots, M_m \). The analysis found that a system using globals will have a lower coupling value than a system using parameters. This finding contradicts the foundation of structural design which states that the use of global variables will produce higher coupling. The main reason favoring the global variables over the parameters in the analysis of Lohse and Zweben is because they based their measure on Halslead's programming effort measure. Halslead's programming effort measure get lower when the number of variables gets lower. This analysis also was conducted on a simple structure and can not be generalized.

2.3 Hutches and Basili Metric

Hutches and Basili proposed a clustering scheme for system modularization based on the data bindings and coupling and cohesion concepts [Hutc85, and Raz93]. They identified a measure called dissimilarity. The main concepts of their scheme are as follows:

A data binding exists between two procedures \( p \) and \( q \) when a variable \( x \) is within the static scope of both procedures. Mathematically, it is an ordered triple \((p, x, q)\) where \( p \) and \( q \) are procedures and \( x \) is a variable exists in the static scope of procedures \( p \) and \( q \). Clustering is grouping a set of procedures into subsystems (modules) to reduce the dependency between any pair of subsystems and increase dependency within a subsystem. A data binding was classified to four classes based in its uses:
1. Potential: The variable x may or may not be referenced or assigned in both of the two procedures p and q.

2. Used: The variable x is used by the two procedures p and q in a reference or an assignment.

3. Actual: Procedure p assigns a value to variable x and another procedure q references variable x.

4. Control flow: Procedure p may possibly passes the control to procedure q. This can happen when there exists a chain of calls from procedure p to procedure q or vice versa. It also can occur when there is a procedure r such that there is a chain of calls from procedure r to procedure p and there is a path connecting the call chain of procedure p with the call chain of procedure q in the directed flow graph of procedure r.

Consider the following portion of code for an example:

Global integer a, b, c, d;

Procedure p1

integer c, x;

c = a + b; /* uses a, b */
a = c + 5; /* assigns a */
d = a * 2; /* assigns d */

... call p2(x);
...
end procedure p1;

Procedure p2(integer e)

integer d

d = a + b + e; /* uses a, b, e */

b = d + 10; /* assigns a */

...

end procedure p2;

Procedure Main

call p1;

end procedure Main;

The following are examples of potential data bindings:

(p1,a,p2), (p1,b,p2), (p1,c,p2), (p1, d, p2),(p1, x, p2)

(p2,a,p1), (p2,b,p1), (p2,c,p1), (p2, d, p1), (p2, e, p1)

However, only the followings are examples of used data bindings:

(p1,a,p2), (p1,b,p2), (p1, d, p2)

(p2,a,p1), (p2,b,p1), (p2, e, p1)

On the other hand, the followings are examples of actual data bindings:

(p1, a, p2), (p1, x, p2)
(p2, b, p2), (p2, e, p1)

Finally, the followings are examples of control flow data bindings:

(p1, x, p2)

(p2, e, p1)

Hutches and Basili defined an \( n \times n \) matrix \( b(i,j) \), \( i, j = 1, ..., n \) to contain the number of control flow data bindings that may exist between procedure \( i \) and procedure \( j \). Mathematically, \( b(i, j) = \) the number of control flow data binding in the form of \( (i, x, j) \) and \( (j, x, i) \) for some \( x \). They also defined a dissimilarity matrix \( p(i, j) \), \( i, j = 1, ..., n \) to contain the number of non-common data bindings among the procedures \( i \) and \( j \) to the total number of data bindings between them.

Mathematically,

\[
p(i, j) = \frac{(\text{sum}_i + \text{sum}_j - 2 \cdot b(i, j))}{\text{sum}_i + \text{sum}_j - b(i, j)}
\]

where:

\( \text{sum}_i \): The number of control data bindings in which \( i \) occurs.

\( \text{sum}_j \): The number of control data bindings in which \( j \) occurs.

\( \text{sum}_i + \text{sum}_j - b(i, j) \): The number of control data bindings in which either \( i \) or \( j \) occur.

\( \text{sum}_i + \text{sum}_j - 2 \cdot b(i, j) \): The number of control data bindings in which either \( i \) or \( j \) occurs but not both.

This dissimilarity matrix is used to group the components with less dissimilarity using an algorithm for clustering [Hutc85] to find the cluster with the minimum dissimilarity values.
The algorithm performs its task in iterations. In each iteration, the two modules (subsystems) with the least dissimilarity are combined. The measurement coupling and cohesion were proposed using the cluster iterations. If several clusters are on level k and they collapse into a single cluster at level k + 1 then the clusters cohesion is k and the clusters coupling is k + 1.

This method seems to be more complex compared to information flow but it restricts the difference between the coupling and cohesion of a cluster to 1. Its outcome depends on the clustering algorithm being used. It does not differentiate between data computation from control uses or parameter values from a global structure [Offu93].

The following problems, also, have been identified:

- It is not general. It only measures the coupling or cohesion of the clusters. It does not measure the coupling or cohesion between pairs of procedures or between a procedure and a set of procedures.

- Its method does not cluster a module with abstract data type and with no local data sharing between the operations of the module. The reason is that there is no direct data bindings between the operations of the module [Hutc85].

- It uses false comparisons since the dominator may vary among pairs of modules. For example, a pair of modules with one non-common data binding out of a total of two data bindings seems more dissimilar than a pair of modules with four non-common data bindings out of a total of nine [Raz93].

- It lacks the monotonic property as it permits a dissimilarity value at a later iteration to be
smaller than all the dissimilarity values in the previous steps which results in a invalid clusters [Raz93].

2.4 Fenton and Melton Metric

Fenton and Melton [Fent90] proposed an ordinal scale coupling measure. They refined Myers coupling classification in order to identify them programmatically and they assigned them numbers as follows:

<table>
<thead>
<tr>
<th>Coupling Type</th>
<th>Assigned Number</th>
<th>Modified Definition Between Modules x and y</th>
</tr>
</thead>
<tbody>
<tr>
<td>Content</td>
<td>5</td>
<td>Module x refers to the inside of module y, i.e., it branches into, changes data, or alters a statement in y.</td>
</tr>
<tr>
<td>Common</td>
<td>4</td>
<td>Modules x and y refer to the same global data.</td>
</tr>
<tr>
<td>Control</td>
<td>3</td>
<td>Module x passes a controlling parameter to y, i.e., the parameter is a flag.</td>
</tr>
<tr>
<td>Stamp</td>
<td>2</td>
<td>Modules x and y accept the same record type as a parameter.</td>
</tr>
<tr>
<td>Data</td>
<td>1</td>
<td>Modules x and y communicate by parameters, each of which is either a single data or homogenous structure that does not incorporate control element.</td>
</tr>
<tr>
<td>No Coupling</td>
<td>0</td>
<td>Modules x and y have no communication, i.e., are totally independent.</td>
</tr>
</tbody>
</table>

*Table 1. Fenton and Melton Modified Definitions for Myers Coupling Levels.*

They proposed a model for studying coupling that represents a design. The model chosen is a directed labeled multi-directed graph with possibly many arcs between the nodes. The label is an ordered pair of two components. The first component represents the coupling level. The second component represents the number of times the annotated type of coupling occurs between the nodes.
Their coupling measure depends on the maximum coupling level existing between any two modules and the number of interconnection of coupling type between the pair of modules. Specifically, the coupling of any pairs of modules $x$ and $y$ $C(x, y)$ is defined as the sum of the coupling level and the ratio of the number of their interconnections to the number of their interconnections plus 1. Mathematically, $C(x, y) = i + (n / (n + 1))$ where $i$ is the level of the greatest coupling type and $n$ is the number of interconnections between $x$ and $y$.

Harrison [Harr93] proposed to use this coupling metric to measure the coupling values in functional programs. He assumed, however, that the coupling type for most pairs of functions written in functional languages is data coupling. Therefore, he assumed the level $i$ of any pair functional procedures is 1.

Fenton and Melton also proposed a global coupling measure. This is a coupling measure of the system or the connectivity of the module design chart. They proposed to use the median of all pairwise module couplings to measure the system global coupling.

This measure posses the following problems:

- It does not present a way to determine the coupling levels automatically. They used hand-derived coupling measure.

- It considers all interconnections to be of uniform complexities while the information may be a simple Boolean or a complex structure such as a record or an array.
• It does not calculate the coupling of a module to a set of modules.

2.5 Selby and Basili Metric

Selby and Basili [Selb91] proposed a coupling and cohesion measures to identify the error prone system structure based on data bindings. They used a set of five tools to calculate the potential data bindings for each procedure or function in the system. They used an interactive clustering algorithm to produce the system hierarchical description. The system description consists of the clusters produced where a large cluster may contain a number of smaller clusters. The process is iteratively producing larger and larger clusters until the elements are collapsed into a single cluster. The selection of elements to cluster is based on the greatest interaction, in terms of potential data bindings. A procedure or function may be contained in more than one cluster.

An attribute of a cluster in a subsystem was identified as ratio of the number of potential data binding between the routines within and outside the cluster to the number of the potential data binding within the cluster. This attribute was interpreted as a ratio of the coupling of the cluster with other clusters in the subsystem to the internal strength of the cluster. In other words, this number captures the coupling and cohesion ratio for a cluster within a subsystem.

This measure was used to analyze a system with 77 subsystems. The software errors data were collected from a high level design through system test and from some field operation of the
system. It was found that routines with high coupling and cohesion ratios had 7.0 times more errors per 1000 source statements excluding comments than did routines with the lowest ratios.

This measure outcome depends on the clustering algorithm being used. It does not differentiate between data computation from control uses or parameter values from global structure [Offu93]. The following problems, also, have been identified:

- It measures only the coupling of the clusters. It does not measure the coupling or cohesion between pairs of procedures or between a procedure and a set of procedures.
- Its method does not cluster a module with abstract data type with no local data sharing between the operations of the module. The reason is that there is no direct data bindings between the operations of the module [Hutc85].

2.6 Offut, Harrold and Kotle Metric

Offut, Harrold and Kotle [Offu93] proposed a general module coupling metric based on data binding, data slicing and coupling levels. They redefined Myers coupling classifications to enable automatic quantification, added other classes, and assigned a number to each class as follows:
<table>
<thead>
<tr>
<th>Coupling Type</th>
<th>Assigned Number</th>
<th>Modified Definition Between Modules x and y</th>
</tr>
</thead>
<tbody>
<tr>
<td>Content</td>
<td>6</td>
<td>Modules x and y access and change each other's internal data state or procedural state.</td>
</tr>
<tr>
<td>Common</td>
<td>5</td>
<td>Modules x and y refer to the same global data.</td>
</tr>
<tr>
<td>External</td>
<td>4</td>
<td>Module x and y communicate through an external medium such as a file.</td>
</tr>
<tr>
<td>Control</td>
<td>3</td>
<td>Module x passes a flag value that is used to control the internal logic of module y.</td>
</tr>
<tr>
<td>Stamp</td>
<td>2</td>
<td>Modules x and y pass data through a parameter which is a record.</td>
</tr>
<tr>
<td>Data</td>
<td>1</td>
<td>Modules x and y pass data through scalar or array parameters.</td>
</tr>
<tr>
<td>Independent</td>
<td>0</td>
<td>Modules x and y have no coupling between them.</td>
</tr>
</tbody>
</table>

*Table 2. Offut, Harrold and Kotle Modified Definitions for Myers Coupling Levels.*

They added six coupling categories to Myers coupling classes:

1. Call coupling: Occurs when a module x calls a module y without passing any arguments or sharing any data.
2. Stamp coupling: Occurs when data flow from a module x that calls a chain of intermediate modules in which data is passed to the last module y. This coupling is defined on multiple modules.
3. Stamp/control coupling: Occurs when module x calls a module y passing a data structure used as a control.
4. Stamp/data control coupling: Occurs when module x calls module y passing a data structure containing some data items used as controls.
5. Non-local coupling: A common coupling in which a variable is used by a number of modules.
6. Global coupling: A common coupling in which a variable is used by all the modules in the
Their coupling measure is similar to Fenton and Melton with a slight modification. The coupling measure depends on the maximum coupling level existing between any two modules and the number of interconnection of coupling type between the pair of modules.

\[ C(x, y) = i + (n - 1 / (2 * (n + 1))) \]

where \( i \) is the level of the greatest coupling type and \( n \) is the number of interconnections between \( x \) and \( y \).

Offut, Harrold and Kotle coupling metric differs from Fenton and Melton coupling measure by subtracting 1/2 from Melton and Fenton measure. The subtraction is done to emphasize that the effect of the number of interaction by intitution should be closer to level \( i \) more than to level \( i + 1 \). Specifically, the coupling \( C(x, y) \) of any pairs of modules \( x \) and \( y \) is defined as the sum of the coupling level and the ratio of the number of their interconnections minus 1 to twice of one plus the number of their interconnections.

This coupling metrics added the following contributions to module coupling metrics:

- has general applicability. It measures the coupling between any pair of modules, between a module and a set of modules or for the entire system.
- offers a precise definitions of coupling levels so they can be determined algorithmically.
- offers bi-directional coupling measurement: In and out coupling. The in coupling of two module \( x \) and \( y \) takes into account the number of interconnections for call from module \( x \) to \( y \) while the out coupling takes into account the number of interconnections after the
return from y and x.

- adds six levels of coupling to Myers coupling classes.

This metric, however, has the following problems:

- does not differentiate between arguments in the same class in terms of scalar types and structure types.
- does not incorporate the lower coupling levels that the module is possessing.

2.7 Chen and Lu Metric

Chen and Lu [Chen93] proposed a new coupling and cohesion metric for object oriented design as part of eight complexity metrics:

1. Operation complexity metric
2. Operation argument complexity metric
3. Attribute complexity metric
4. Operation coupling metric
5. Class coupling metric.
6. Cohesion metric.
7. Class hierarchy metric
8. Reuse metric

The operation, i.e., procedure inside a class (module consisting of more than one procedure), coupling metric is defined as the summation of the followings:
1. The number of operations which access (calls) other classes.

2. The number of operations which are accessed by other classes.

3. The number of operations which are co-operated with other classes. The co-operation means that an operation accesses some other class operation and vice versa.

The class coupling metric is defined as the summation of the followings:

1. The number of accesses (calls) to other classes.

2. The number accesses by other classes.

3. The number co-operated classes. The co-operation means that a class accesses some other classes and vice versa.

The cohesion metric is defined in terms of the following:

1. N: Is the number of operations.

2. M: Is the number of disjoint sets formed by the intersections of the N operations sets of arguments.

The cohesion is defined as the ratio of M over N.

Cohesion = M/N * 100%

The lower the value, the more cohesive the operations is in the class.

These above dimensions were evaluated in an experiment and it was concluded that the following four metrics contribute significantly to the class complexity:

1. Operation complexity metric

2. Operation argument complexity metric

3. Operation coupling metric
4. Class coupling metric

As can be seen the coupling metrics contribute 50% of class complexity.

2.8 Raz and Yaung Metric

Raz and Yaung [Raz93] proposed a coupling metric that measures the strength of the linkage between processes. This is used as a basis for grouping related processes to reduce and control the amount of effort needed to measure and manage the processes in an organization. In their model, a link between process i and process j exists when there is a flow of data, material, equipment, or any other deliverable between the two processes. Process i is called the supplier and process j is called the customer. There are three possible linkages between two processes i and j:

1. No link: The two processes are mutually independent.
2. One link: There is either a link from process i to process j or a link from process j to process i.
3. Two links: There is a link from process i to process j and a link from process j to process i.

Considering a set of n processes identified by an integer in the range 1 - n, their linkage matrix is an n x n matrix M that captures the binary linkage values between any two processes. Each row represents a process and its role as a supplier. Each column, on the other hand, represents a process and its role as a customer. Formally M = m_{ij} for i, j = 1, ..., n where:
- $m_{ij} = 1$, if process $i$ involves sending a deliverable to process $j$
- $m_{ij} = 0$, if process $i$ does not involve sending a deliverable to process $j$
- $m_{ii} = 0$, by definition since the interest is in the linkages across processes, we ignore the deliverables which are internal to a process.

The linkage matrix is modified when two clusters $i$ and $j$ being joined into one cluster $k$ by:

1. Adding row $i$ and $j$ to form the $k$ cluster column
2. Adding column $i$ and $j$ to form the $k$ cluster column
3. Deleting the clusters $i$ and $j$ rows and columns

As a result, a row and column in the linkage matrix may represent either a simple one process cluster or cluster of more than one process. The matrix elements correspondingly represents the number of deliverables between the two clusters.

The coupling metric is defined as the ratio of the number of actual links between clusters of processes to the number of possible links between the two. The application of this metric has three different situations depending on the number of procedures in the two clusters $i$ and $j$:

1. Simple: The number of processes in each of two cluster $i$ and $j$ is only one. This implies that the number of actual links may be 0, 1, or 2 and the number of possible links is 2. So, the coupling metric may take the value of 0, 1/2, or 1. The calculation of the coupling metric value denoted by $C(i, j)$ from the linkage matrix $M$ is $C(i, j) = (m_{ij} + m_{ji}) / 2$. 


2. Intermediate: The number of processes in the first clusters i is \( n_i \) and in cluster j is one process. The number of possible links from process j to any one of the cluster i's individual processes is \( n_i \). It is also the number of possible links from any of cluster i's individual processes to cluster j. This implies that the number of possible links is \( 2n_i \). The calculation of the coupling metric value denoted by \( C(i, j) \) from the linkage matrix \( M \) is \( C(i, j) = \frac{(m_{ij} + m_{ji})}{(2 \times n_i)} \).

3. Complex: The number of processes in the first clusters i is \( n_i \) and in cluster j is \( n_j \) processes. The number of possible links from each of cluster i process to any one of the cluster j's individual processes is \( n_j \). Because there are \( n_i \) processes in cluster i, the total number of links between clusters i and j is \( n_i \times n_j \). This is also the number of possible links from any of cluster i's individual processes to cluster j. This means that the total number of possible links is \( 2 \times n_i \times n_j \). The calculation of the coupling metric value denoted by \( C(i, j) \) from the linkage matrix \( M \) is \( C(i, j) = \frac{(m_{ij} + m_{ji})}{(2 \times n_i \times n_j)} \).

A cluster cohesion metric is also proposed as the ratio of cluster actual links to the number of possible links in the cluster.

2.9 Rotenstreich Metric

Rotenstreich [Rote94] proposed a coupling metric that measures the potential coupling between a program and other modules in the environment. He developed bounds on the sum of all coupling in the program. He also defined the design density measure as well as analytic potential coupling.
He eliminated content and control coupling classes out of Myers six classes. In his opinion, current programming languages made content coupling impossible to occur. Control coupling was ignored because control coupling is difficult to automate and it is reflected as part of the remaining four coupling types. As a result, Rotenstreich assumed that a program may have common, external, stamp and data couplings.

Rotenstreich developed his analytic coupling sum measure as follows:
Suppose there is a fixed measure associated with each type of coupling. Let $C_d$, $C_s$, $C_e$, $C_c$ be measures of data, stamp, external and common couplings respectively. The coupling sum of a program $P$ with $n$ modules denoted by $C(P)$ is defined as the sum of coupling values between all pairs of modules in $P$. Mathematically, $C(P) = \sum_{i=1}^{n} \sum_{j=1}^{n} C_{ij}$ where $C_{ij}$ denotes the coupling value between module $i$ and module $j$ in program $P$.

The coupling sum was analyzed and found it lies between $(n-1)$ times the value of data coupling and $(n) (n - 1) / 2$ times the value of common coupling. Mathematically, $(n - 1) * C_d \leq C(P) \leq (n * (n - 1) / 2) * C_c$. By further analysis, the coupling sum was found to equal the common coupling value times $n$ raised to the power in the range of 1 - 2 exclusive. Mathematically, $C(P) = C_c * n^{1+x}$ where $0 < x < 1$, $x$ is called the design density of the program $P$.

The design density is bounded. It can be used to compare the design densities of two programs regardless of their sizes.

The coupling sum was applied to 29 C programs that include a compiler, a text editor and network utilities. It was calculated using the assignments: $C_d = 2$, $C_s = 3$, $C_e = 4$ and $C_c = 5$. 
Four other metrics were also calculated. It was found that changing the number of assignment does not significantly change the relationships between the coupling sum and the other measures. Moreover, he proposed another coupling metric, called potential coupling, that measure the coupling between a program and its environment. He developed his analytic potential coupling denoted by cp as follows:

Suppose a program consists of n modules. Let E be the number of modules in the environment and the number of modules in the program P. So, there are E - n modules in the environment. Assuming that every module in the program has a constant potential weight w, the potential coupling was derived to be the potential weight times the number of modules in the program n times the number of modules in the environment E - n. Mathematically, \( cp(E, n) = w \cdot n \cdot (E - n) \).

He incorporated both coupling measure into one called the total coupling sum, denoted by TCS. The total coupling sum of a program was defined as the sum of coupling sum and its potential coupling. Mathematically, \( TCS(n) = C(P) + pc(E, n) = C_c \cdot n^{1+x} + w \cdot n \cdot (E - n) \) where \( 0 < x < 1 \) and \( w < C_d \). The total sum of a program is found to achieve its maximum values of n in the open interval \( (E/2, E) \).

2.10 Dhama Metric

Dhama [Dham95] proposed a coupling and cohesion measure each with four categories. He defined a module as a compilation unit of code. Therefore, a module may contain one or more
procedures or functions. He calculated the coupling and cohesion measures among 15 Ada modules and they agree with the evaluation of the modules. His metric assigns high number to indicate low coupling. The coupling of a number of modules was defined as the average of the constituent modules.

His coupling metric identified four categories:

1. Data flow: It is caused by the interface parameters.
2. Control flow: It is also caused by the interface parameters.
3. Global flow: It is caused by global variables.
4. Environmental: It is caused by calling and being called by other modules.

Dhama proposed the following qualitative coupling model:

Let $C$ be the total module coupling

For data and control categories define:

$i_1$: The input data parameters

$i_2$: The input control parameters

$u_1$: The output data parameters

$u_2$: The output control parameters

For the global category define:

$g_1$: Number of global variables used as data

$g_2$: Number of global variables used as control
For the environment category define:

\[ w: \text{Number of modules called} \]

\[ r: \text{Number of module calling the module under consideration.} \]

Dhama tied all of the above attributes into one called \( m \) and defined as:

\[ m = i_1 + q_1 i_2 + u_1 + q_2 u_2 + g_1 + q_3 g_2 + w + r \quad \text{where} \]

\( q_1, q_2, q_3 \) are constants and they are assumed to be 2. These assumptions may change when the model is fully evaluated.

The module coupling now is taken to be inversely proportional to \( m \), i.e.,

\[ C \propto \frac{1}{m}, \text{ which implies that } C = \frac{K}{m} \text{ where } K \text{ is constant.} \]

Assuming \( K = 1 \), we have \( C = 1/m \).

This metric was calculated manually for two different modules achieving the same function. The function was sorting an array of numbers. The first module conducts the sorting in one procedure while the second implements the function using four procedures. It is found that the coupling of the second module is higher than the coupling of the first even though it has four interfaces. Also, the coupling of each of four procedures was found to be higher than the coupling of the first procedure. This may be due to the fact that each of the four procedures has a higher functional cohesion. This result agrees with intuition that four smaller modules are easier to understand than a single monolithic module.
2.11 Motivation of the Study

- Current coupling metrics are not general. They measure coupling without giving insight into the cohesion that relates to coupling. The cohesion can be estimated by measuring the coupling of the module with itself.

- Most coupling measures are not based on a theoretical model. A theoretical model will allow the measure to be tested and verified. It may be used as the basis for comparing and qualifying different coupling metrics. Current coupling metrics include those proposed by Offut, Harold and Kotle [Offu93], Raz and Yaung [Raz93], Rotenstreich [Rote94], Fenton and Melton [Fent90].

- Most coupling metrics do not incorporate all of the items affecting the coupling between module pairs discussed in the literature, although each of these items has an impact of coupling as discussed in [Chen93, Dham95, Mart85, Lohs84, Offu93, Shep89, Shep94, Troy81, Your79].

2.12 Objective of the Study

The objectives of the study are to:

- Develop a general coupling metric that estimates the measurement of cohesion along with it.

- Develop, enhance or use a theoretical model of coupling measurement to be used as the
basis of comparisons between different coupling metrics.

- Incorporate all items affecting the coupling of a pair modules in a metric that consolidate all of these items in a single measure.
Chapter 3

Model Based Coupling Metric

Shepperd and Ince [Shep93] proposed a model based methodology for metric development that consists of six stages:

1. Problem identification: The developer determine the goal of the metric with appropriate domain.

2. Informal model: Identifies the important factors affecting the metric through intitutions and software engineering knowledge.

3. Formal model and axiomations: Transforms the informal model to a formal model by defining a mapping between them. The mapping should satisfy the axioms governing the relationships between the informal model variables and the formal model inputs, parameters and outputs.

4. Theoretical evaluation: Concerns with making sure that the metric satisfies a theoretical unified framework of axioms like those proposed by Prather or Weyuker.

5. Empirical evaluation: Tests the hypothesis under investigation with a large scale commercial professional software that employee valid statistics.
6. New models/hypothesis: By repeated utilizations of the metric in some applications, new important hypothesis may be discovered. The model may be enhanced to capture these hypothesis. This may initiate re-examining and changing one or more of the previous stages.

The first three stages of this methodology is followed in developing the coupling metric presented in this thesis. The theoretical evaluation may not be applied because these axioms are not in accordance because some of them require a ratio scale and some others reject ratio scale [Hend96, Zuse90]. Also some of these axioms may only apply to specific class of software metrics [Shep93, Hend96]. Showing the type of scale of the proposed measure is treated as the theoretical validation.

The empirical evaluation for any metric should be first tested on small scale professional systems, because a human developer can capture their empirical relations which can be verified in the numerical relational system. This metric is validated by comparing it with four other metrics on how they capture the empirical relations. The other type of empirical evaluation is to compare the proposed method with four metrics on how they are sensitive on atomic modification operations. The type of scale determines the valid statistics that may applied.

The final stage can be followed when the metric is applied on a sufficient number of software systems to assist in recognizing new empirical relations.
3.1 Problem Identification

Coupling metrics suffer from one or more of the following drawbacks:

1. They may not be general. This means that they do not measure the coupling between any pair of modules or between a module and a set of modules or for the system at large.
2. They may treat the data elements equally. The data element, however, may be a simple flag or multi-dimensional data structure.
3. They may not incorporate the purpose of the data item. Obviously, the coupling between two modules increases if the element is being used as a control than if it is being used as data.
4. They may not incorporate the type of the element being used. They should at least differentiate between the non-homogenous types and homogenous or scalar types.

3.2 Measuring the Coupling between Objects Model

This is the third stage in Sheppered and Ince methodology. This research proposes a coupling metric to eliminate or reduce the above problems. The proposed coupling metric utilizes a general model to measure the interactions between objects [Algh94, Can84, Can90].

In real world, a system consists of a number of components. These components has to interact or be coupled to form the overall system. Measuring the interactions of system components enables
us to [Algh94, Algh95, Dham95, Offu93, Selb91]:

- Control the system.
- Achieve better system allocation and utilization.
- Understand the system better.
- Minimize the errors caused by these interactions.

A real world system is represented as a collection of objects grouped together to form the system. An object consists of a number of elements. The objects are bounded in some way. Their bound strength varies from week to strong. The four identified types of bonds are [Algh95]:

1. Shared External Resources: These resources are part of environment the are used by the system.
2. Interface Resources: These are resources that facilitate the communication and movement of information between different components of the system.
3. Common Elements: These elements are part of two or more objects.
4. Inclusions: These objects are totally included in another object.

These types can be repeatedly decomposed until the system elements are simple and can not be factored out [Algh95].

The above information can be automatically captured in a definition matrix \( D = d_{ij} \) in which, each row represents a component, each column represents a basic element or a property of the component, and \( d_{ij} \) represent the weight of the \( j^{th} \) element in the \( i^{th} \) component [Algh94, Algh95,
The coupling matrix is an $m \times m$ matrix that relates each pair of components. The coupling coefficient between two components $i$ and $j$ is calculated as follows:

$$c_{ij} = \alpha_i \sum_{k=1}^{n} d_{ik} \beta_k d_{jk}$$

for $i$ and $j = 1, \ldots, m$ where:

$d_{ik}$ and $d_{jk}$ are elements of the definition matrix,

$$\alpha_i = \frac{1}{\sum_{j=1}^{n} d_{ij}}$$

is the definition matrix $i$th row sum inverse and

$$\beta_k = \frac{1}{\sum_{k=1}^{m} d_{ik}}$$

is the definition matrix $j$th row sum inverse.

The $c_{ij}$ entry measures the coverage extent of $i$th component in the $j$th component. The $c_{ii}$ entry measures the separation extent of $i$th component from other components. The coupling matrix possess the following properties [Algh94, Algh95, Can84, Can90]:

1. Its element is in the range of 0 and 1 inclusive, i.e., $0 \leq c_{ij} \leq 1$.

2. The sum of each row is equal to one, i.e., $\sum_{j=1}^{m} d_{ij} = 1$ for $i = 1, \ldots, m$.

3. If $c_{ij} = 0$ then $c_{ji} = 0$, i.e., $c_{ij} = 0 \Rightarrow c_{ji} = 0$.

4. If $c_{ij} > 0$ then $c_{ji} > 0$, i.e., $c_{ij} > 0 \Rightarrow c_{ji} > 0$.

5. $c_{ij}$ may or may not equal to $c_{ji}$, i.e., $c_{ij} = c_{ji}$ or $c_{ij} \neq c_{ji}$. 
6. $c_{ii} = c_{ij} = c_{ji} = c_{jj}$ if and only if $d$ are identical.

$C_{ii}$ measures the extent to which procedure $i$ is covered by itself with respect to how many and how much other procedures in the same row covers $c_{ii}$. By dividing entries of row $i$ of the coupling matrix will make $c_{ii}$ equal to 1 which indicates exact coverage. Therefore, $c_{ii}$ is treated as a normalizing factor of entries of row $i$. Four coupling measures can be derived from the coupling matrix taking into account $c_{ii}$ as a normalization factor for row $i$ [Algh94].

First, the coupling between any two procedures $i$ and $j$ where $i$ is not equal to $j$ denoted by $\Psi_{ij}$. $\Psi_{ij}$ is the entry $c_{ij}$ in the normalized matrix. $\Psi_{ij}$ may be calculated by:

$$\Psi_{ij} = \frac{c_{ij}}{c_{ii}}$$

$\Psi_{ij}$ measures the interaction between procedure $i$ and procedure $j$. This coupling can be listed in the normalized coupling matrix. The normalized matrix omits the original coupling matrix restriction to compare the entries in the same row, thus allows comparisons across row boundary.

Second, the coupling of procedure $i$ denoted by $\Psi_i$ is the sum of extents of coverage for all procedures interacted with procedure $i$ except itself. $\Psi_i$ is calculated by the following formula:

$$\Psi_i = \sum_{j=1,...,n} (c_{ij} / c_{ii}) + \sum_{i=1,...,n} (c_{ji} / c_{jj})$$

$\Psi_i$ is the sum of the $i^{th}$ row and the $j^{th}$ column of the coupling matrix excluding $c_{ii}$. The $i^{th}$ row sum identifies the extent to which other procedures cover procedure $i$. The $i^{th}$ column sum signifies the extent to which procedure $i$ covers the other procedures in the system.
Third, the coupling of a set of procedures (a subsystem) is the average of the coupling of its constituent procedures. Let $S$ be a subset of the set of all procedures in the system $P$. Let $k$ be the number of procedures in set $S$ and $s_i$ is the number of the procedure in the system. The coupling of the subset $S$ denoted by $\Psi_s$ is calculated by:

$$\Psi_s = \frac{\sum_{i=1}^{k}(\Psi s_i)}{k}$$

$\Psi_s$ denotes the extent to which the subsystem procedures are connected together.

Finally, the overall coupling of the system denoted by $\Psi$ is the sum of the coupling of all system procedures over the number of procedures in the system. The system coupling is calculated by:

$$\Psi = \frac{\sum_{i=1}^{n}(\Psi_i)}{n}$$

$\Psi$ shows the extent to which the system procedures are connected to each other.

Clearly, when the objects and elements are defined and when the weight for the elements in each object is defined then this method can be used to automatically measure the coupling between any pair of objects [Algh94, Algh95]. So, the problem is, given a program written in a chosen programming language, calculate the coupling value for each two modules. To solve the problem using the above model the followings are needed:

- Determine the objects and elements of interest to the problem.
- Determine a method to assign a coupling weight for each element in an object.
- Analyze programs written in imperative languages manually, generate the definition
matrix, and feed it to the model to calculate the coupling matrix.

In this case the objects are modules, specifically, procedures and functions. The elements are distributed in the four elements types as follows [Algh95]:

1. External: Files, External functions, Dynamic link libraries, Directories, Semaphores, Condition variables and Shared variables.
2. Interface: Pipes, Sockets and Parameter lists.
4. Inclusion: A procedure that is defined inside another procedure. In this case the non-local elements are identified for the procedures that uses them.

3.3 Proposed Coupling Metric

To develop a new coupling measure, we need to identify the items that affects the coupling between two modules and the weights assigned to these items based on their contribution to the coupling.

The coupling between two modules is through the:

1. parameters;
2. global data; or
3. Inclusion
The different factors affecting coupling weight found in the literature are:

1. The number of interface items. [Dham95, Mart85, Troy81, Your79]
2. The items access purpose. (Data, Control) [Dham95, Mart85, Troy81, Your79]
3. The items passing scope (Parameter, Global). [Dham95, Mart85, Your79]
4. The interface item type (Non-Structure, Structure). [Lohs84, Offu93, Shep89, Shep94, Your79]
5. The interface access direction (Input, Output) [Dham95, Lohs84]
6. The number of calls of other modules in module. [Chen93, Dham95]
7. The number of calls for the module from other modules. [Chen93, Dham95]
8. Return items. [Offu93]

The following table shows the results of the comparisons of above factors among the most recent coupling measures surveyed [Chen93, Dham95, Fent90, Offu93]:

<table>
<thead>
<tr>
<th>Empirical Condition</th>
<th>Dham95</th>
<th>Offu93</th>
<th>Chen93</th>
<th>Fent90</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Number of Interface items</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2. Items Access Purpose (Data, Control)</td>
<td>x</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3. Items Scopes (Parameter, Global)</td>
<td>x</td>
<td></td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>4. Items Types (Non-Structure, Structure)</td>
<td></td>
<td>x</td>
<td></td>
<td>x</td>
</tr>
<tr>
<td>5. Item Access Direction (Output, Input)</td>
<td></td>
<td></td>
<td></td>
<td>x</td>
</tr>
<tr>
<td>6. Number of Times of Calls in the Module</td>
<td>x</td>
<td></td>
<td></td>
<td>x</td>
</tr>
<tr>
<td>7. Number of Times of Calling the Module</td>
<td>x</td>
<td></td>
<td></td>
<td>x</td>
</tr>
<tr>
<td>8. Return Items.</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 3. Comparison between the Four Recent Coupling Measures among the Eight Identified Coupling Factors.

This table shows that Dhama measure includes most of the empirical conditions in measuring the
coupling. Dhama assumes the following weights for the following coupling factors:

- **Element access scope:** The global access scope weight is the same as the parameter access scope weight. They are implied to be 1.

- **Element access purpose:** The control access purpose weight is 2 while the data access purpose weight is 1.

- **Element access direction:** The output access direction weight is the same as the input access direction. They are implied to be 1.

However, the other three recent measures which are based on Myers coupling levels did not assign quantitative weights to the above factors. They assigned the worst qualitative coupling level identified between the two modules.

Although, the element type is not counted in Dhama measure, some measures [Chen93, Muns93] assigned the element weights according to their types.

Munson and Kohshgoftaar [Muns93] develop a measure of modules data structure. They assigned the weights based on the amount of information that a compiler should create and manage in order to implement a particular data structure while compiling a program. On the other hand, Chen and Lu [Chen93] assigned the attributes fixed numbers or ranges of numbers when they measure the module arguments attribute complexity.
Munson and Kohshgoftaar [Muns93] assigned all of the above types a fixed value of 2. Chen and Lu [Chen93] assigned Boolean and integer types the value 0, character type the value 1 and real type the value 2.

Munson and Kohshgoftaar [Muns93] assigned the array type the value $2 + 2 \times n + \text{element type value}$ where $n$ is the number of dimensions of the array. Chen and Lu [Chen93] assigned the array type a value in the range of 3 to 4. The range of value is specified for the array type to assign different array dimensions different values within the range.

Munson and Kohshgoftaar [Muns93] assigned the structure type the value $2 + \text{summation of the elements type values}$. Chen and Lu [Chen93] assigned the structure type a value in the range of 6 to 9. The range of value is specified for the structure type to assign different structure types different values within the range. Munson and Kohshgoftaar [Muns93] did not assigned the file type a value. Chen and Lu [Chen93] assigned the file type a fixed value of 10.

Munson and Kohshgoftaar [Muns93] assigned the pointer type a fixed value of 2 as any other scalar type. Chen and Lu [Chen93] assigned the pointer type a fixed value of 5. However, Myers coupling classification which is widely accepted qualitative measure differentiates between structures and unions versus arrays and simple types. I recommend to differentiates between structure and non-structure types. This is summarized in table 4.
<table>
<thead>
<tr>
<th>Element Type</th>
<th>Muns93</th>
<th>Chen93</th>
<th>Dham95</th>
<th>Thesis</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Integer</td>
<td>2</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Char</td>
<td>2</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Enumeration</td>
<td>2</td>
<td></td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>Real</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Pointer</td>
<td>2</td>
<td>5</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Array</td>
<td>2 * Number of indices + Element type value</td>
<td>3-4</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Structure</td>
<td>2 * Number of levels + Average element type value</td>
<td>6-9</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>File</td>
<td></td>
<td>10</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

*Table 4. Element Type Weight Comparison between Three Methods and the Proposed Method.*

The Dhama method is used to measure the coupling of Ada programs where the item access direction is specified in the procedure or function header. But this information may not stated directly as in C [Sch90] in the function header. It can be deduced from the element type. When the element type is a structure or an array or a pointer to a scalar, the access direction is assumed to be In and Out. Otherwise, it is assumed to be In only.

I believe the assignment of weights should be proportional as it can be to the type of coupling the interface item contribute in the Myers levels. Looking at his levels, he assumes global items weights more than control items and control items weights more than stamp items. However, he did not explicitly assigned a number as a weight for the above coupling types. In his classification, two modules may have more than one coupling type. In this case, he proposed to assign the worst qualitative coupling level identified between a pair of modules.

The coupling measure in this thesis captures all of the possible coupling types recognized between module pairs. This suggests assigning the elements weights according to their combination of
contribution in Myers level dimensions.

Rotenstreich [Rote94] stated that the current programming languages made content coupling impossible to occur. So, the only remaining coupling classes are common, control, stamp and data. Common coupling is caused by global data. Control coupling is caused by the use of the interface item to control the logic of one or the two pair of modules. Stamp coupling is caused by non homogenous interface data items, i.e., structures and unions. Data coupling is caused by using homogenous or simple interface items. Equivenlantly, data coupling is caused by not using any of globals, control or stamp interface items. The data coupling is the lowest coupling between any pair of modules.

The above categorization can be used as dimensions for each interface items. These dimensions are classified as flags to indicate whether the interface item is global, used for control or is a structure or union. Table 5 shows the thesis coupling weight for each possible combination.

<table>
<thead>
<tr>
<th>Parameter (0) / Global (1)</th>
<th>Data (0)/ Control (1)</th>
<th>Non-Structure (0) / Structure (1)</th>
<th>Thesis Weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>3</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>4</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>5</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>6</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>7</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>8</td>
</tr>
</tbody>
</table>

Table 5. The Combination of Myers Three Coupling Factors and their Corresponding Proposed Weights.
Clearly, from the above table, the weights for the three factors can be deduced as follows:

- **Element access scope**: The global access scope weight is 4 while the parameter accesses scope weight is 0.
- **Element access purpose**: The control access purpose weight is 2 while the data access purpose weight is 0.
- **Element type**: The structure type weight is 1 while the non-structure type weight is 0.

In most coupling measures, these items are assigned equal values which are implied to be 1.

The following tables, compare the thesis coupling measure weight among the three factors with the Dhama measure.

<table>
<thead>
<tr>
<th>Element Access Scope</th>
<th>Dhama</th>
<th>Thesis</th>
</tr>
</thead>
<tbody>
<tr>
<td>Global</td>
<td>1</td>
<td>4</td>
</tr>
<tr>
<td>Parameter</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

*Table 6. Element Scope Weight Comparisons between Dhama and the Proposed Method.*

<table>
<thead>
<tr>
<th>Element Access Purpose</th>
<th>Dhama</th>
<th>Thesis</th>
</tr>
</thead>
<tbody>
<tr>
<td>Control</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>Data</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

*Table 7. Element Access Purpose Weight Comparisons between Dhama and the Proposed Method.*

<table>
<thead>
<tr>
<th>Element Type</th>
<th>Dhama</th>
<th>Thesis</th>
</tr>
</thead>
<tbody>
<tr>
<td>Structure</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Non-Structure</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

*Table 8. Element Type Weight Comparisons between Dhama and the Proposed Method.*
In this way, the coupling weight identifies data coupling when the thesis weight is 1. It identifies stamp coupling when the weight value is even. It identifies control coupling when weight value is 3, 4, 7, or 8. It identifies global coupling when the weight is greater than 4.

This method will consolidate factor numbers 2-4 in the comparison table directly. Factor number 5 will be treated when the assignment of the jth element to the ith subprogram is stated. Factor number 1 will be taken care off by the model indirectly.

One problem with Dhama method, it calculates the coupling of a function or procedure. The definition of coupling assumes that coupling is measured between module pairs. In other words, coupling is an inter-modular relationship arising from procedure or function calling, parameter passing or global variable usage’s.

Factors number 6 and 7 may be accounted for by assigning the weight of an element in proportion to the number of times the module has been called where the element is in the interface of that module. Item number 8 can be captured by counting an interface element twice, one in the called procedure or function and the other in the calling procedure or function.

Based on the above discussion, thesis coupling measure is described as follows:

Let there are \( n \) interface elements, i.e., parameter or global or return items from a function, in a system of \( m \) modules
Let $M_i$ be the $i$th module in the system of $m$ modules.

Also, let

$G_{ij}$: a flag stating whether the $j$th interface is global (1) or parameter (0) in the $i$th module.

$C_{ij}$: a flag stating whether the $j$th interface is control (1) or data (0) in the $i$th module.

$S_{ij}$: a flag stating whether the $j$th interface is Structure (1) or Not (0) in the $i$th module.

Let the $i$th module calls the $k$th module then

\[
d_{ij} = d_{ij} + \sum (4 \times G_{ij} + 2 \times C_{ij} + S_{ij} + 1) \&
\]

\[
d_{kj} = d_{kj} + \sum (4 \times G_{kj} + 2 \times C_{kj} + S_{kj} + 1)
\]

Where:

$d_{ij}, d_{kj}$: is the coupling weight of the $j$th interface element in the $i$th module and the $k$th module or the definition matrix entry for the $j$th interface element in the $i$th module and the $k$th module respectively.

In this way, each invocation of a module using an interface element will be counted. Also, each interface item will be counted in the called module and in the calling module. The item access direction weight is assumed to be the same because only Dhama includes it in its measure and he assigned them the same weight. Also, the other measures do not incorporate this factor in them at all. As a result, this method identified four primary factors that affect coupling. The first three were described above. The fourth factor is the number of times the module is called or is being called.
From this definition matrix, a coupling matrix relating each module to each other is calculated automatically as described in measuring the interaction between object model section above.

This assignment has the following properties:

1. The element coupling weight will increase when using global data element versus a parameter. This is because the scope weight of global is higher than the scope weight of parameters.

2. The element coupling weight will be the same regardless of its element access type. This is because this method treats input and output access type equally based on the other methods that assign it equal weight.

3. The element coupling weight will increase when the purpose of the element is control than data. This is because the control weight is higher than the data weight. The element coupling weight will increase when its corresponding type is structure versus non-structure types. This is evident because the weights of the structure types are assigned higher weights than non-structure types.

4. The element coupling weight will increase as the number of interface arguments increases. This is because as the number of interface arguments increases, their cumulative weights increase resulting in an increase in their coupling weight.

In addition, the measure possess the following features:

- It is a source code metric, i.e., it assumes that the source code of the modules exists using any imperative higher level language.
• This method can be applied as a design metric when the design is specified using structure charts method.

• It consolidate the items that affect the coupling between pair of modules in the literature.

• This method assigns the elements higher numbers in association with the order of Myers classification of coupling. However, this assignment does not determine the content type of coupling between a pair of modules due to the argument that the current programming languages do not allow this type of coupling to occur.

• This assignment assumes that the data elements are static, i.e., they are created in the compilation phase. Dynamic data structure measurements depend on the execution of the program and are harder to measure.
Chapter 4

New Measure Assignment Theoretical Validation

This is the fourth stage in Sheppered and Ince methodology. Starting from this chapter, this thesis measure is referred to as the new measure. This chapter shows that the new method assignment is a ratio scale using the method of Zuse [Zuse90]. This implies that ratios of the new measure are meaningful operations. Hence, the model can be utilized to calculate the coupling of procedures based on these ratios. The new assignment is shown to be a ratio scale in two steps. First, ordinal scale axioms are discussed. These axioms are satisfied by the original assignment of the new measure. This shows that the original assignment is an ordinal scale. Second, ratio scale axioms are listed. These axioms are preserved by the new measure full assignment. This shows that the new measure assignment is a ratio scale.

4.1 Ordinal Scale

Let E be the set of all variables pertaining to a system of procedures. Let $\geq$ be a relation on the set E stating whether the one element is contributing more in coupling than the other. Formally $e_1 \geq e_2$
means that \( e_1 \) is contributing greater or equal in coupling of a procedure than \( e_2 \). \( A = (E, \; ^\geq) \) is called an empirical relational system. Another relational system with known mathematical properties is \( B = (R, \; \geq) \) where \( R \) is the set of real numbers and \( \geq \) is the greater than or equal operator. A measure is mapping from the empirical relational system to numerical relational system that preserves the empirical properties. The system \( (A, B, u) \) is called a scale. There are basically five types of scales that differs from each other by how much of the empirical properties are preserved and what kind of statistical operations can be performed and make sense. These scales ordered from lowest to highest, nominal, ordinal, interval, ratio and absolute.

The new measure function denoted by \( u \) ranks the elements in order of the amount of contribution to coupling. Each element \( e \) is assigned a number \( u(e) \) in the range of 0 to 8. This assignment satisfies the following:

- If you know the type of coupling an element holds, you know the measure assigned number.
- If you know the measure assigned number to an element, you know the type of coupling the element holds.

Formally, the new measure function is a homomorphism that satisfies the following conditions:

- If an element \( e_1 \) is contributing more in coupling than another element \( e_2 \), denoted by \( e_1 \; ^\geq \; e_2 \), then the new measure assigned number to the element \( e_1 \) is greater than or equal to the new measure assigned number to the element \( e_2 \), denoted by \( u(e_1) \; \geq \; u(e_2) \).
- If the new measure assigned number to an element \( e_1 \) that is greater than or equal to the new measure assigned number to other element \( e_2 \), denoted by \( u(e_1) \; \geq \; u(e_2) \) then the element \( e_1 \) is contributing more in coupling than the element \( e_2 \), denoted by \( e_1 \; ^\geq \; e_2 \).
Ordinal scale preserves the ranking between the elements. A scale is ordinal if its mapping satisfies transitivity and completeness \([\text{Hend96}, \text{Zuse90}]\).

- **Transitive**: \(e_1 \geq e_2 \text{ and } e_2 \geq e_3 \Rightarrow e_1 \geq e_3 \) for all \(e_1, e_2, e_3 \) belongs to \(E\).

- **Completeness**: either \(e_1 \geq e_2 \) or \(e_2 \geq e_1 \) for all \(e_1, e_2 \) belongs to \(E\).

These two conditions are called weak order. Certainly our mapping function stratifies both conditions:

**Transitive**: we have \(e_1 \geq e_2 \Rightarrow u(e_1) \geq u(e_2) \) and

\[
e_2 \geq e_3 \Rightarrow u(e_2) \geq u(e_3) \text{ and}
\]

\[
u(e_1) \geq u(e_2) \text{ and } u(e_2) \geq u(e_3) \Rightarrow u(e_1) \geq u(e_3) \Rightarrow e_1 \geq e_3.
\]

**Complete**: from normal arithmetic laws we have: either \(u(e_1) \geq u(e_2) \) or \(u(e_2) \geq u(e_1) \) ⇒

\[
either e_1 \geq e_2 \text{ or } e_2 \geq e_1.
\]

Therefore, the new method is of ordinal scale.

### 4.2 Ratio Scale

A ratio scale must have another operation called concatenation \((\circ)\) defining what happens when two elements are concatenated. The concatenation operation may be defined as \(o(e_1, e_2) = \text{Sum up of contribution of both elements when the containing procedure is called with these two elements as parameters or globals.}\) The empirical relational system \(A = (E, \geq, o)\) includes the concatenation
operation. The corresponding numerical relational system \( B = (R, \geq, +) \) includes the sum operator.

Now, the new system \((A, B, u)\) is a ratio scale if the empirical relation system is extensive structure and the mapping \(u\) is additive [Hend96, Zuse90]. The extensive structure must satisfy the following conditions:

- Transitive: \( e_1 \geq e_2 \) and \( e_2 \geq e_3 \) \(\Rightarrow\) \( e_1 \geq e_3 \) for all \( e_1, e_2, e_3 \) belongs to \( E \).
- Completeness: either \( e_1 \geq e_2 \) and \( e_2 \geq e_1 \) all \( e_1, e_2 \) belongs to \( E \)
- Transitivity: \( (e_1 \circ e_2) \circ e_3 = e_1 \circ (e_2 \circ e_3) \)
- Monotonicity: \( e_1 \geq e_2 \) if and only if \( (e_1 \circ e_3) \geq (e_2 \circ e_3) \) if and only if \( (e_3 \circ e_1) \geq (e_3 \circ e_2) \)
- Archimedean: if \( e_1 \Rightarrow e_2 \) then for any \( e_3, e_4 \) belongs to \( E \), there exist a positive integer \( n \) such that: \( n \ e_1 \circ e_3 \geq n \ e_2 \circ e_4 \) where \( n a \) is defined recursively \( 1 a = a, (n + 1) a = n a \circ a \) where \( \Rightarrow \) is the greater than operator in the contribution of coupling of elements.

Clearly, our empirical \( \geq \) operation is transitive and complete, i.e., a weak order. The concatenation operations is transitive because it is a sum:

Transitivity: Let \( e_1, e_2 \) and \( e_3 \) be any elements belonging to \( E \) then

\[
\begin{align*}
    u(e_1) + u(e_2) + u(e_3) &= u(e_1) + u(e_2) + u(e_3) \Rightarrow \\
    [u(e_1) + u(e_2)] + u(e_3) &= u(e_1) + [u(e_2) + u(e_3)] \Rightarrow \\
    (e_1 \circ e_2) + u(e_3) &= u(e_1) + (e_2 \circ e_3) \Rightarrow \\
    (e_1 \circ e_2) \circ e_3 &= e_1 \circ (e_2 \circ e_3).
\end{align*}
\]

The Monotonicity is satisfied because of the definition of \( \geq \) and the concatenation operation as
follows:

Monotonicity: \( e_1 \preceq e_2 \iff (e_1 \circ e_3) \preceq (e_2 \circ e_3) \iff (e_3 \circ e_1) \preceq (e_3 \circ e_2) \)

Let \( e_1 \preceq e_2 \iff u(e_1) \geq u(e_2) \iff u(e_1) + u(e_3) \geq u(e_2) + u(e_3) \iff (e_3 \circ e_1) \preceq (e_3 \circ e_2) \)

\[ \iff u(e_3) + u(e_1) \geq u(e_3) + u(e_2) \iff (e_3 \circ e_1) \preceq (e_3 \circ e_2) \]

Let \( (e_1 \circ e_3) \preceq (e_2 \circ e_3) \iff u(e_1) + u(e_3) \geq u(e_2) + u(e_3) \iff u(e_1) \geq u(e_2) \iff e_1 \preceq e_2 \)

Let \( (e_3 \circ e_1) \preceq (e_3 \circ e_2) \iff u(e_3) + u(e_1) \geq u(e_3) + u(e_2) \iff u(e_1) \geq u(e_2) \iff e_1 \preceq e_2 \)

The fourth condition is satisfied because it is always true for \( n = \text{maximum number of the absolute difference of } e_3 \) and \( e_4 \) as follows:

Archimedean: if \( e_1 \rightarrow e_2 \) then for all \( e_3, e_4 \) belongs to \( E \), there exist a positive integer \( n \) such that: \( n e_1 \circ e_3 \preceq n e_2 \circ e_4 \) where \( n a \) is defined recursively \( 1 \circ a = a, (n + 1) \circ a = n \circ a \circ a \)

Let \( e_1 \rightarrow e_2 \Rightarrow u(e_1) > u(e_2) \Rightarrow u(e_1) \geq u(e_2) + c \) where \( c = u(e_1) - u(e_2) \)

and

Let \( e_3, e_4 \) be any arbitrary elements then

either \( e_3 \preceq e_4 \) or \( e_4 \preceq e_3 \)

Take \( m = \text{Absolute } (u(e_3) - u(e_4)) \)

when \( e_3 \preceq e_4 : u(e_1) \geq u(e_2) + c \Rightarrow m \circ u(e_1) \geq m \circ u(e_2) + m \circ c \Rightarrow m \circ u(e_1) \geq m \circ u(e_2) \)

\[ \Rightarrow m \circ u(e_1) + u(e_3) \geq m \circ u(e_2) + u(e_4) \]

\[ \Rightarrow m \circ e_1 \circ e_3 \preceq m \circ e_2 \circ e_4 \]

when \( e_4 \preceq e_3 : u(e_4) \geq u(e_3) + m \Rightarrow u(e_4) \geq u(e_3) - m \)

\[ \Rightarrow u(e_1) \geq u(e_2) + c \Rightarrow m \circ u(e_1) \geq m \circ u(e_2) + c \circ m \]
\[ m \, u(e_1) + u(e_3) \geq m \, u(e_2) + [u(e_4) - m] + c \, m \]

\[ \Rightarrow m \, u(e_1) + u(e_3) \geq m \, u(e_2) + u(e_4) \]

\[ \Rightarrow m \, e_1 \circ e_3 \preceq m \, e_2 \circ e_4 \]

Therefore, the empirical relational system is an extensive structure.

In addition, the extensive structure is called positive if it satisfies the condition:

\[(e_1 \circ e_2) \rightarrow e_1.\] Our extensive structure is not positive because \(e_1\) and \(e_2\) may have no contribution in the coupling of a procedure.

The additivity of the mapping is

\[ u(e_1 \circ e_2) = u(e_1) + u(e_2) \]

This condition is also satisfied from the definition of concatenation operation. Hence, our measure is a ratio scale.
Chapter 5

New Measure Experimental Validation

This is the fifth stage in Sheppered and Ince methodology. The objective of this chapter is to test if the developed measure conforms to system analysts observed relations on sample modules. Moreover, it tests if the new measure is sensitive to common basic operations on sample modules. In addition, the new measure is compared with the four recent measures on the above two tests. The evaluation is conducted on six modules from five high level languages. The first two were selected from Dhama method [Dham95]. The source language of these two implementations is Ada. He used these modules to show that his measure differentiates between two implementations of the same problem. The third and fifth modules were selected from structural design teaching books on Fortran [Law83] and Pascal [Grah82] respectively. These two modules are considered well designed. The forth and the sixth modules were selected from an oil company to show typical industrial systems. The source language of the forth module is PLI while the sixth module is written in C. The evaluation is achieved in three tasks. First, five coupling measures are applied on a sample module. This shows the developer how to apply the five coupling measures manually to get the module procedures coupling and system coupling.
The five coupling measures are applied in order:

1. New Measure
2. Dhama
3. Fenton and Melton
4. Offut, Harrold and Kotle
5. Chen and Lu

Second, the results of applying the five measures on each module are listed and compared on capturing the observed relations. Finally, the outcomes of applying the five measures when six atomic modifications are conducted on the corresponding modules are stated.

5.1 Coupling Methods Application

The procedure for getting the description matrix from the source code is listed on figure 1. Now, we will apply the new method and the most recent methods on module V because it captures most of the attributes being discussed on figure 1. Module V source code is shown on figure 6 on appendix A. Similarly, the values of the methods on different examples are identified.
Procedure generate the description matrix (system source code) is

1 - find all subprograms
2 - find all global variables used by each subprogram
3 - find the calling scheme of the system with the actual parameter listed
4 - match the actual parameter with the arguments
5 - find the used variables in the system
   5.1 - list the unique global variables used by procedures
   5.2 - for each argument to actual matched pair
      5.2.1 - if actual parameter is a number or expression then list argument
      5.2.2 - else it is a variable or part of structure list it
      5.2.3 - else it is a function then
         5.2.3.1 - if function value is returned by a function name, list it
         5.2.3.2 - else it is returned by argument then list it
6 - Make the description matrix listing the subprograms found in 1 as rows and the variables in step 5 as columns.
7 - Go through the matching in step 5 and mark the non relevant variables for each procedure i
8 - for each relevant name j, for each procedure i, set the three flags Gij Cij Sij initialized to zero's from source code of the containing procedure
   8.1 - Set Global flag (Gij) when variable is global
   8.2 - Set Control flag (Cij) when variable is used in control or selection statements
   8.3 - Set Structure flag (Sij) when variable is a structure or derived from structure
9 - Calculate the coupling weight from the flags as follows:
   9.1 - Get the decimal value of the binary number (Gij Cij Sij) Plus One
10 - For the called sequence, for each calling i , called j pair (i,k) do
   10.1 - for the calling subprogram increase the weight in the position of the calling subprogram and variable name by the calculated weight assuming a value of zero on the start [dij = dij + Decimal (Gij Cij Sij) + 1].
   10.2 - for the called subprogram increase the weight in the position of the calling subprogram and variable name by the calculated weight assuming a value of zero on the start [dkj = dkj + Decimal (Gkj Ckj Skj) + 1].
11 - Calculate the resulting numbers and Generate the description matrix

Figure 1. Calculating Definition Matrix from Source Code Procedure
5.1.1 Sample Application of Coupling Methods

Now, we will apply the new method and the most recent methods on module V. The source code is listed on figure 6.

5.1.1.1 New Method

1. The subprograms are: CRAPS, RANDOM - [F], ROLLDICE, INITIALIZE, ACCEPTBET, PLAY and UPDATE, where [F] means that the preceding subprogram is a function.

2. The global variable used by each program is as follows:
   CRAPS : none.
   RANDOM : SEED
   ROLLDICE : none.
   INITIALIZE : none.
   ACCEPTBET : none.
   PLAY : none.
   UPDATE : none.

3. The system calling scheme with actual parameters is as follows:
   CRAPS
   INITIALIZE(SEED, AMOUNT)
   ACCEPTBET(BET, AMOUNT)
   PLAY(WON)
   ROLLDICE(VALUE)
   RANDOM(6)
   ROLLDICE(VALUE)
   RANDOM(6)
   UPDATE(BET, AMOUNT, WON, FINISHED)

4. The actual parameter matched with the argument per each (calling, called) procedures pair is as follows:
   (CRAPS, INITIALIZE) : SEED <-> SEED, AMOUNT <-> AMOUNT
   (CRAPS, ACCEPTBET) : BET <-> BET, AMOUNT <-> AMOUNT
   (CRAPS, PLAY) : WON <-> WON
   (CRAPS, UPDATE) : BET <-> BET, AMOUNT <-> AMOUNT,
                     WON <-> WON, FINISHED <-> FINISHED
   (PLAY, ROLLDICE) : VALUE <-> VALUE
   (ROLLDICE, RANDOM) : 6 <-> N
5. The used variables used by the system is as follows:
   5.1 The only global variable used is: SEED
   5.2
   5.2.1 The only number passed is to random subprogram included as: N
   5.2.2 The actual parameters variables passed are: SEED, AMOUNT, BET, WON,
       FINISHED and VALUE
   5.2.3 The only function that passed its value through its name is RANDOM

6. The description matrix template resulted with the non-relevance variables marked with cross for the corresponding procedure and the three flags set is shown in Table 9

7. Going through each call, I get the decimal value of the weight of each variable, put the value in the box and the value for the calling procedure against the variable in its appropriate places separated by commas as listed in Table 10

8. Sum up the comma separated values and generate the definition matrix as listed in Table 11

<table>
<thead>
<tr>
<th>Variable</th>
<th>SEED</th>
<th>AMOUNT</th>
<th>BET</th>
<th>WON</th>
<th>FINISHED</th>
<th>VALUE</th>
<th>RANDOM</th>
<th>N</th>
</tr>
</thead>
<tbody>
<tr>
<td>CRAPS</td>
<td>000</td>
<td>000</td>
<td>000</td>
<td>000</td>
<td>010</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>INITIALIZE</td>
<td>000</td>
<td>010</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>ACCEPTBET</td>
<td>x</td>
<td>010</td>
<td>010</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>PLAY</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>010</td>
<td>x</td>
<td>010</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>ROLLDICE</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>000</td>
<td>000</td>
<td>000</td>
<td>000</td>
</tr>
<tr>
<td>RANDOM</td>
<td>100</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>000</td>
<td>000</td>
<td>000</td>
</tr>
<tr>
<td>UPDATE</td>
<td>x</td>
<td>010</td>
<td>000</td>
<td>010</td>
<td>010</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
</tbody>
</table>

Table 9. The Description Matrix with Weights Flags Assigned from Source Code with Non-relevance Variables Marked with ‘x’.
<table>
<thead>
<tr>
<th>SEED</th>
<th>AMOUNT</th>
<th>BET</th>
<th>WON</th>
<th>FINISHED</th>
<th>VALUE</th>
<th>RANDOM</th>
<th>N</th>
</tr>
</thead>
<tbody>
<tr>
<td>CRAPS</td>
<td>1,1,1,1</td>
<td>1,1</td>
<td>1,1</td>
<td>1,1</td>
<td>3</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>INITIALIZE</td>
<td>1</td>
<td>3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>ACCEPTBET</td>
<td>0</td>
<td>3</td>
<td>3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>PLAY</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>3</td>
<td>0</td>
<td>3,3</td>
<td>0</td>
</tr>
<tr>
<td>ROLLDICE</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1,1</td>
<td>1,1,1</td>
<td>1,1,1</td>
</tr>
<tr>
<td>RANDOM</td>
<td>5,5,5</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1,1,1</td>
<td>1,1,1</td>
</tr>
<tr>
<td>UPDATE</td>
<td>0</td>
<td>3</td>
<td>1</td>
<td>3</td>
<td>3</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Table 10. The Description Matrix with Coupling Weights Assigned Separated by Commas.

<table>
<thead>
<tr>
<th>SEED</th>
<th>AMOUNT</th>
<th>BET</th>
<th>WON</th>
<th>FINISHED</th>
<th>VALUE</th>
<th>RANDOM</th>
<th>N</th>
</tr>
</thead>
<tbody>
<tr>
<td>CRAPS</td>
<td>5</td>
<td>3</td>
<td>2</td>
<td>2</td>
<td>3</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>INITIALIZE</td>
<td>1</td>
<td>3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>ACCEPTBET</td>
<td>0</td>
<td>3</td>
<td>3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>PLAY</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>3</td>
<td>0</td>
<td>6</td>
<td>0</td>
</tr>
<tr>
<td>ROLLDICE</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>RANDOM</td>
<td>20</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>UPDATE</td>
<td>0</td>
<td>3</td>
<td>1</td>
<td>3</td>
<td>3</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Table 11. The Final Generated Description Matrix.

### 5.1.1.2 Dhama Method

Module coupling = \( \frac{1}{m} \) where \( m \) is:

\[
m = i_1 + 2i_2 + u_1 + 2u_2 + g_1 + 2g_2 + w + r
\]

where:
\( i_1 \): Number of input data parameters

\( i_2 \): Number of input control parameters

\( u_1 \): Number of output data parameters

\( u_2 \): Number of output control parameters

\( g_1 \): Number of global variables used as data

\( g_2 \): Number of global variables used as control

\( w \): Number of modules called

\( r \): Number of module calling the module under consideration.

Applying Dhama Method on Module V yields the followings:

**CRAPS:**

Parameter Control: 1 (FINISHED)
Parameter Data : 4 (SEED, AMOUNT, BET, WON)
Module Called : 5 (INITIALIZE, ROLLDICE, ACCEPTBET, PLAY, UPDATE)
Module calling : 1 (Operating System)

**INITIALIZE:**

Parameter Control: 1 (AMOUNT)
Parameter Data : 1 (SEED)
Module Called : 0 (none)
Module calling : 1 (CRAPS)

**ACCEPTBET:**

Parameter Control: 2 (AMOUNT, BET)
Module Called : 0 (none)
Module calling : 1 (CRAPS)

**PLAY:**

Parameter Control: 1 (WON)
Module Called : 2 (ROLLDICE, ROLLDICE)
Module calling : 1 (CRAPS)

**ROLLDICE:**

Parameter Data: 1 (VALUE)
Module Called : 2 (RANDOM, RANDOM)
Module calling : 1 (CRAPS)
RANDOM:
Global Input : 1 (SEED)
Parameter Data : 1 (N)
Module Called : 0 (none)
Module calling: 2 (ROLLDICE, ROLLDICE)

UPDATE:
Parameter Data : 1 (BET)
Parameter Control : 3 (AMOUNT, WON, FINISHED)
Module Called : 0 (none)
Module calling: 1 (CRAPS)

The above calculation can be summarized in Table 12.

<table>
<thead>
<tr>
<th></th>
<th>i₁</th>
<th>2i₂</th>
<th>u₁</th>
<th>2u₂</th>
<th>2g₁</th>
<th>2g₂</th>
<th>w</th>
<th>r</th>
<th>m</th>
<th>l/m</th>
</tr>
</thead>
<tbody>
<tr>
<td>CRAPS</td>
<td>4</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>5</td>
<td>1</td>
<td>12</td>
<td>0.0833</td>
</tr>
<tr>
<td>INITIALIZE</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>4</td>
<td>0.2500</td>
</tr>
<tr>
<td>ACCEPTBET</td>
<td>0</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>5</td>
<td>0.2000</td>
</tr>
<tr>
<td>PLAY</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>1</td>
<td>5</td>
<td>0.2000</td>
</tr>
<tr>
<td>ROLLDICE</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>1</td>
<td>4</td>
<td>0.2500</td>
</tr>
<tr>
<td>RANDOM</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>5</td>
<td>0.2000</td>
</tr>
<tr>
<td>UPDATE</td>
<td>1</td>
<td>3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>8</td>
<td>0.1250</td>
</tr>
<tr>
<td>System</td>
<td>8</td>
<td>5</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>9</td>
<td>8</td>
<td>8</td>
<td>0.1869</td>
</tr>
</tbody>
</table>

Table 12. Dhama Method Weights Calculation Summary

From the above values assigned to each procedure, Dhama new to calculate the coupling of procedure
to procedure by taking their average values are shown in Table 13.

<table>
<thead>
<tr>
<th></th>
<th>CRAPS</th>
<th>INITIALIZE</th>
<th>ACCEPTBET</th>
<th>PLAY</th>
<th>ROLLDICE</th>
<th>RANDOM</th>
<th>UPDATE</th>
</tr>
</thead>
<tbody>
<tr>
<td>CRAPS</td>
<td>0.0833</td>
<td>0.1667</td>
<td>0.1417</td>
<td>0.1417</td>
<td>0.1667</td>
<td>0.1417</td>
<td>0.1042</td>
</tr>
<tr>
<td>INITIALIZE</td>
<td>0.1667</td>
<td>0.2500</td>
<td>0.2250</td>
<td>0.2250</td>
<td>0.2500</td>
<td>0.2250</td>
<td>0.1875</td>
</tr>
<tr>
<td>ACCEPTBET</td>
<td>0.1417</td>
<td>0.2250</td>
<td>0.2000</td>
<td>0.2000</td>
<td>0.2250</td>
<td>0.2000</td>
<td>0.1625</td>
</tr>
<tr>
<td>PLAY</td>
<td>0.1417</td>
<td>0.2250</td>
<td>0.2000</td>
<td>0.2000</td>
<td>0.2250</td>
<td>0.2000</td>
<td>0.1625</td>
</tr>
<tr>
<td>ROLLDICE</td>
<td>0.1667</td>
<td>0.2500</td>
<td>0.2250</td>
<td>0.2250</td>
<td>0.2500</td>
<td>0.2250</td>
<td>0.1875</td>
</tr>
<tr>
<td>RANDOM</td>
<td>0.1417</td>
<td>0.2250</td>
<td>0.2000</td>
<td>0.2000</td>
<td>0.2250</td>
<td>0.2000</td>
<td>0.1625</td>
</tr>
<tr>
<td>UPDATE</td>
<td>0.1042</td>
<td>0.1875</td>
<td>0.1625</td>
<td>0.1625</td>
<td>0.1875</td>
<td>0.1625</td>
<td>0.1250</td>
</tr>
</tbody>
</table>

Table 13. Procedure by Procedure Coupling Values Using Dhama Method.
5.1.1.3 Fenton and Melton Method

The coupling of modules \( x, y \) denoted \( C(x, y) \) is calculated by

\[
C(x, y) = i + \frac{n}{n+1}
\]

where:

- \( i \) the level of the greatest coupling type (defined in reprintable 5-6) and
- \( n \) is the number of interconnections between \( x \) and \( y \)

<table>
<thead>
<tr>
<th>Coupling Type</th>
<th>Assigned Number</th>
<th>Modified Definition Between Modules ( x ) and ( y )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Content</td>
<td>5</td>
<td>Module ( x ) refers to the inside of module ( y ), i.e., it branches into, changes data, or alters a statement in ( y ).</td>
</tr>
<tr>
<td>Common</td>
<td>4</td>
<td>Modules ( x ) and ( y ) refer to the same global data.</td>
</tr>
<tr>
<td>Control</td>
<td>3</td>
<td>Module ( x ) passes a controlling parameter to ( y ), i.e., the parameter is a flag.</td>
</tr>
<tr>
<td>Stamp</td>
<td>2</td>
<td>Modules ( x ) and ( y ) accept the same record type as a parameter.</td>
</tr>
<tr>
<td>Data</td>
<td>1</td>
<td>Modules ( x ) and ( y ) communicate by parameters, each of which is either a single data or homogeneous structure that does not incorporate control element.</td>
</tr>
<tr>
<td>No Coupling</td>
<td>0</td>
<td>Modules ( x ) and ( y ) have no communication, i.e., are totally independent.</td>
</tr>
</tbody>
</table>

*Table 14. Fenton and Melton Modified Definitions for Myers Coupling Levels.*

Applying Fenton and Melton Method on Module V yields for each (called, calling) procedure pair:

**(CRAPS, INITIALIZE):**
- Parameter Control : AMOUNT
- Parameter data : SEED
- Number of Interconnections : 2 (SEED, AMOUNT)
- \( C(X, Y) = 3 + 1 \div (2+1) = 3.3333 \)

**(CRAPS, ACCEPTBET):**
- Parameter Control : AMOUNT, BET
- Number of Interconnections : 2 (AMOUNT, BET)
- \( C(X, Y) = 3 + 1 \div (2+1) = 3.3333 \)

**(CRAPS, PLAY):**
- Parameter Control : WON
- Number of Interconnections : 1 (WON)
- \( C(X, Y) = 3 + 1 \div (1+1) = 3.5000 \)
(CRAPS, UPDATE):
Parameter Control : FINISHED
Parameter data : AMOUNT, BET, WON
Number of Interconnections : 4 (WON)
\[ C(X, Y) = 3 + 1 / (4+1) = 3.2000 \]

(CRAPS, RANDOM):
Global : SEED
Number of Interconnections : 1 (WON)
\[ C(X, Y) = 4 + 1 / (1+1) = 4.5000 \]

(PLAY, ROLLDICE):
Parameter Data : VALUE
Number of Interconnections : 1 (VALUE)
\[ C(X, Y) = 1 + 1 / (1+1) = 1.5000 \]

(ROLLDICE, RANDOM):
Parameter Data : 6
Number of Interconnections : 1 (VALUE)
\[ C(X, Y) = 1 + 1 / (1+1) = 1.5000 \]

They new to use the median of all pairwise module couplings to measure the system global coupling.

(System):
\[ C \text{ (System)} = \text{Median (1.5, 1.5, 3.3, 3.3, 3.5, 4.3)} = 3.3333 \]

The above coupling calculation between procedure pairs is summarized in the following table:

<table>
<thead>
<tr>
<th>Function Pair Coupled</th>
<th>CRAPS, INITIALIZE</th>
<th>CRAPS, ACCEPT BET</th>
<th>CRAPS, PLAY</th>
<th>CRAPS, UP DATE</th>
<th>CRAPS, RANDOM</th>
<th>PLAY, ROLLDICE</th>
<th>ROLLDICE, RANDOM</th>
<th>System</th>
</tr>
</thead>
</table>

*Table 15. Procedure Pair Coupling Weights Using Fenton and Melton Method.*

### 5.1.1.4 Offut, Harrold and Kotele Method

The coupling of modules \( x, y \) denoted \( C(x, y) \) is calculated by

\[ C(x, y) = i + (n / (n + 1)) - 0.5 \]

where:
i the level of the greatest coupling type (defined in the reprinted Table 16) and

n is the number of interconnections between x and y

<table>
<thead>
<tr>
<th>Coupling Type</th>
<th>Assigned Number</th>
<th>Modified Definition Between Modules x and y</th>
</tr>
</thead>
<tbody>
<tr>
<td>Content</td>
<td>6</td>
<td>Modules x and y access and change each others internal data or procedural states.</td>
</tr>
<tr>
<td>Common</td>
<td>5</td>
<td>Modules x and y refer to the same global data.</td>
</tr>
<tr>
<td>External</td>
<td>4</td>
<td>Modules x and y communicate through an external medium such as a file.</td>
</tr>
<tr>
<td>Control</td>
<td>3</td>
<td>Module x passes a flag value that is used to control the internal logic of module y.</td>
</tr>
<tr>
<td>Stamp</td>
<td>2</td>
<td>Modules x and y communicate through a parameter which is a record.</td>
</tr>
<tr>
<td>Data</td>
<td>1</td>
<td>Modules x and y pass data through scalar or array parameters.</td>
</tr>
<tr>
<td>Independent</td>
<td>0</td>
<td>Modules x and y have no coupling between them.</td>
</tr>
</tbody>
</table>

*Table 16. Offut, Harrold and Kotle Modified Definitions for Myers Coupling Levels.*

Applying Offut, Harrold and Kotle Method on Module V yields for each (called, calling) procedure pair:

(CRAPS, INITIALIZE):
Parameter Control : AMOUNT
Parameter data : SEED
Number of Interconnections : 2 (SEED, AMOUNT)
\[ C(X, Y) = 3 + 1 / (2+1) - 0.5 = 2.8333 \]

(CRAPS, ACCEPTBET):
Parameter Control : AMOUNT, BET
Number of Interconnections : 2 (AMOUNT, BET)
\[ C(X, Y) = 3 + 1 / (2+1) - 0.5 = 2.8333 \]

(CRAPS, PLAY):
Parameter Control : WON
Number of Interconnections : 1 (WON)
\[ C(X, Y) = 3 + 1 / (1+1) - 0.5 = 3.0000 \]

(CRAPS, UPDATE):
Parameter Control : FINISHED
Parameter data : AMOUNT, BET, WON
Number of Interconnections : 4 (WON)
\[ C(X, Y) = 3 + 1 / (4+1)- 0.5 = 2.2000 \]
(CRAPS, RANDOM):
Global : SEED
Number of Interconnections : 1 (WON)
C(X, Y) = 5 + 1 / (1+1) - 0.5 = 5.0000

(PLAY, ROLLDICE):
Parameter Data : VALUE
Number of Interconnections : 1 (VALUE)
C(X, Y) = 1 + 1 / (1+1) - 0.5 = 1.000

(ROLLDICE, RANDOM):
Parameter Data : 6
Number of Interconnections : 1 (VALUE)
C(X, Y) = 1 + 1 / (1+1) - 0.5 = 1.0000

Applying Offut, Harrold and Kotle Method on Module V yield for each (called, calling) procedure pair:

(INITIALIZE, CRAPS):
Parameter data : SEED, AMOUNT
Number of Interconnections : 2 (SEED, AMOUNT)
C(X, Y) = 1 + 1 / (2+1) - 0.5 = 0.8333

(ACEPTBET,CRAPS):
Parameter Data : AMOUNT, BET
Number of Interconnections : 2 (AMOUNT, BET)
C(X, Y) = 1 + 1 / (2+1) - 0.5 = 2.8333

(PLAY, CRAPS):
Parameter Data : WON
Number of Interconnections : 1 (WON)
C(X, Y) = 1 + 1 / (1+1) - 0.5 = 1.0000

(UPDATE, CRAPS):
Parameter Control : FINISHED, AMOUNT, WON
Parameter data : BET
Number of Interconnections : 4 (WON)
C(X, Y) = 3 + 1 / (4+1) - 0.5 = 2.2000

(RANDOM,CRAPS):
Global : SEED
Number of Interconnections : 1 (WON)
C(X, Y) = 5 + 1 / (1+1) - 0.5 = 5.0000

(ROLLDICE, PLAY):
Parameter Control : VALUE
Number of Interconnections : 1 (VALUE)
\[ C(X, Y) = 3 + 1 - \frac{1}{(1+1)} - 0.5 = 3.000 \]

(RANDOM, ROLLDICE):
Parameter Data : N
Number of Interconnections : 1 (VALUE)
\[ C(X, Y) = 1 + 1 - \frac{1}{(1+1)} - 0.5 = 1.0000 \]

The overall measure of the system is sum of all pairs over number of pairs. Table 17 summarizes the coupling weights.

<table>
<thead>
<tr>
<th>Function Pair Coupled</th>
<th>CRAPS, INITIALIZE</th>
<th>CRAPS, ACCEPTBET</th>
<th>CRAPS, PLAY</th>
<th>CRAPS, UPDATE</th>
<th>CRAPS, RANDOM</th>
<th>PLAY, ROLLDICE</th>
<th>ROLLDICE, RANDOM</th>
<th>System</th>
</tr>
</thead>
<tbody>
<tr>
<td>Calling to Called Coupling</td>
<td>2.8333</td>
<td>2.8333</td>
<td>3.0000</td>
<td>2.2000</td>
<td>5.0000</td>
<td>1.0000</td>
<td>1.0000</td>
<td>2.4095</td>
</tr>
<tr>
<td>Called to Calling Coupling</td>
<td>0.83333</td>
<td>2.8333</td>
<td>1.0000</td>
<td>2.2000</td>
<td>5.0000</td>
<td>3.0000</td>
<td>1.0000</td>
<td></td>
</tr>
</tbody>
</table>

*Table 17. Procedure Pair Coupling Weights Using Offut, Harrold and Kotle Method.*

### 5.1.1.5 Chen and Lu Method

Chen and Lu coupling metric is defined as the summary of the followings:

1. The number of procedures called by this procedure (Non - repeating).
2. The number of procedures calling this procedures (Non - repeating).
3. The number of co-operated procedures (Across all levels from procedure level ).

Applying this method on Module V for each procedure yields:

**CRAPS:**
Number of procedure called : 4 (INITIALIZE, ACCEPTBET, PLAY, UPDATE)
Number of procedure calling : 1 (Operating System)
Number of co-operated procedures : 6 (INITIALIZE, ACCEPTBET, PLAY, ROLLDICE, UPDATE, RANDOM)
Coupling = 4 + 1 + 6 = 11

INITIALIZE:
Number of procedure called : 0 (none)
Number of procedure calling : 1 (CRAPS)
Number of co-operated procedures : 0 (none)
Coupling = 0 + 1 + 0 = 2

ACCEPTBAT:
Number of procedure called : 0 (none)
Number of procedure calling : 1 (CRAPS)
Number of co-operated procedures : 0 (none)
Coupling = 0 + 1 + 0 = 1

PLAY:
Number of procedure called : 1 (ROLDICE)
Number of procedure calling : 1 (CRAPS)
Number of co-operated procedures : 2 (ROLDICE, RANDOM)
Coupling = 1 + 1 + 2 = 4

ROLDICE:
Number of procedure called : 1 (RANDOM)
Number of procedure calling : 1 (PLAY)
Number of co-operated procedures : 1 (RANDOM)
Coupling = 1 + 1 + 1 = 3

RANDOM:
Number of procedure called : 0 (none)
Number of procedure calling : 1 (ROLDICE)
Number of co-operated procedures : 0 (none)
Coupling = 0 + 1 + 0 = 1

UPDATE:
Number of procedure called : 0 (none)
Number of procedure calling : 1 (CRAPS)
Number of co-operated procedures : 0 (none)
Coupling = 0 + 1 + 0 = 1

The system coupling values can be computed by taking the average of all procedure values as in the following table. We do this for the sake of comparisons.

<table>
<thead>
<tr>
<th>Function Name</th>
<th>CRAPS</th>
<th>INITIALIZE</th>
<th>ACCEPTBET</th>
<th>PLAY</th>
<th>ROLLDICE</th>
<th>RANDOM</th>
<th>UPDATE</th>
<th>System</th>
</tr>
</thead>
<tbody>
<tr>
<td>Coupling</td>
<td>11</td>
<td>1</td>
<td>1</td>
<td>4</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>3.1428</td>
</tr>
</tbody>
</table>

Table 18. Procedure Coupling Weights Using Chen and Lu Method.
The procedure to procedure coupling values can be computed by taking the average of two values as illustrated in the following table. This is done for the sake of comparisons.

<table>
<thead>
<tr>
<th></th>
<th>CRAPS</th>
<th>INITIALIZE</th>
<th>ACCEPTBET</th>
<th>PLAY</th>
<th>ROLLDICE</th>
<th>RANDOM</th>
<th>UPDATE</th>
</tr>
</thead>
<tbody>
<tr>
<td>CRAPS</td>
<td>11.0000</td>
<td>6.0000</td>
<td>6.0000</td>
<td>7.5000</td>
<td>7.0000</td>
<td>6.0000</td>
<td>6.0000</td>
</tr>
<tr>
<td>INITIALIZE</td>
<td>6.0000</td>
<td>1.0000</td>
<td>1.0000</td>
<td>2.5000</td>
<td>2.0000</td>
<td>1.0000</td>
<td>1.0000</td>
</tr>
<tr>
<td>ACCEPTBET</td>
<td>6.0000</td>
<td>1.0000</td>
<td>1.0000</td>
<td>2.5000</td>
<td>2.0000</td>
<td>1.0000</td>
<td>1.0000</td>
</tr>
<tr>
<td>PLAY</td>
<td>7.5000</td>
<td>2.5000</td>
<td>2.5000</td>
<td>4.0000</td>
<td>3.5000</td>
<td>2.5000</td>
<td>2.5000</td>
</tr>
<tr>
<td>ROLLDICE</td>
<td>7.0000</td>
<td>2.0000</td>
<td>2.0000</td>
<td>3.5000</td>
<td>3.0000</td>
<td>2.0000</td>
<td>2.0000</td>
</tr>
<tr>
<td>RANDOM</td>
<td>6.0000</td>
<td>1.0000</td>
<td>1.0000</td>
<td>2.5000</td>
<td>2.0000</td>
<td>1.0000</td>
<td>1.0000</td>
</tr>
<tr>
<td>UPDATE</td>
<td>6.0000</td>
<td>1.0000</td>
<td>1.0000</td>
<td>2.5000</td>
<td>2.0000</td>
<td>1.0000</td>
<td>1.0000</td>
</tr>
</tbody>
</table>

Table 19. Procedure by Procedure Coupling Values Using Chen and Lu Method.

5.2 Empirical Validation

Empirical validation corresponds to applying the five methods on each module. Each module begins with concise information about its function. After that, the five measures are applied and the results are shown. Then, the results are analyzed and compared on preserving the observed relations stated in the beginning of the analysis. Finally, the comparison results are summarized for all modules on the five measures.

5.2.1 Module I

Module I is an Ada program that implements the selection sort by a single procedure [Dham95]. The source code of this procedure is listed on figure 2. Since this model measures the interaction between objects, there must be at least two subprograms in the system. For the sake of comparisons, a main
program was added to the Dharma two implementations with the same parameters as their arguments. Applying the new method yields the following definition matrix. Feeding this definition matrix into the model generates the coupling matrix in unity forms which gives the coupling proportion for each procedure in the row to each other. The value for the same procedure in the row and the column signifies how is this procedure separated from the others.

### 5.2.1.1 New Method

Applying the new metric generates the following results:

The definition matrix:

<table>
<thead>
<tr>
<th></th>
<th>n</th>
<th>to_be_sorted</th>
<th>a_or_d</th>
</tr>
</thead>
<tbody>
<tr>
<td>main</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Sort1</td>
<td>3</td>
<td>3</td>
<td>3</td>
</tr>
</tbody>
</table>

*Table 20. Module I Definition Matrix.*

The coupling matrix is

<table>
<thead>
<tr>
<th>Function Name</th>
<th>main</th>
<th>sort1</th>
</tr>
</thead>
<tbody>
<tr>
<td>main</td>
<td>0.2500</td>
<td>0.7500</td>
</tr>
<tr>
<td>sort1</td>
<td>0.2500</td>
<td>0.7500</td>
</tr>
</tbody>
</table>

*Table 21. Module I Coupling Matrix.*

The normalized coupling matrix:

<table>
<thead>
<tr>
<th>Function Name</th>
<th>main</th>
<th>sort1</th>
</tr>
</thead>
<tbody>
<tr>
<td>main</td>
<td>1.0000</td>
<td>3.0000</td>
</tr>
<tr>
<td>sort1</td>
<td>0.3333</td>
<td>1.0000</td>
</tr>
</tbody>
</table>

*Table 22. Module I Normalized Coupling Matrix.*

The cumulative coupling of functions is:
<table>
<thead>
<tr>
<th>Coupling</th>
<th>main</th>
<th>sort1</th>
<th>System</th>
</tr>
</thead>
<tbody>
<tr>
<td>main</td>
<td>1.66667</td>
<td>0.60000</td>
<td>0.60000</td>
</tr>
<tr>
<td>Coupling Inverse</td>
<td>1.66667</td>
<td>0.60000</td>
<td>0.60000</td>
</tr>
</tbody>
</table>

*Table 23. Module I Procedures Overall Coupling.*

### 5.2.1.2 Dhama Method

However, applying Dhama method on the module I yields the following coupling matrices:

<table>
<thead>
<tr>
<th>Coupling</th>
<th>main</th>
<th>sort1</th>
<th>System</th>
</tr>
</thead>
<tbody>
<tr>
<td>main</td>
<td>5</td>
<td>10</td>
<td>7.5</td>
</tr>
<tr>
<td>Coupling Inverse</td>
<td>0.2000</td>
<td>0.1000</td>
<td>0.150</td>
</tr>
</tbody>
</table>

*Table 24. Module I Procedures Coupling Weights Using Dhama Method.*

The coupling of procedure to procedure is shown in the following table:

<table>
<thead>
<tr>
<th>Function Name</th>
<th>main</th>
<th>sort1</th>
</tr>
</thead>
<tbody>
<tr>
<td>main</td>
<td>0.2000</td>
<td>0.1500</td>
</tr>
<tr>
<td>sort1</td>
<td>0.1500</td>
<td>0.1000</td>
</tr>
</tbody>
</table>

*Table 25. Module I Procedure to Procedure Coupling Weight Using Dhama Method.*

### 5.2.1.3 Fenton and Melton Method

On the other hand, applying Fenton and Melton method on module I yields the following results:

<table>
<thead>
<tr>
<th>Function Pair Coupled</th>
<th>main, sort2</th>
<th>system</th>
</tr>
</thead>
<tbody>
<tr>
<td>Coupling</td>
<td>2.2500</td>
<td>2.2500</td>
</tr>
</tbody>
</table>

*Table 26. Module I Procedure to Procedure Coupling Weight Using Fenton and Melton Method.*
5.2.1.4 Offut, Harrold and Kotel Method

On comparison, applying Offut, Harrold and Kotel method on module I yields the following coupling matrix:

<table>
<thead>
<tr>
<th>Function Pair Coupled</th>
<th>main, sort2</th>
<th>system</th>
</tr>
</thead>
<tbody>
<tr>
<td>Calling to Called Coupling</td>
<td>1.7500</td>
<td>1.2500</td>
</tr>
<tr>
<td>Called to Calling Coupling</td>
<td>0.7500</td>
<td></td>
</tr>
</tbody>
</table>

Table 27. Module I Procedure to Procedure Coupling Weight Using Offut, Harrold and Kotel Method.

5.2.1.5 Chen and Lu Method

Finally, applying Chen and Lu method on module I yields the following coupling matrix:

<table>
<thead>
<tr>
<th>Function Name</th>
<th>main</th>
<th>sort1</th>
<th>System</th>
</tr>
</thead>
<tbody>
<tr>
<td>Coupling</td>
<td>3</td>
<td>1</td>
<td>2</td>
</tr>
</tbody>
</table>

Table 28. Module I Procedure to Procedure Coupling Weight Using Chen and Lu Method.

The procedure to procedure coupling values are shown in Table 29.

<table>
<thead>
<tr>
<th>Function Name</th>
<th>main</th>
<th>sort1</th>
</tr>
</thead>
<tbody>
<tr>
<td>main</td>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>sort1</td>
<td>2</td>
<td>1</td>
</tr>
</tbody>
</table>

Table 29. Module I Procedure to Procedure Coupling Weight Using Chen and Lu Method.

5.2.1.6 Analysis

The program source code shows that main procedure relies heavily on sort1 procedure to do the work.

So, you would expect that main is coupled more with sort1 procedure.
The new method preserves this relation because the coupling value of main and sort1 is greater than the coupling value of sort1 to main.

Dhama measure preserves this relation because it assigns a coupling value of main and sort1 is higher than the value assigned to main and main because in Dhama measure a lower coupling value indicates higher coupling and a higher coupling value indicates lower coupling.

Fenton and Melton measure preserves the relation because it assigns the main and sort1 pair a value of control coupling and this is the only pair that has a value.

Offut, Harrold and Kotle measure preserves the relation by assigning higher values for direct calls (from main to sort1) and lower values for the other direction.

Chen and Lu method does not preserve this relation because it assigns main and sort1 pair a value which is less than the value assigned to main to itself. All methods assigned the system a coupling value which is greater than or equal to main and less than sort1 values.

5.2.2 Module II

Module II is an Ada program that implements the selection sort by using four procedures [Dham95]. The source code of this procedure is listed on figure 3.
5.2.2.1 New Method

The definition matrix is

<table>
<thead>
<tr>
<th></th>
<th>n</th>
<th>to_be_sorted</th>
<th>a_or_d</th>
<th>start</th>
<th>location</th>
</tr>
</thead>
<tbody>
<tr>
<td>main</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>sort2</td>
<td>9</td>
<td>12</td>
<td>3</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>find_max</td>
<td>3</td>
<td>3</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>find_min</td>
<td>3</td>
<td>3</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>exchange</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

Table 30. Module II Definition Matrix.

The coupling matrix is

<table>
<thead>
<tr>
<th></th>
<th>main</th>
<th>sort2</th>
<th>find_max</th>
<th>find_min</th>
<th>exchange</th>
</tr>
</thead>
<tbody>
<tr>
<td>main</td>
<td>0.1208</td>
<td>0.6375</td>
<td>0.1125</td>
<td>0.1125</td>
<td>0.0167</td>
</tr>
<tr>
<td>sort2</td>
<td>0.0637</td>
<td>0.5838</td>
<td>0.1496</td>
<td>0.1496</td>
<td>0.0533</td>
</tr>
<tr>
<td>find_max</td>
<td>0.0422</td>
<td>0.5609</td>
<td>0.1682</td>
<td>0.1682</td>
<td>0.0604</td>
</tr>
<tr>
<td>find_min</td>
<td>0.0422</td>
<td>0.5609</td>
<td>0.1682</td>
<td>0.1682</td>
<td>0.0604</td>
</tr>
<tr>
<td>exchange</td>
<td>0.0167</td>
<td>0.5333</td>
<td>0.1611</td>
<td>0.1611</td>
<td>0.1278</td>
</tr>
</tbody>
</table>

Table 31. Module II Procedure to Procedure Coupling Matrix.

The normalized coupling matrix: (coupling of functions i & j where i ≠ j) is

<table>
<thead>
<tr>
<th></th>
<th>main</th>
<th>sort2</th>
<th>find_max</th>
<th>find_min</th>
<th>exchange</th>
</tr>
</thead>
<tbody>
<tr>
<td>main</td>
<td>1.0000</td>
<td>5.2759</td>
<td>0.9310</td>
<td>0.9310</td>
<td>0.1379</td>
</tr>
<tr>
<td>sort2</td>
<td>0.1092</td>
<td>1.0000</td>
<td>0.2562</td>
<td>0.2562</td>
<td>0.0914</td>
</tr>
<tr>
<td>find_max</td>
<td>0.2508</td>
<td>3.3344</td>
<td>1.0000</td>
<td>1.0000</td>
<td>0.3591</td>
</tr>
<tr>
<td>find_min</td>
<td>0.2508</td>
<td>3.3344</td>
<td>1.0000</td>
<td>1.0000</td>
<td>0.3591</td>
</tr>
<tr>
<td>exchange</td>
<td>0.1304</td>
<td>4.1739</td>
<td>1.2609</td>
<td>1.2609</td>
<td>1.0000</td>
</tr>
</tbody>
</table>

Table 32. Module II Procedure to Procedure Normalized Coupling Matrix.

The cumulative coupling of functions is:
Table 33. Module II Procedures Overall Coupling.

<table>
<thead>
<tr>
<th>Function Name</th>
<th>main</th>
<th>sort2</th>
<th>find_max</th>
<th>find_min</th>
<th>exchange</th>
<th>System</th>
</tr>
</thead>
<tbody>
<tr>
<td>main</td>
<td>4.008526</td>
<td>8.415785</td>
<td>4.196211</td>
<td>4.196211</td>
<td>3.886824</td>
<td>4.940711</td>
</tr>
<tr>
<td>sort2</td>
<td>0.249468</td>
<td>0.118824</td>
<td>0.238310</td>
<td>0.238310</td>
<td>0.257279</td>
<td>0.202400</td>
</tr>
</tbody>
</table>

5.2.2.2 Dhama Method

Table 34. Module II Procedures Coupling Weights Using Dhama Method.

<table>
<thead>
<tr>
<th>Coupling (Increasing)</th>
<th>main</th>
<th>sort2</th>
<th>find_max</th>
<th>find_min</th>
<th>exchange</th>
<th>System</th>
</tr>
</thead>
<tbody>
<tr>
<td>Coupling (Decreasing)</td>
<td>5</td>
<td>8</td>
<td>7</td>
<td>7</td>
<td>5</td>
<td>6.75</td>
</tr>
<tr>
<td>Coupling</td>
<td>0.1000</td>
<td>0.1300</td>
<td>0.1400</td>
<td>0.1400</td>
<td>0.2000</td>
<td>0.1420</td>
</tr>
</tbody>
</table>

The coupling of procedure to procedure is shown in the following table.

Table 35. Module II Procedure to Procedure Coupling Weight Using Dhama Method.

<table>
<thead>
<tr>
<th></th>
<th>main</th>
<th>sort2</th>
<th>find_max</th>
<th>find_min</th>
<th>exchange</th>
</tr>
</thead>
<tbody>
<tr>
<td>main</td>
<td>0.1000</td>
<td>0.1150</td>
<td>0.1200</td>
<td>0.1200</td>
<td>0.1500</td>
</tr>
<tr>
<td>sort2</td>
<td>0.1150</td>
<td>0.1300</td>
<td>0.1350</td>
<td>0.1350</td>
<td>0.1650</td>
</tr>
<tr>
<td>find_max</td>
<td>0.1200</td>
<td>0.1350</td>
<td>0.1400</td>
<td>0.1400</td>
<td>0.1700</td>
</tr>
<tr>
<td>find_min</td>
<td>0.1200</td>
<td>0.1350</td>
<td>0.1400</td>
<td>0.1400</td>
<td>0.1700</td>
</tr>
<tr>
<td>exchange</td>
<td>0.1500</td>
<td>0.1650</td>
<td>0.1700</td>
<td>0.1700</td>
<td>0.2000</td>
</tr>
</tbody>
</table>

5.2.2.3 Fenton and Melton Method

Table 36. Module II Procedure to Procedure Coupling Weight Using Fenton and Melton Method.

<table>
<thead>
<tr>
<th>Function Name</th>
<th>main_sort2</th>
<th>sort2_find_max</th>
<th>sort2_find_min</th>
<th>sort2_exchange</th>
<th>System</th>
</tr>
</thead>
<tbody>
<tr>
<td>Coupling</td>
<td>3.2500</td>
<td>3.200</td>
<td>3.200</td>
<td>3.3333</td>
<td>3.2500</td>
</tr>
</tbody>
</table>

Table 36. Module II Procedure to Procedure Coupling Weight Using Fenton and Melton Method.
5.2.2.4 Offut, Harrold and Kotte Method

<table>
<thead>
<tr>
<th>Function Pair Coupled</th>
<th>main, sort2</th>
<th>sort2, find_max</th>
<th>sort2, find_min</th>
<th>sort2, exchange</th>
<th>system</th>
</tr>
</thead>
<tbody>
<tr>
<td>Calling to Called Coupling</td>
<td>2.7500</td>
<td>2.7000</td>
<td>2.7000</td>
<td>2.8333</td>
<td>2.2458</td>
</tr>
<tr>
<td>Called to Calling Coupling</td>
<td>0.7500</td>
<td>2.7000</td>
<td>2.7000</td>
<td>0.8333</td>
<td></td>
</tr>
</tbody>
</table>

*Table 37. Module II Procedure to Procedure Coupling Weight Using Offut, Harrold and Kotte Method.*

5.2.2.6 Chen and Lu Method

<table>
<thead>
<tr>
<th>Function Name</th>
<th>main</th>
<th>sort2</th>
<th>find_max</th>
<th>find_min</th>
<th>exchange</th>
<th>System</th>
</tr>
</thead>
<tbody>
<tr>
<td>Coupling</td>
<td>6</td>
<td>7</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>3.2</td>
</tr>
</tbody>
</table>

*Table 38. Module II Procedures Coupling Weight Using Chen and Lu Method.*

The procedure to procedure coupling values are shown in Table 39.

<table>
<thead>
<tr>
<th></th>
<th>main</th>
<th>sort2</th>
<th>find_max</th>
<th>find_min</th>
<th>exchange</th>
</tr>
</thead>
<tbody>
<tr>
<td>main</td>
<td>6.0000</td>
<td>6.5000</td>
<td>3.5000</td>
<td>3.5000</td>
<td>3.5000</td>
</tr>
<tr>
<td>sort2</td>
<td>6.5000</td>
<td>7.0000</td>
<td>4.0000</td>
<td>4.0000</td>
<td>4.0000</td>
</tr>
<tr>
<td>find_max</td>
<td>3.5000</td>
<td>4.0000</td>
<td>1.0000</td>
<td>1.0000</td>
<td>1.0000</td>
</tr>
<tr>
<td>find_min</td>
<td>3.5000</td>
<td>4.0000</td>
<td>1.0000</td>
<td>1.0000</td>
<td>1.0000</td>
</tr>
<tr>
<td>exchange</td>
<td>3.5000</td>
<td>4.0000</td>
<td>1.0000</td>
<td>1.0000</td>
<td>1.0000</td>
</tr>
</tbody>
</table>

*Table 39. Module II Procedure to Procedure Coupling Weight Using Chen and Lu Method.*

5.2.2.6 Analysis

The program source code shows that main is coupled more with sort2 than others and sort2 is coupled more with the three routines (find_min, find_max, and exchange) than others. Also, you would expect the coupling of sort2 to both find_max and find_min to be equal.
The new method preserves this because the coupling value of main and sort2 is greater than the coupling value of all others and the coupling values of sort2 to find_max, find_min, and exchange is the greatest among others in the same row. In addition, the coupling matrix rows and columns entries for find_max and find_min procedures are identical. It also, shows that sort2 procedure is the main routine for computation. The new method also shows the coupling that caused by calling scheme and the coupling caused by using the same variables directly or indirectly.

Dhama measure preserves this relation because it assigns a coupling for main and sort2 which is higher than the coupling assigned between main and find_max, find_min, and exchange respectively. Also, it assigns a coupling of sort2 to find_max, find_min, and exchange which is higher than others. Moreover, the coupling values of find_max and find_min procedures are equal.

Fenton and Melton measure preserves the relation because main is coupled more with sort2 and sort2 is coupled more with find_max, find_min, and exchange procedures. Also, the coupling value of sort2 to both find_max and find_min is equal.

Offut, Harrold and Kotle measure preserves the relation by the same reasoning stated on Fenton and Melton measure.

Chen and Lu method preserves the above relation because it assigns the maximum values to main and sort2 and looking at the column of sort2 you see it has the maximum contribution among each row, so it is the main computation routine.
5.2.3 Module III

Module III is a Fortran program that computes the average and the number of elements above the average of an array of real numbers [Law83]. The source code of this procedure is listed on figure 4.

5.2.3.1 New Method

The definition matrix is

<table>
<thead>
<tr>
<th></th>
<th>PROMPT</th>
<th>NUM</th>
<th>LOW</th>
<th>BND</th>
<th>U LP</th>
<th>BND</th>
<th>A MVC</th>
<th>N DIM</th>
<th>S UMC</th>
<th>G AVG</th>
<th>A AVG</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAIN</td>
<td>1</td>
<td>9</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>RCHECK</td>
<td>1</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>AVCALC</td>
<td>0</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>SUMCAL</td>
<td>0</td>
<td>3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>GTAVG</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>3</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>3</td>
<td></td>
</tr>
</tbody>
</table>

*Table 40. Module III Definition Matrix.*

The coupling matrix is

<table>
<thead>
<tr>
<th></th>
<th>MAIN</th>
<th>RCHECK</th>
<th>AVCALC</th>
<th>SUMCAL</th>
<th>GTAVG</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAIN</td>
<td>0.4167</td>
<td>0.1842</td>
<td>0.1404</td>
<td>0.1096</td>
<td>0.1491</td>
</tr>
<tr>
<td>RCHECK</td>
<td>0.3500</td>
<td>0.5500</td>
<td>0.0333</td>
<td>0.0500</td>
<td>0.0167</td>
</tr>
<tr>
<td>AVCALC</td>
<td>0.3333</td>
<td>0.0417</td>
<td>0.2986</td>
<td>0.1771</td>
<td>0.1493</td>
</tr>
<tr>
<td>SUMCAL</td>
<td>0.3472</td>
<td>0.0833</td>
<td>0.2361</td>
<td>0.2153</td>
<td>0.1181</td>
</tr>
<tr>
<td>GTAVG</td>
<td>0.3148</td>
<td>0.0185</td>
<td>0.1327</td>
<td>0.0787</td>
<td>0.4552</td>
</tr>
</tbody>
</table>

*Table 41. Module III Procedure to Procedure Coupling Matrix.*
The normalized coupling matrix: (coupling of functions i & j where i ≠ j) is

<table>
<thead>
<tr>
<th></th>
<th>MAIN</th>
<th>RCHECK</th>
<th>AVCALC</th>
<th>SUMCAL</th>
<th>GTAVG</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAIN</td>
<td>1.0000</td>
<td>0.4421</td>
<td>0.3368</td>
<td>0.2632</td>
<td>0.3579</td>
</tr>
<tr>
<td>RCHECK</td>
<td>0.6364</td>
<td>1.0000</td>
<td>0.0606</td>
<td>0.0909</td>
<td>0.0303</td>
</tr>
<tr>
<td>AVCALC</td>
<td>1.1163</td>
<td>0.1395</td>
<td>1.0000</td>
<td>0.5930</td>
<td>0.5000</td>
</tr>
<tr>
<td>SUMCAL</td>
<td>1.6129</td>
<td>0.3871</td>
<td>1.0968</td>
<td>1.0000</td>
<td>0.5484</td>
</tr>
<tr>
<td>GTAVG</td>
<td>0.6915</td>
<td>0.0407</td>
<td>0.2915</td>
<td>0.1729</td>
<td>1.0000</td>
</tr>
</tbody>
</table>

Table 42. Module III Procedure to Procedure Normalized Coupling Matrix.

The cumulative coupling of functions is:

<table>
<thead>
<tr>
<th>Function Name</th>
<th>MAIN</th>
<th>RCHECK</th>
<th>AVCALC</th>
<th>SUMCAL</th>
<th>GTAVG</th>
<th>System</th>
</tr>
</thead>
<tbody>
<tr>
<td>Coupling</td>
<td>2.728536</td>
<td>0.913798</td>
<td>2.067292</td>
<td>2.382566</td>
<td>1.316597</td>
<td>1.881758</td>
</tr>
<tr>
<td>Coupling Inverse</td>
<td>0.366497</td>
<td>1.094333</td>
<td>0.483725</td>
<td>0.419716</td>
<td>0.759534</td>
<td>0.531418</td>
</tr>
</tbody>
</table>

Table 43. Module III Procedures Overall Coupling.

5.2.3.2 Dhama Method

<table>
<thead>
<tr>
<th>Function Name</th>
<th>MAIN</th>
<th>RCHECK</th>
<th>AVCALC</th>
<th>SUMCAL</th>
<th>GTAVG</th>
<th>System</th>
</tr>
</thead>
<tbody>
<tr>
<td>Coupling (Increasing)</td>
<td>9</td>
<td>8</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>6.4</td>
</tr>
<tr>
<td>Coupling (Decreasing)</td>
<td>0.1111</td>
<td>0.1250</td>
<td>0.2000</td>
<td>0.2000</td>
<td>0.2000</td>
<td>0.1672</td>
</tr>
</tbody>
</table>

Table 44. Module III Procedures Coupling Weights Using Dhama Method.

The coupling of procedure to procedure is shown in the following table.

<table>
<thead>
<tr>
<th></th>
<th>MAIN</th>
<th>RCHECK</th>
<th>AVCALC</th>
<th>SUMCAL</th>
<th>GTAVG</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAIN</td>
<td>0.1111</td>
<td>0.1181</td>
<td>0.1556</td>
<td>0.1556</td>
<td>0.1556</td>
</tr>
<tr>
<td>RCHECK</td>
<td>0.1181</td>
<td>0.1250</td>
<td>0.1625</td>
<td>0.1625</td>
<td>0.1625</td>
</tr>
<tr>
<td>AVCALC</td>
<td>0.1556</td>
<td>0.1625</td>
<td>0.2000</td>
<td>0.2000</td>
<td>0.2000</td>
</tr>
<tr>
<td>SUMCAL</td>
<td>0.1556</td>
<td>0.1625</td>
<td>0.2000</td>
<td>0.2000</td>
<td>0.2000</td>
</tr>
<tr>
<td>GTAVG</td>
<td>0.1556</td>
<td>0.1625</td>
<td>0.2000</td>
<td>0.2000</td>
<td>0.2000</td>
</tr>
</tbody>
</table>

Table 45. Module III Procedure to Procedure Coupling Weights Using Dhama Method.
### 5.2.3.3 Fenton and Melton Method

<table>
<thead>
<tr>
<th>Function Pair Coupled</th>
<th>MAIN, RCHECK</th>
<th>MAIN, AVCALC</th>
<th>MAIN, GTA VG</th>
<th>A VCALC, SUMCAL</th>
<th>System</th>
</tr>
</thead>
</table>

*Table 46. Module III Procedure to Procedure Coupling Weights Using Fenton and Melton Method.*

### 5.2.3.4 Offut, Harrold and Kotle Method

<table>
<thead>
<tr>
<th>Function Pair Coupled</th>
<th>MAIN, RCHECK</th>
<th>MAIN, AVCALC</th>
<th>MAIN, GTA VG</th>
<th>A VCALC, SUMCAL</th>
<th>System</th>
</tr>
</thead>
<tbody>
<tr>
<td>Calling to Called Coupling Level</td>
<td>2.7000</td>
<td>0.7000</td>
<td>2.7000</td>
<td>2.7500</td>
<td>1.9625</td>
</tr>
<tr>
<td>Called to Calling Coupling Level</td>
<td>2.7000</td>
<td>0.7000</td>
<td>2.7000</td>
<td>0.7500</td>
<td></td>
</tr>
</tbody>
</table>

*Table 47. Module III Procedure to Procedure Coupling Weights Using Offut, Harrold and Kotle Method.*

### 5.2.3.5 Chen and Lu Method

<table>
<thead>
<tr>
<th>Function Name</th>
<th>MAIN</th>
<th>RCHECK</th>
<th>AVCALC</th>
<th>SUMCAL</th>
<th>GTA VG</th>
<th>System</th>
</tr>
</thead>
<tbody>
<tr>
<td>Coupling</td>
<td>8</td>
<td>1</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>2.8</td>
</tr>
</tbody>
</table>

*Table 48. Module III Procedures Coupling Weights Using Chen and Lu Method.*

The procedure to procedure coupling values are shown in Table 49.

<table>
<thead>
<tr>
<th></th>
<th>MAIN</th>
<th>RCHECK</th>
<th>AVCALC</th>
<th>SUMCAL</th>
<th>GTA VG</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAIN</td>
<td>8.0000</td>
<td>4.5000</td>
<td>5.5000</td>
<td>4.5000</td>
<td>4.5000</td>
</tr>
<tr>
<td>RCHECK</td>
<td>4.5000</td>
<td>1.0000</td>
<td>2.0000</td>
<td>1.0000</td>
<td>1.0000</td>
</tr>
<tr>
<td>AVCALC</td>
<td>5.5000</td>
<td>2.0000</td>
<td>3.0000</td>
<td>2.0000</td>
<td>2.0000</td>
</tr>
<tr>
<td>SUMCAL</td>
<td>4.5000</td>
<td>1.0000</td>
<td>2.0000</td>
<td>1.0000</td>
<td>1.0000</td>
</tr>
<tr>
<td>GTA VG</td>
<td>4.5000</td>
<td>1.0000</td>
<td>2.0000</td>
<td>1.0000</td>
<td>1.0000</td>
</tr>
</tbody>
</table>

*Table 49. Module III Procedure to Procedure Coupling Weight Using Chen and Lu Method.*
5.2.3.6 Analysis

The program source code shows that MAIN is the driver of computation. So, you would expect that MAIN is attached more with RCHECK, AVCALC and GTAVG than others. Moreover, you would expect that AVCALC is attached more to SUMCALC than others.

The new method preserves this because the coupling value of MAIN and RCHECK, AVCALC and GTAVG is greater than the coupling value of all others. MAIN procedure is attached more with RCHECK than AVCALC than GTAVG procedures. The new method shows that AVCALC is attached more with MAIN than SUMCAL but it is attached more with SUMCAL than the remaining. It also, shows that MAIN procedure is the main routine for computation. The new method also shows the coupling caused by calling scheme and the coupling caused by using the same variables directly or indirectly.

Dhama measure preserves this relation because MAIN is coupled equally to AVCALC and GTAVG but higher than RCHECK and these contribute the high proportion among others. Also, AVCALC is coupled more with SUMCAL.

Fenton and Melton measure preserves the relation because MAIN is coupled more with RCHECK, GAVG, MAIN and AVCALC. Also, AVCALC is coupled maximally with SUMCAL.

Offut, Harrold and Kote measure preserves the relation by the same reasoning cited on Fenton and Melton measure.
Chen and Lu method preserves the above relation because MAIN is coupled more with itself, AVCALC, RCHECK and GTAVG than others. Also, AVCALC is coupled more with its MAIN, itself and SUMCAL than others.

5.2.4 Module IV

Module IV is a PL/1 program that computes the number of days between two specified dates entered in any order. The source code of this procedure is listed on figure 5.

5.2.4.1 New Method

The definition matrix is

<table>
<thead>
<tr>
<th></th>
<th>S DATE</th>
<th>E DATE</th>
<th>N DAY</th>
<th>VAL DAY</th>
<th>LEAP</th>
<th>YS</th>
<th>IE</th>
<th>YE</th>
<th>LEAP</th>
<th>YS</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAIN</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>DAYS</td>
<td>6</td>
<td>6</td>
<td>1</td>
<td>2</td>
<td>18</td>
<td>4</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>VALDAY</td>
<td>3</td>
<td>3</td>
<td>0</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>LEAP</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>6</td>
<td>12</td>
<td>3</td>
<td>3</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>LEAP</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>3</td>
<td></td>
</tr>
</tbody>
</table>

Table 50. Module IV Definition Matrix.
The coupling matrix is

<table>
<thead>
<tr>
<th></th>
<th>MAIN</th>
<th>DAYS</th>
<th>VALDAY</th>
<th>LEAP</th>
<th>LEAP</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAIN</td>
<td>0.2333</td>
<td>0.5667</td>
<td>0.2000</td>
<td>0.0000</td>
<td>0.0000</td>
</tr>
<tr>
<td>DAYS</td>
<td>0.0436</td>
<td>0.6077</td>
<td>0.1179</td>
<td>0.2308</td>
<td>0.0000</td>
</tr>
<tr>
<td>VALDAY</td>
<td>0.0600</td>
<td>0.4600</td>
<td>0.3550</td>
<td>0.0000</td>
<td>0.1250</td>
</tr>
<tr>
<td>LEAP</td>
<td>0.0000</td>
<td>0.3750</td>
<td>0.0000</td>
<td>0.6250</td>
<td>0.0000</td>
</tr>
<tr>
<td>LEAP</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.3125</td>
<td>0.0000</td>
<td>0.6875</td>
</tr>
</tbody>
</table>

*Table 51. Module IV Procedure to Procedure Coupling Matrix.*

The normalized coupling matrix: (coupling of functions i & j where i≠j) is

<table>
<thead>
<tr>
<th></th>
<th>MAIN</th>
<th>DAYS</th>
<th>VALDAY</th>
<th>LEAP</th>
<th>LEAP</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAIN</td>
<td>1.0000</td>
<td>2.4286</td>
<td>0.8571</td>
<td>0.0000</td>
<td>0.0000</td>
</tr>
<tr>
<td>DAYS</td>
<td>0.0717</td>
<td>1.0000</td>
<td>0.1941</td>
<td>0.3797</td>
<td>0.0000</td>
</tr>
<tr>
<td>VALDAY</td>
<td>0.1690</td>
<td>1.2958</td>
<td>1.0000</td>
<td>0.0000</td>
<td>0.3521</td>
</tr>
<tr>
<td>LEAP</td>
<td>0.0000</td>
<td>0.6000</td>
<td>0.0000</td>
<td>1.0000</td>
<td>0.0000</td>
</tr>
<tr>
<td>LEAP</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.4545</td>
<td>0.0000</td>
<td>1.0000</td>
</tr>
</tbody>
</table>

*Table 52. Module IV Procedure to Procedure Normalized Coupling Matrix.*

The cumulative coupling of functions is:

<table>
<thead>
<tr>
<th>Function Name</th>
<th>MAIN</th>
<th>DAYS</th>
<th>VALDAY</th>
<th>LEAP</th>
<th>LEAP</th>
<th>System</th>
</tr>
</thead>
<tbody>
<tr>
<td>Coupling</td>
<td>1.763229</td>
<td>2.484958</td>
<td>1.661341</td>
<td>0.489873</td>
<td>0.403329</td>
<td>1.360546</td>
</tr>
<tr>
<td>Coupling Inverse</td>
<td>0.567141</td>
<td>0.402421</td>
<td>0.601923</td>
<td>2.041344</td>
<td>2.479365</td>
<td>0.734999</td>
</tr>
</tbody>
</table>

*Table 53. Module IV Procedures Overall Coupling.*

### 5.2.4.2 Dhama Method

<table>
<thead>
<tr>
<th>Function Name</th>
<th>MAIN</th>
<th>DAYS</th>
<th>VALDAY</th>
<th>LEAP</th>
<th>LEAP</th>
<th>System</th>
</tr>
</thead>
<tbody>
<tr>
<td>Coupling</td>
<td>8</td>
<td>8</td>
<td>5</td>
<td>9</td>
<td>4</td>
<td>6.8</td>
</tr>
<tr>
<td>Coupling Inverse</td>
<td>0.1250</td>
<td>0.1250</td>
<td>0.2000</td>
<td>0.1111</td>
<td>0.2500</td>
<td>0.1622</td>
</tr>
</tbody>
</table>

*Table 54. Module IV Procedures Coupling Weights Using Dhama Method.*

The coupling of procedure to procedure is shown in the following table.
<table>
<thead>
<tr>
<th>MAIN</th>
<th>DAYS</th>
<th>VALDAY</th>
<th>LEAP</th>
<th>LEAP</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAIN</td>
<td>0.1250</td>
<td>0.1250</td>
<td>0.1625</td>
<td>0.1181</td>
</tr>
<tr>
<td>DAYS</td>
<td>0.1250</td>
<td>0.1250</td>
<td>0.1875</td>
<td>0.1181</td>
</tr>
<tr>
<td>VALDAY</td>
<td>0.1625</td>
<td>0.1875</td>
<td>0.2000</td>
<td>0.1556</td>
</tr>
<tr>
<td>LEAP</td>
<td>0.1181</td>
<td>0.1181</td>
<td>0.1556</td>
<td>0.1111</td>
</tr>
<tr>
<td>LEAP</td>
<td>0.1875</td>
<td>0.1875</td>
<td>0.2250</td>
<td>0.1801</td>
</tr>
</tbody>
</table>

*Table 55. Module IV Procedure to Procedure Coupling Weights Using Dhama Method.*

### 5.2.4.3 Fenton and Melton Method

<table>
<thead>
<tr>
<th>Function Pair Coupled</th>
<th>MAIN, DAYS</th>
<th>DAYS, VALDAY</th>
<th>DAYS, LEAP</th>
<th>VALDAY, LEAP</th>
<th>System</th>
</tr>
</thead>
<tbody>
<tr>
<td>Coupling Level</td>
<td>3.2500</td>
<td>3.3333</td>
<td>3.3333</td>
<td>3.3333</td>
<td>3.3333</td>
</tr>
</tbody>
</table>

*Table 56. Module IV Procedure to Procedure Coupling Weights Using Fenton and Melton Method.*

### 5.2.4.4 Offut, Harrold and Kotle Method

<table>
<thead>
<tr>
<th>Function Pair Coupled</th>
<th>MAIN, DAYS</th>
<th>DAYS, VALDAY</th>
<th>DAYS, LEAP</th>
<th>VALDAY, LEAP</th>
<th>System</th>
</tr>
</thead>
<tbody>
<tr>
<td>Calling to Called Coupling Level</td>
<td>2.7500</td>
<td>2.8333</td>
<td>2.8333</td>
<td>2.8333</td>
<td>2.0624</td>
</tr>
<tr>
<td>Calling to Called Coupling Level</td>
<td>0.7500</td>
<td>2.8333</td>
<td>0.8333</td>
<td>0.8333</td>
<td>0.8333</td>
</tr>
</tbody>
</table>

*Table 57. Module IV Procedure to Procedure Coupling Weights Using Offut, Harrold and Kotle Method.*

### 5.2.4.5 Chen and Lu Method

<table>
<thead>
<tr>
<th>Function Name</th>
<th>MAIN</th>
<th>DAYS</th>
<th>VALDAY</th>
<th>LEAP</th>
<th>LEAP</th>
<th>System</th>
</tr>
</thead>
<tbody>
<tr>
<td>Coupling</td>
<td>6</td>
<td>6</td>
<td>1</td>
<td>3</td>
<td>1</td>
<td>3.4</td>
</tr>
</tbody>
</table>

*Table 58. Module IV Procedures Coupling Weights Using Chen and Lu Method.*
The procedure to procedure coupling values are shown in Table 59.

<table>
<thead>
<tr>
<th></th>
<th>MAIN</th>
<th>DAYS</th>
<th>VALDAY</th>
<th>LEAP</th>
<th>LEAP</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAIN</td>
<td>6.0000</td>
<td>6.0000</td>
<td>3.5000</td>
<td>4.5000</td>
<td>3.5000</td>
</tr>
<tr>
<td>DAYS</td>
<td>6.0000</td>
<td>6.0000</td>
<td>3.5000</td>
<td>4.5000</td>
<td>3.5000</td>
</tr>
<tr>
<td>VALDAY</td>
<td>3.5000</td>
<td>3.5000</td>
<td>1.0000</td>
<td>2.0000</td>
<td>1.0000</td>
</tr>
<tr>
<td>LEAP</td>
<td>4.5000</td>
<td>4.5000</td>
<td>2.0000</td>
<td>3.0000</td>
<td>2.0000</td>
</tr>
<tr>
<td>LEAP</td>
<td>3.5000</td>
<td>3.5000</td>
<td>1.0000</td>
<td>2.0000</td>
<td>1.0000</td>
</tr>
</tbody>
</table>

*Table 59. Module IV Procedure to Procedure Coupling Weight Using Chen and Lu Method.*

### 5.2.4.6 Analysis

The program source code shows that DAYS is the driver of computation. So, you would expect that DAYS is attached more with MAIN than others. Also, you would expect that DAYS is attached more with its LEAP and VALDAY procedures more than others. Finally, you would expect that VALDAY is attached with it its LEAP more than others.

The new method preserves this because the coupling value of MAIN and DAYS is greater than others. DAYS procedure is attached more with itself than its LEAP and VALDAY procedures. The new method shows that VALDAY is attached more with DAYS than VALDAY and then LEAP than the remaining. It also, shows that DAYS procedure is the main routine for computation. The new method shows the coupling of MAIN to both LEAP routines to be zero because they do not have any parameters shared with MAIN. However, it shows that DAYS is coupled with its LEAP and VALDAY is coupled with its LEAP. In addition, it shows that DAYS has no relationships with VALDAY internal LEAP routine and VALDAY has no relationships with DAYS internal routine. The main reason is from the way these routine are being called that caused by calling scheme and the
coupling caused by using the same variables directly or indirectly.

Dharma measure does not preserve this relation because MAIN is coupled more with DAYS’s LEAP, DAYS, MAIN, VALDAY and VALDAY’s LEAP. However, DAYS is attached more with LEAP which is attached less with VALDAY.

Fenton and Melton measure does not preserve the relation because it assigns MAIN and DAYS the minimum value while it assigns the remaining pairs the same value.

Offut, Harrold and Kotle measure does not preserve the relation by the same reasoning mentioned on Fenton and Melton measure.

Chen and Lu method preserves the above relation because MAIN is attached more to itself, DAYS, DAY’s LEAP, VALDAY and its LEAP. Also, DAYS is attached more to MAIN, itself its LEAP and VALDAY.

5.2.5 Module V

Module V is a Pascal program that plays a dice game with the user [Grz82]. The source code of this procedure is listed on figure 6.
5.2.5.1 New Method

The definition matrix is

<table>
<thead>
<tr>
<th>Module</th>
<th>SEED</th>
<th>AMOUNT</th>
<th>BET</th>
<th>WON</th>
<th>FINISHED</th>
<th>VALUE</th>
<th>RANDOM</th>
<th>N</th>
</tr>
</thead>
<tbody>
<tr>
<td>CRAPS</td>
<td>5</td>
<td>3</td>
<td>2</td>
<td>3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>INITIALIZE</td>
<td>1</td>
<td>3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>ACCEPTBET</td>
<td>0</td>
<td>3</td>
<td>3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>PLAY</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>3</td>
<td>0</td>
<td>6</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>ROLLDICE</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>RANDOM</td>
<td>20</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>4</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>UPDATE</td>
<td>0</td>
<td>3</td>
<td>1</td>
<td>3</td>
<td>3</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Table 60. Module V Description Matrix.

The coupling matrix is

<table>
<thead>
<tr>
<th></th>
<th>CRAPS</th>
<th>INITIALIZE</th>
<th>ACCEPTBET</th>
<th>PLAY</th>
<th>ROLLDICE</th>
<th>RANDOM</th>
<th>UPDATE</th>
</tr>
</thead>
<tbody>
<tr>
<td>CRAPS</td>
<td>0.2919</td>
<td>0.0628</td>
<td>0.1167</td>
<td>0.0500</td>
<td>0.0000</td>
<td>0.2564</td>
<td>0.2222</td>
</tr>
<tr>
<td>INITIALIZE</td>
<td>0.2356</td>
<td>0.1971</td>
<td>0.1875</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.1923</td>
<td>0.1875</td>
</tr>
<tr>
<td>ACCEPTBET</td>
<td>0.2917</td>
<td>0.1250</td>
<td>0.3750</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.2083</td>
</tr>
<tr>
<td>PLAY</td>
<td>0.0833</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.6250</td>
<td>0.1667</td>
<td>0.0000</td>
<td>0.1250</td>
</tr>
<tr>
<td>ROLLDICE</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.1500</td>
<td>0.4500</td>
<td>0.4000</td>
<td>0.0000</td>
</tr>
<tr>
<td>RANDOM</td>
<td>0.1374</td>
<td>0.0275</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.1429</td>
<td>0.6923</td>
<td>0.0000</td>
</tr>
<tr>
<td>UPDATE</td>
<td>0.3333</td>
<td>0.0750</td>
<td>0.1250</td>
<td>0.1125</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.3542</td>
</tr>
</tbody>
</table>

Table 61. Module V Procedure to Procedure Coupling Matrix.

The normalized coupling matrix: (coupling of functions i & j where i≠j) is
Table 62. Module V Procedure to Procedure Normalized Coupling Matrix.

<table>
<thead>
<tr>
<th></th>
<th>CRAPS</th>
<th>INITIALIZE</th>
<th>ACCEPT BET</th>
<th>PLAY</th>
<th>ROLL DICE</th>
<th>RANDOM</th>
<th>UP DATE</th>
</tr>
</thead>
<tbody>
<tr>
<td>CRAPS</td>
<td>1.0000</td>
<td>0.2152</td>
<td>0.3997</td>
<td>0.1713</td>
<td>0.0000</td>
<td>0.8785</td>
<td>0.7613</td>
</tr>
<tr>
<td>INITIALIZED</td>
<td>1.1951</td>
<td>1.0000</td>
<td>0.9512</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.9756</td>
<td>0.9512</td>
</tr>
<tr>
<td>ACCEPT BET</td>
<td>0.7778</td>
<td>0.3333</td>
<td>1.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.5556</td>
</tr>
<tr>
<td>PLAY</td>
<td>0.1333</td>
<td>0.0000</td>
<td>0.0000</td>
<td>1.0000</td>
<td>0.2667</td>
<td>0.0000</td>
<td>0.2000</td>
</tr>
<tr>
<td>ROLLDICE</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.3333</td>
<td>1.0000</td>
<td>0.8889</td>
<td>0.0000</td>
</tr>
<tr>
<td>RANDOM</td>
<td>0.1984</td>
<td>0.0397</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.2063</td>
<td>1.0000</td>
<td>0.0000</td>
</tr>
<tr>
<td>UPDATE</td>
<td>0.9412</td>
<td>0.2118</td>
<td>0.3529</td>
<td>0.3176</td>
<td>0.0000</td>
<td>0.0000</td>
<td>1.0000</td>
</tr>
</tbody>
</table>

The cumulative coupling of functions is:

<table>
<thead>
<tr>
<th>Function Name</th>
<th>CRAPS</th>
<th>INITIALIZE</th>
<th>ACCEPT BET</th>
<th>PLAY</th>
<th>ROLL DICE</th>
<th>RANDOM</th>
<th>UP DATE</th>
<th>System</th>
</tr>
</thead>
<tbody>
<tr>
<td>Coupling</td>
<td>2.835942</td>
<td>2.436589</td>
<td>1.685267</td>
<td>0.71114</td>
<td>0.84761</td>
<td>1.59371</td>
<td>2.14582</td>
<td>1.75087</td>
</tr>
<tr>
<td>Coupling Inverse</td>
<td>0.352617</td>
<td>0.410410</td>
<td>0.593378</td>
<td>1.40618</td>
<td>1.17977</td>
<td>0.62746</td>
<td>0.46602</td>
<td>0.57114</td>
</tr>
</tbody>
</table>

Table 63. Module V Procedures Overall Coupling.

5.2.5.2 Dhama Method

<table>
<thead>
<tr>
<th>Function Name</th>
<th>CRAPS</th>
<th>INITIALIZE</th>
<th>ACCEPT BET</th>
<th>PLAY</th>
<th>ROLL DICE</th>
<th>RANDOM</th>
<th>UP DATE</th>
<th>System</th>
</tr>
</thead>
<tbody>
<tr>
<td>Coupling</td>
<td>8</td>
<td>4</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>8</td>
<td>5.7143</td>
</tr>
<tr>
<td>Coupling Inverse</td>
<td>0.1250</td>
<td>0.2500</td>
<td>0.2000</td>
<td>0.2000</td>
<td>0.2000</td>
<td>0.1250</td>
<td>0.1857</td>
<td></td>
</tr>
</tbody>
</table>

Table 64. Module V Procedures Coupling Weights Using Dhama Method.

The coupling of procedure to procedure is shown in the following table.
<table>
<thead>
<tr>
<th></th>
<th>CRAPS</th>
<th>INITIALIZE</th>
<th>ACCEPT Bet</th>
<th>PLAY</th>
<th>ROLL DICE</th>
<th>RANDOM</th>
<th>UP DATE</th>
</tr>
</thead>
<tbody>
<tr>
<td>CRAPS</td>
<td>0.1250</td>
<td>0.1875</td>
<td>0.1625</td>
<td>0.1625</td>
<td>0.2250</td>
<td>0.1625</td>
<td>0.1250</td>
</tr>
<tr>
<td>INITALIZE</td>
<td>0.1875</td>
<td>0.2500</td>
<td>0.2250</td>
<td>0.2250</td>
<td>0.2500</td>
<td>0.2250</td>
<td>0.1875</td>
</tr>
<tr>
<td>ACCEPTBET</td>
<td>0.1625</td>
<td>0.2250</td>
<td>0.2000</td>
<td>0.2000</td>
<td>0.2000</td>
<td>0.2000</td>
<td>0.1625</td>
</tr>
<tr>
<td>PLAY</td>
<td>0.1625</td>
<td>0.2250</td>
<td>0.2000</td>
<td>0.2000</td>
<td>0.2000</td>
<td>0.2000</td>
<td>0.1625</td>
</tr>
<tr>
<td>ROLLDICE</td>
<td>0.1625</td>
<td>0.2250</td>
<td>0.2000</td>
<td>0.2000</td>
<td>0.2000</td>
<td>0.2000</td>
<td>0.1625</td>
</tr>
<tr>
<td>RANDOM</td>
<td>0.1625</td>
<td>0.2250</td>
<td>0.2000</td>
<td>0.2000</td>
<td>0.2000</td>
<td>0.2000</td>
<td>0.1625</td>
</tr>
<tr>
<td>UPDATE</td>
<td>0.1250</td>
<td>0.1875</td>
<td>0.1625</td>
<td>0.1625</td>
<td>0.1625</td>
<td>0.1625</td>
<td>0.1250</td>
</tr>
</tbody>
</table>

Table 65. Module V Procedure to Procedure Coupling Weights Using Dhama Method.

### 5.2.5.3 Fenton and Melton Method

<table>
<thead>
<tr>
<th>Function Pair Coupled</th>
<th>CRAPS, INITIALIZE</th>
<th>CRAPS, ACCEPT BET</th>
<th>CRAPS, PLAY</th>
<th>CRAPS, UP DATE</th>
<th>CRAPS, RANDOM</th>
<th>PLAY, ROLL DICE</th>
<th>ROLLDICE, RANDOM</th>
<th>System</th>
</tr>
</thead>
</table>

Table 66. Module V Procedure to Procedure Coupling Weights Using Fenton and Melton Method.

### 5.2.5.4 Offut, Harrold and Kotle Method

<table>
<thead>
<tr>
<th>Function Pair Coupled</th>
<th>CRAPS, INITIALIZE</th>
<th>CRAPS, ACCEPT BET</th>
<th>CRAPS, PLAY</th>
<th>CRAPS, UP DATE</th>
<th>CRAPS, RANDOM</th>
<th>PLAY, ROLL DICE</th>
<th>ROLLDICE, RANDOM</th>
<th>System</th>
</tr>
</thead>
<tbody>
<tr>
<td>Calling to Called</td>
<td>2.8333</td>
<td>2.8333</td>
<td>3.0000</td>
<td>2.7000</td>
<td>4.8333</td>
<td>1.0000</td>
<td>1.0000</td>
<td>2.4571</td>
</tr>
<tr>
<td>Coupling</td>
<td>0.83333</td>
<td>2.8333</td>
<td>1.0000</td>
<td>2.7000</td>
<td>4.8333</td>
<td>3.0000</td>
<td>1.0000</td>
<td></td>
</tr>
</tbody>
</table>

Table 67. Module V Procedure to Procedure Coupling Weights Using Offut, Harrold and Kotle Method.
5.2.5.5 Chen and Lu Method

<table>
<thead>
<tr>
<th>Function Name</th>
<th>CRAPS</th>
<th>INITIALIZE</th>
<th>ACCEPT BET</th>
<th>PLAY</th>
<th>ROLL DICE</th>
<th>RANDOM</th>
<th>UP DATE</th>
<th>System</th>
</tr>
</thead>
<tbody>
<tr>
<td>Coupling</td>
<td>11</td>
<td>1</td>
<td>1</td>
<td>4</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>3.1428</td>
</tr>
</tbody>
</table>

*Table 68. Module V Procedure Coupling Weights Using Chen and Lu Method.*

The procedure to procedure coupling values as in the following table.

<table>
<thead>
<tr>
<th></th>
<th>CRAPS</th>
<th>INITIALIZE</th>
<th>ACCEPT BET</th>
<th>PLAY</th>
<th>ROLL DICE</th>
<th>RANDOM</th>
<th>UP DATE</th>
</tr>
</thead>
<tbody>
<tr>
<td>CRAPS</td>
<td>11.0000</td>
<td>6.0000</td>
<td>6.0000</td>
<td>7.5000</td>
<td>7.0000</td>
<td>6.0000</td>
<td>6.0000</td>
</tr>
<tr>
<td>INITIALIZE</td>
<td>6.0000</td>
<td>1.0000</td>
<td>1.0000</td>
<td>2.5000</td>
<td>2.0000</td>
<td>1.0000</td>
<td>1.0000</td>
</tr>
<tr>
<td>ACCEPTBET</td>
<td>6.0000</td>
<td>1.0000</td>
<td>1.0000</td>
<td>2.5000</td>
<td>2.0000</td>
<td>1.0000</td>
<td>1.0000</td>
</tr>
<tr>
<td>PLAY</td>
<td>7.5000</td>
<td>2.5000</td>
<td>2.5000</td>
<td>4.0000</td>
<td>3.5000</td>
<td>2.5000</td>
<td>2.5000</td>
</tr>
<tr>
<td>ROLLDICE</td>
<td>7.0000</td>
<td>2.0000</td>
<td>2.0000</td>
<td>3.5000</td>
<td>3.0000</td>
<td>2.0000</td>
<td>2.0000</td>
</tr>
<tr>
<td>RANDOM</td>
<td>6.0000</td>
<td>1.0000</td>
<td>1.0000</td>
<td>2.5000</td>
<td>2.0000</td>
<td>1.0000</td>
<td>1.0000</td>
</tr>
<tr>
<td>UPDATE</td>
<td>6.0000</td>
<td>1.0000</td>
<td>1.0000</td>
<td>2.5000</td>
<td>2.0000</td>
<td>1.0000</td>
<td>1.0000</td>
</tr>
</tbody>
</table>

*Table 69. Module V Procedure by Procedure Coupling Values Using Chen and Lu Method.*

5.2.5.6 Analysis

The program source code shows that CRAPS is the driver of computation. So, you would expect that CRAPS is attached more with INITIALIZE, ACCEPTBET, PLAY and UPDATE than others. Also, you would expect that PLAY is attached more with ROLLDICE and CRAPS procedures than others. Finally, you would expect that ROLLDICE is attached with it RANDOM and PLAY than others.

The new method preserves this because the coupling value of CRAPS is concentrated on CRAPS, INITIALIZE, ACCEPTBET, PLAY and UPDATE than others. PLAY procedure is attached more with itself than ROLLDICE than RANDOM procedures. The new method shows that ROLLDICE is
attached more with itself than PLAY and then RANDOM than the remaining. It also, shows that CRAPS procedure is the main routine for computation. The new method shows the coupling of 14 pairs of routines to be zero because they do not have any parameters shared with each others.

Dhamal measure does not preserve this relation because CRAPS is coupled more with UPDATE, ACCEPTBET, PLAY, RANDOM, and INITIALIZE respectively. However, PLAY is attached less with ROLLDICE and ROLLDICE is attached less with RANDOM.

Fenton and Melton measure preserves the relation because CRAPS is coupled more with RANDOM, PLAY, ACCEPTBET and INITIALIZE than others. Also, PLAY is coupled more with ROLLDICE and ROLLDICE is coupled more with RANDOM.

Offut, Harrold and Kotle measure preserves the relation by the same reasoning referred on Fenton and Melton measure.

Chen and Lu method does not preserve the above relation because although CRAPS is attached more to itself, PLAY, INITIALIZE, RANDOM and UPDATE procedures and PLAY is attached more with ROLLDICE, ROLLDICE is coupled minimally with RANDOM.

5.2.6 Module VI

Module VI is a C program that computes the statistics about a document in a file such as the number of
words, blank lines, sentences and paragraphs. The source code of module VI procedures is listed on figure 7.

### 5.2.6.1 New Method

The definition matrix is

<table>
<thead>
<tr>
<th></th>
<th>new_doc</th>
<th>lines</th>
<th>i_o_area</th>
<th>par lines</th>
<th>b_line_s</th>
<th>all_phas</th>
<th>w_ears</th>
<th>s_phrases</th>
<th>e_sentences</th>
<th>c_title</th>
<th>s_tall_line</th>
<th>c_tall_line</th>
<th>s_tall_line</th>
<th>b_tall_line</th>
<th>s_tall_line</th>
<th>o_tall_line</th>
<th>r_tall_line</th>
<th>s_tall_line</th>
<th>i_o_num</th>
</tr>
</thead>
<tbody>
<tr>
<td>main</td>
<td>3</td>
<td>5</td>
<td>2</td>
<td>0</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>1</td>
<td>5</td>
<td>2</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>initialize</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>open_files</td>
<td>8</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>readdoc</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>4</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>writedoc</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>4</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>do_stats</td>
<td>0</td>
<td>5</td>
<td>0</td>
<td>5</td>
<td>7</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>0</td>
</tr>
<tr>
<td>write_stats</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>close_files</td>
<td>8</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>8</td>
<td>8</td>
<td>0</td>
</tr>
</tbody>
</table>

*Table 70. Module VI Definition Matrix.*
The coupling matrix is

<table>
<thead>
<tr>
<th></th>
<th>main</th>
<th>initialize</th>
<th>open_files</th>
<th>read_doc</th>
<th>write_doc</th>
<th>do_stats</th>
<th>write_stats</th>
<th>close_files</th>
</tr>
</thead>
<tbody>
<tr>
<td>main</td>
<td>0.4429</td>
<td>0.0359</td>
<td>0.0435</td>
<td>0.0911</td>
<td>0.0899</td>
<td>0.2111</td>
<td>0.0522</td>
<td>0.0435</td>
</tr>
<tr>
<td>initialize</td>
<td>0.5000</td>
<td>0.3333</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.1667</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
</tr>
<tr>
<td>open_files</td>
<td>0.3495</td>
<td>0.0000</td>
<td>0.1123</td>
<td>0.0833</td>
<td>0.0769</td>
<td>0.1208</td>
<td>0.1449</td>
<td>0.1123</td>
</tr>
<tr>
<td>read doc</td>
<td>0.4884</td>
<td>0.0000</td>
<td>0.0556</td>
<td>0.2870</td>
<td>0.1134</td>
<td>0.0000</td>
<td>0.0556</td>
<td>0.0000</td>
</tr>
<tr>
<td>writedoc</td>
<td>0.3469</td>
<td>0.0333</td>
<td>0.0369</td>
<td>0.0817</td>
<td>0.2309</td>
<td>0.0972</td>
<td>0.1361</td>
<td>0.0369</td>
</tr>
<tr>
<td>do_stats</td>
<td>0.1877</td>
<td>0.0000</td>
<td>0.0134</td>
<td>0.0000</td>
<td>0.0224</td>
<td>0.4092</td>
<td>0.3540</td>
<td>0.0134</td>
</tr>
<tr>
<td>write_stats</td>
<td>0.0480</td>
<td>0.0000</td>
<td>0.0166</td>
<td>0.0000</td>
<td>0.0324</td>
<td>0.3658</td>
<td>0.5207</td>
<td>0.0166</td>
</tr>
<tr>
<td>close_files</td>
<td>0.3495</td>
<td>0.0000</td>
<td>0.1123</td>
<td>0.0833</td>
<td>0.0769</td>
<td>0.1208</td>
<td>0.1449</td>
<td>0.1123</td>
</tr>
</tbody>
</table>

*Table 71. Module VI Procedure to Procedure Coupling Matrix.*

The normalized coupling matrix: (coupling of functions i & j where i≠j) is

<table>
<thead>
<tr>
<th></th>
<th>main</th>
<th>initialize</th>
<th>open_files</th>
<th>read_doc</th>
<th>write_doc</th>
<th>do_stats</th>
<th>write_stats</th>
<th>close_files</th>
</tr>
</thead>
<tbody>
<tr>
<td>main</td>
<td>1.0000</td>
<td>0.0585</td>
<td>0.0981</td>
<td>0.2057</td>
<td>0.2029</td>
<td>0.4766</td>
<td>0.1179</td>
<td>0.0981</td>
</tr>
<tr>
<td>initialize</td>
<td>1.5000</td>
<td>1.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.5000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
</tr>
<tr>
<td>open_files</td>
<td>3.1129</td>
<td>0.0000</td>
<td>1.0000</td>
<td>0.7422</td>
<td>0.6851</td>
<td>1.0757</td>
<td>1.2909</td>
<td>1.0000</td>
</tr>
<tr>
<td>read doc</td>
<td>1.7016</td>
<td>0.0000</td>
<td>0.1935</td>
<td>1.0000</td>
<td>0.3952</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.1935</td>
</tr>
<tr>
<td>writedoc</td>
<td>1.5020</td>
<td>0.1443</td>
<td>0.1599</td>
<td>0.3536</td>
<td>1.0000</td>
<td>0.4210</td>
<td>0.5894</td>
<td>0.1599</td>
</tr>
<tr>
<td>do_stats</td>
<td>0.4588</td>
<td>0.0000</td>
<td>0.0326</td>
<td>0.0000</td>
<td>0.0547</td>
<td>1.0000</td>
<td>0.8561</td>
<td>0.0326</td>
</tr>
<tr>
<td>write_stats</td>
<td>0.0922</td>
<td>0.0000</td>
<td>0.0318</td>
<td>0.0000</td>
<td>0.0622</td>
<td>0.7025</td>
<td>1.0000</td>
<td>0.0318</td>
</tr>
<tr>
<td>close_files</td>
<td>3.1129</td>
<td>0.0000</td>
<td>1.0000</td>
<td>0.7422</td>
<td>0.6851</td>
<td>1.0757</td>
<td>1.2909</td>
<td>1.0000</td>
</tr>
</tbody>
</table>

*Table 72. Module VI Procedure to Procedure Normalized Coupling Matrix.*

The cumulative coupling of functions is:

<table>
<thead>
<tr>
<th>Function Name</th>
<th>main</th>
<th>initialize</th>
<th>open_files</th>
<th>read_doc</th>
<th>write_doc</th>
<th>do_stats</th>
<th>write_stats</th>
<th>close_files</th>
<th>System</th>
</tr>
</thead>
<tbody>
<tr>
<td>Coupling</td>
<td>6.369155</td>
<td>1.101415</td>
<td>4.711469</td>
<td>2.263840</td>
<td>2.957733</td>
<td>2.597697</td>
<td>2.537310</td>
<td>4.711459</td>
<td>3.406259</td>
</tr>
<tr>
<td>Coupling Inverse</td>
<td>0.157007</td>
<td>0.907923</td>
<td>0.212248</td>
<td>0.441727</td>
<td>0.338097</td>
<td>0.384936</td>
<td>0.394118</td>
<td>0.212248</td>
<td>0.293577</td>
</tr>
</tbody>
</table>

*Table 73. Module VI Procedures Overall Coupling.*
5.2.6.2 Dharma Method

<table>
<thead>
<tr>
<th>Function Name</th>
<th>main</th>
<th>initialize</th>
<th>open_files</th>
<th>read_doc</th>
<th>write_doc</th>
<th>do_stats</th>
<th>write_stats</th>
<th>close_files</th>
<th>System</th>
</tr>
</thead>
<tbody>
<tr>
<td>Coupling</td>
<td>26</td>
<td>5</td>
<td>7</td>
<td>8</td>
<td>19</td>
<td>7</td>
<td>16</td>
<td>7</td>
<td>10.5555</td>
</tr>
<tr>
<td>Coupling_Inverse</td>
<td>0.0384</td>
<td>0.2000</td>
<td>0.1428</td>
<td>0.1250</td>
<td>0.0526</td>
<td>0.1428</td>
<td>0.0625</td>
<td>0.1428</td>
<td>0.1133</td>
</tr>
</tbody>
</table>

*Table 74. Module VI Procedures Coupling Weights Using Dharma Method.*

The coupling of procedure to procedure is shown in the following table.

<table>
<thead>
<tr>
<th></th>
<th>main</th>
<th>initialize</th>
<th>open_files</th>
<th>read_doc</th>
<th>write_doc</th>
<th>do_stats</th>
<th>write_stats</th>
<th>close_files</th>
</tr>
</thead>
<tbody>
<tr>
<td>main</td>
<td>0.0384</td>
<td>0.1192</td>
<td>0.1812</td>
<td>0.0817</td>
<td>0.0455</td>
<td>0.1812</td>
<td>0.0505</td>
<td>0.1812</td>
</tr>
<tr>
<td>initialize</td>
<td>0.1192</td>
<td>0.2000</td>
<td>0.1740</td>
<td>0.1625</td>
<td>0.1263</td>
<td>0.1740</td>
<td>0.1313</td>
<td>0.1740</td>
</tr>
<tr>
<td>open_files</td>
<td>0.1812</td>
<td>0.1740</td>
<td>0.1428</td>
<td>0.1339</td>
<td>0.0977</td>
<td>0.1428</td>
<td>0.1027</td>
<td>0.1428</td>
</tr>
<tr>
<td>readdoc</td>
<td>0.0817</td>
<td>0.1625</td>
<td>0.1339</td>
<td>0.1250</td>
<td>0.0888</td>
<td>0.1339</td>
<td>0.0938</td>
<td>0.1339</td>
</tr>
<tr>
<td>writedoc</td>
<td>0.0455</td>
<td>0.1263</td>
<td>0.0977</td>
<td>0.0888</td>
<td>0.0526</td>
<td>0.0977</td>
<td>0.0576</td>
<td>0.0977</td>
</tr>
<tr>
<td>do_stats</td>
<td>0.1812</td>
<td>0.1740</td>
<td>0.1428</td>
<td>0.1339</td>
<td>0.0977</td>
<td>0.1428</td>
<td>0.1027</td>
<td>0.1428</td>
</tr>
<tr>
<td>write_stats</td>
<td>0.0505</td>
<td>0.1313</td>
<td>0.1027</td>
<td>0.0938</td>
<td>0.0576</td>
<td>0.1027</td>
<td>0.0625</td>
<td>0.1027</td>
</tr>
<tr>
<td>close_files</td>
<td>0.1812</td>
<td>0.1740</td>
<td>0.1428</td>
<td>0.1339</td>
<td>0.0977</td>
<td>0.1428</td>
<td>0.1027</td>
<td>0.1428</td>
</tr>
</tbody>
</table>

*Table 75. Module VI Procedure to Procedure Coupling Weights Using Dharma Method.*

5.2.6.3 Fenton and Melton Method

<table>
<thead>
<tr>
<th>Function Pair Coupled</th>
<th>main, initialize</th>
<th>main, open_files</th>
<th>main, read_doc</th>
<th>main, write_doc</th>
<th>main, do_stats</th>
<th>main, close_files</th>
<th>do_stats, write_stats</th>
<th>System</th>
</tr>
</thead>
</table>

*Table 76. Module VI Procedure to Procedure Coupling Weights Using Fenton and Melton Method.*
### 5.2.6.4 Offut, Harrold and Kotle Method

<table>
<thead>
<tr>
<th>Function Pair Coupled</th>
<th>main, initialize</th>
<th>main, open_files</th>
<th>main, read doc</th>
<th>main, write doc</th>
<th>main, do_stats</th>
<th>main, close_files</th>
<th>do_stats, write_stats</th>
<th>System</th>
</tr>
</thead>
<tbody>
<tr>
<td>Calling to Called Coupling Level</td>
<td>3.7500</td>
<td>3.7500</td>
<td>3.8333</td>
<td>3.6000</td>
<td>3.7000</td>
<td>3.7000</td>
<td>3.8333</td>
<td>3.7380</td>
</tr>
<tr>
<td>Called to Calling Coupling Level</td>
<td>3.7500</td>
<td>3.7500</td>
<td>3.8333</td>
<td>3.6000</td>
<td>3.7000</td>
<td>3.7000</td>
<td>3.8333</td>
<td>3.7380</td>
</tr>
</tbody>
</table>

*Table 77. Module VI Procedure to Procedure Coupling Weights Using Offut, Harrold and Kotle Method.*

### 5.2.6.5 Chen and Lu Method

<table>
<thead>
<tr>
<th>Function Name</th>
<th>main</th>
<th>initialize</th>
<th>open_files</th>
<th>read doc</th>
<th>write doc</th>
<th>do_stats</th>
<th>write stats</th>
<th>close_files</th>
<th>System</th>
</tr>
</thead>
<tbody>
<tr>
<td>Coupling</td>
<td>16</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>3.125</td>
</tr>
</tbody>
</table>

*Table 78. Module VI Procedures Coupling Values Using Chen and Lu Method.*

The procedure to procedure coupling values as in the following table.

<table>
<thead>
<tr>
<th>main initialize open_files read doc write doc do_stats write stats close_files</th>
<th>main</th>
<th>initialize</th>
<th>open_files</th>
<th>read doc</th>
<th>write doc</th>
<th>do_stats</th>
<th>write stats</th>
<th>close_files</th>
</tr>
</thead>
<tbody>
<tr>
<td>main initialize open_files read doc write doc do_stats write stats close_files</td>
<td>16.0000</td>
<td>8.5000</td>
<td>8.5000</td>
<td>8.5000</td>
<td>8.5000</td>
<td>9.5000</td>
<td>8.5000</td>
<td>8.5000</td>
</tr>
<tr>
<td>initialize open_files read doc write doc do_stats write stats close_files</td>
<td>8.5000</td>
<td>1.0000</td>
<td>1.0000</td>
<td>1.0000</td>
<td>1.0000</td>
<td>2.0000</td>
<td>1.0000</td>
<td>1.0000</td>
</tr>
<tr>
<td>open_files read doc write doc do_stats write stats close_files</td>
<td>8.5000</td>
<td>1.0000</td>
<td>1.0000</td>
<td>1.0000</td>
<td>1.0000</td>
<td>2.0000</td>
<td>1.0000</td>
<td>1.0000</td>
</tr>
<tr>
<td>read doc write doc do_stats write stats close_files</td>
<td>8.5000</td>
<td>1.0000</td>
<td>1.0000</td>
<td>1.0000</td>
<td>1.0000</td>
<td>2.0000</td>
<td>1.0000</td>
<td>1.0000</td>
</tr>
<tr>
<td>do_stats write stats close_files</td>
<td>9.5000</td>
<td>2.0000</td>
<td>2.0000</td>
<td>2.0000</td>
<td>2.0000</td>
<td>3.0000</td>
<td>2.0000</td>
<td>2.0000</td>
</tr>
<tr>
<td>write_stats close_files</td>
<td>8.5000</td>
<td>1.0000</td>
<td>1.0000</td>
<td>1.0000</td>
<td>1.0000</td>
<td>2.0000</td>
<td>1.0000</td>
<td>1.0000</td>
</tr>
</tbody>
</table>

*Table 79. Module VI Procedure by Procedure Coupling Values Using Chen and Lu Method.*
5.2.6.6 Analysis

The program source code shows that main is the driver of computation. So, you would expect that main is attached more with all except writestat with a higher portion toward do_stats. Also, you would expect that do_stats is attached more with write_stats than others. Finally, you would expect that most of these procedures are related to each others because they used global variables to communicate.

The new method preserves this because the coupling value of main is concentrated on all others. Main procedure is attached more with itself than do_stats, readdoc, writedoc, write_stats, open_files, close_files, and initialize procedures respectively. The new method shows that do_stats is attached more with itself than write_stats than the remaining. It also, shows that main procedure is the main routine for computation. The new method shows the coupling of 12 pairs of routines to be zero because they do not have any parameters shared with each others.

Dhama measure preserves this relation because its main is attached more with itself, write_doc, write_stats, read_doc, initialize, do_stats, close_files and open_files respectively. Also, it assigns a coupling value for do_stats and write_doc which is greater than others. Moreover, it shows that all procedures are related to each others.

Fenton and Melton measure preserves the relation because main is coupled more with readdoc, initialize, open_files, do_stats, close_files, and writedoc. Also, do_stats is coupled maximally to write_stats.
Offut, Harrold and Kotle measure preserves the relation by the same reasoning cited on Fenton and Melton measure.

Chen and Lu method does not preserve the above relation because although it shows that main is attached heavily to all routines, it assigns do_stats and writestats a lower value.

5.2.7 Analysis Summary

The new method seems to capture all relations among all modules. Dhama preserves 3 relations in three modules. Fenton and Melton measure and Offut, Harrold and Kotle preserve 5 relations for five modules. The main reason for that is both measures are limited to calling and called pairs with bi-directional pairs in Offut, Harrold and Kotle measure. Chen and Lu measure captures 3 relations for three modules. Table 80 summarize the results.

<table>
<thead>
<tr>
<th></th>
<th>Module I</th>
<th>Module II</th>
<th>Module III</th>
<th>Module IV</th>
<th>Module V</th>
<th>Module VI</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>New</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>6</td>
</tr>
<tr>
<td>Dhama</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>Fenton and Melton</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>5</td>
</tr>
<tr>
<td>Offut, Harrold and Kotle</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>5</td>
</tr>
<tr>
<td>Chen and Lu</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>3</td>
</tr>
</tbody>
</table>

*Table 80. Comparisons of New and Recent Measures on Capturing Relation on Modules.*

In addition, the new method satisfies the following :

1. It gives bi-directional coupling not necessarily symmetric as in Dhama and Chen and Lu measures.
2. It does not limit that every procedure is coupled with every others like in Dhama and Chen and Lu
because some pairs may be independent.

3. It helps to identify the main computation routine that has the column with high proportion among each row.

4. It is general and generates the coupling of module pairs. The coupling can be calculated at module, sub-system and system levels.

The new method outperforms the other methods because:

1. It captures more factors than all the others.

2. It incorporates all coupling types a variable possess.

3. Its calculation is based on a general model that measures the interactions between system components.

5.3 Effect of Change in Coupling Values

A good measure should be sensitive to atomic changes [Zuse90]. Six relevant change operations have been identified with the expected change in coupling values. These atomic modifications are applied on the corresponding modules. The results from applying the five methods on the modified modules are obtained. These values are tested and compared with the values obtained from the original modules for each measure on each module. Finally, the results across modules are summarized. The following changes will be done to the modules to see how the methods are affected in correspondence with the change:

1. Remove a procedure from Module II: You expect this to decrease the coupling between all
procedures and hence decrease system coupling because in general lowering the interactions between procedures usually will decrease the coupling between them.

2. Combine AVCALC and SUMCAL to a new AVCALC in Module III: The coupling between all procedures should decrease and so decrease the system coupling because decreasing the interactions between procedures usually decreases the coupling among them.

3. Eliminate one LEAP and use the other LEAP in Module IV: You would expect that the coupling of existing LEAP and DAYS procedures to increase because their weights have increased. On the other hand, the other procedures coupling should decrease or stay the same because their proportion is lowered when LEAP and DAYS procedures weights have increased. However, the system coupling should increase because the net result is adding more weight to the system.

4. Increase the value of an element in a procedure in Module V: You would expect that the coupling of CRAPS procedure that includes the element to increase while the coupling of at least one procedure that use the element should decrease and the system coupling should increase because as the element weight increases, its proportion to the coupling also increases.

5. Reduce the value of an element in a procedure in Module VI: You would expect that the coupling of write_stats procedure that includes the element to decrease while the coupling of at least one procedure that uses the element should increase and the system coupling decreases because as the element weight decreases, its proportion to the coupling also decreases.

6. Make Module VI procedures communicate through parameter passing instead of globals: You would expect the coupling of procedures reduces or at least remains the same and the overall coupling decreases since the weight for the elements should decrease.
5.3.1 Remove a procedure from Module II

In this impact analysis, main procedure which was added to really compare between Dhama 's two Implementation will be removed and see how this change is affecting the different methods.

5.3.1.1 New Method

The definition matrix is

<table>
<thead>
<tr>
<th>n</th>
<th>to_be_sorted</th>
<th>start</th>
<th>location</th>
</tr>
</thead>
<tbody>
<tr>
<td>sort2</td>
<td>9</td>
<td>12</td>
<td>3</td>
</tr>
<tr>
<td>find_max</td>
<td>3</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>find_min</td>
<td>3</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>exchange</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

*Table 81. Module II Definition Matrix After Removal of Main Procedure.*

The coupling matrix is

<table>
<thead>
<tr>
<th></th>
<th>sort2</th>
<th>find_max</th>
<th>find_min</th>
<th>exchange</th>
</tr>
</thead>
<tbody>
<tr>
<td>sort2</td>
<td>0.5918</td>
<td>0.1739</td>
<td>0.1739</td>
<td>0.0604</td>
</tr>
<tr>
<td>find_max</td>
<td>0.5868</td>
<td>0.1759</td>
<td>0.1759</td>
<td>0.0614</td>
</tr>
<tr>
<td>find_min</td>
<td>0.5868</td>
<td>0.1759</td>
<td>0.1759</td>
<td>0.0614</td>
</tr>
<tr>
<td>exchange</td>
<td>0.5439</td>
<td>0.1637</td>
<td>0.1637</td>
<td>0.1287</td>
</tr>
</tbody>
</table>

*Table 82. Module II Procedure to Procedure Coupling Matrix after Removal of Main Procedure.*

The normalized coupling matrix: (coupling of functions i & j where $\neq j$) is
<table>
<thead>
<tr>
<th>Function Name</th>
<th>sort2</th>
<th>find_max</th>
<th>find_min</th>
<th>exchange</th>
</tr>
</thead>
<tbody>
<tr>
<td>Coupling</td>
<td>5.795156</td>
<td>3.626160</td>
<td>3.626160</td>
<td>3.786545</td>
</tr>
<tr>
<td>Coupling Inverse</td>
<td>0.172558</td>
<td>0.275774</td>
<td>0.275774</td>
<td>0.264093</td>
</tr>
</tbody>
</table>

*Table 84. Module II Procedures Overall Coupling After Removal of Main Procedure.*

### 5.3.1.2 Dhama Method

<table>
<thead>
<tr>
<th>Function Name</th>
<th>sort2</th>
<th>find_max</th>
<th>find_min</th>
<th>exchange</th>
<th>System</th>
</tr>
</thead>
<tbody>
<tr>
<td>Coupling (Increasing)</td>
<td>8</td>
<td>7</td>
<td>7</td>
<td>5</td>
<td>6.75</td>
</tr>
<tr>
<td>Coupling (Decreasing)</td>
<td>0.1300</td>
<td>0.1400</td>
<td>0.1400</td>
<td>0.2000</td>
<td>0.1525</td>
</tr>
</tbody>
</table>

*Table 85. Module II Procedures Coupling Weights Using Dhama Method.*

The coupling of procedure to procedure is shown in the following table.

<table>
<thead>
<tr>
<th>Function Name</th>
<th>sort2</th>
<th>find_max</th>
<th>find_min</th>
<th>exchange</th>
</tr>
</thead>
<tbody>
<tr>
<td>sort2</td>
<td>0.1300</td>
<td>0.1350</td>
<td>0.1350</td>
<td>0.1650</td>
</tr>
<tr>
<td>find_max</td>
<td>0.1350</td>
<td>0.1400</td>
<td>0.1400</td>
<td>0.1700</td>
</tr>
<tr>
<td>find_min</td>
<td>0.1350</td>
<td>0.1400</td>
<td>0.1400</td>
<td>0.1700</td>
</tr>
<tr>
<td>exchange</td>
<td>0.1650</td>
<td>0.1700</td>
<td>0.1700</td>
<td>0.2000</td>
</tr>
</tbody>
</table>

*Table 86. Module II Procedure to Procedure Coupling Weight Using Dhama Method After Removal of Main Procedure.*
5.3.1.3 Fenton and Melton Method

<table>
<thead>
<tr>
<th>Function Name</th>
<th>sort2, find_max</th>
<th>sort2, find_min</th>
<th>sort2, exchange</th>
<th>System</th>
</tr>
</thead>
<tbody>
<tr>
<td>Coupling Level</td>
<td>3.200</td>
<td>3.200</td>
<td>3.333</td>
<td>3.200</td>
</tr>
</tbody>
</table>

*Table 87. Module II Procedure to Procedure Coupling Weight Using Fenton and Melton Method After Removal of Main Procedure.*

5.3.1.4 Offut, Harrold and Kotle Method

<table>
<thead>
<tr>
<th>Function Pair Coupled</th>
<th>sort2, find_max</th>
<th>sort2, find_min</th>
<th>sort2, exchange</th>
<th>System</th>
</tr>
</thead>
<tbody>
<tr>
<td>Calling to Called Coupling</td>
<td>2.7000</td>
<td>2.7000</td>
<td>2.833</td>
<td>2.4111</td>
</tr>
<tr>
<td>Called to Calling Coupling</td>
<td>2.7000</td>
<td>2.7000</td>
<td>0.833</td>
<td></td>
</tr>
</tbody>
</table>

*Table 88. Module II Procedure to Procedure Coupling Weight Using Offut, Harrold and Kotle Method.*

5.3.1.5 Chen and Lu Method

<table>
<thead>
<tr>
<th>Function Name</th>
<th>sort2</th>
<th>find_max</th>
<th>find_min</th>
<th>exchange</th>
<th>System</th>
</tr>
</thead>
<tbody>
<tr>
<td>Coupling</td>
<td>5</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>2</td>
</tr>
</tbody>
</table>

*Table 89. Module II Procedures Coupling Weight Using Chen and Lu Method After Removal of Main Procedure.*

The procedure to procedure coupling values are shown in Table 90.

<table>
<thead>
<tr>
<th></th>
<th>sort2</th>
<th>find_max</th>
<th>find_min</th>
<th>exchange</th>
</tr>
</thead>
<tbody>
<tr>
<td>sort2</td>
<td>5.0000</td>
<td>3.0000</td>
<td>3.0000</td>
<td>3.0000</td>
</tr>
<tr>
<td>find_max</td>
<td>3.0000</td>
<td>1.0000</td>
<td>1.0000</td>
<td>1.0000</td>
</tr>
<tr>
<td>find_min</td>
<td>3.0000</td>
<td>1.0000</td>
<td>1.0000</td>
<td>1.0000</td>
</tr>
<tr>
<td>exchange</td>
<td>3.0000</td>
<td>1.0000</td>
<td>1.0000</td>
<td>1.0000</td>
</tr>
</tbody>
</table>

*Table 90. Module II Procedure to Procedure Coupling Weight Using Chen and Lu Method After Removal of Main Procedure.*
5.3.1.6 Analysis

The effect of removing main on the new method starts from the definition matrix. The row containing
the main procedure is deleted. The new method preserves the relation because when main procedure is
deleted, the proportion of all element values have decreased. So, the total coupling of each procedure is
decreased and correspondingly the system coupling. Therefore, the new method captures the empirical
relation.

Dharma measure does not capture the relation because when the main procedure is deleted, the system
coupling is only decreased. Fenton and Melton measure also does not capture the relation because
when main to sort2 coupling is deleted the system coupling has only decreased. Offut, Harroid and
Kotle measure does not capture the relation because when the main to sort2 and sort2 to main coupling
is deleted only the system coupling is increased. Chen and Lu measure does not satisfy the relation
because the coupling of sort2 and the overall system have decreased.

5.3.2 Combine Two Procedures in Module III

In this impact analysis, the logic of SUMCAL procedure will be included in AVCALC procedure in
Module III to see how this change is affecting the different methods. As follows:

REAL FUNCTION AVCALC (A, N, NDIM)
INTEGER N, NDIM, I
REAL SUM, A(NDIM)
SUM = 0
DO 10 I = 1, N
    SUM = SUM + A(I)
10 CONTINUE
AVCALC = SUM / N
RETURN
END

5.3.2.1 New Method

The definition matrix is

|       | P | R | O | M | E | L | N | L | O | W | B | N | D | A | O | V | C | A | L | C | X | N | D | I | M | G | T | A | V | G | A | V | G |
| MAIN  | 1 | 9 | 1 | 1 | 1 | 1 | 2 | 2 | 1 | 1 |    |    |    |    |    |    |    |    |    |    |    |    |    |    |
| RCHECK| 1 | 3 | 3 | 3 | 0 | 0 | 0 | 0 | 0 | 0 |    |    |    |    |    |    |    |    |    |    |    |    |    |    |
| AVCALC| 0 | 3 | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 0 |    |    |    |    |    |    |    |    |    |    |    |    |    |    |
| GTAVG | 0 | 1 | 0 | 0 | 3 | 0 | 1 | 3 |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |

Table 91. Module III Definition Matrix After Combining AVCALC and SUMCAL into new AVCALC Procedure.

The coupling matrix is

<table>
<thead>
<tr>
<th></th>
<th>MAIN</th>
<th>RCHECK</th>
<th>AVCALC</th>
<th>GTAVG</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAIN</td>
<td>0.4901</td>
<td>0.1941</td>
<td>0.1678</td>
<td>0.1480</td>
</tr>
<tr>
<td>RCHECK</td>
<td>0.3687</td>
<td>0.5563</td>
<td>0.0563</td>
<td>0.0188</td>
</tr>
<tr>
<td>AVCALC</td>
<td>0.5312</td>
<td>0.0938</td>
<td>0.2604</td>
<td>0.1146</td>
</tr>
<tr>
<td>GTAVG</td>
<td>0.3516</td>
<td>0.0234</td>
<td>0.0859</td>
<td>0.5391</td>
</tr>
</tbody>
</table>

Table 92. Module III Procedure to Procedure Coupling Matrix After Combining AVCALC and SUMCAL into new AVCALC Procedure.
The normalized coupling matrix: (coupling of functions $i$ & $j$ where $i \neq j$) is

<table>
<thead>
<tr>
<th></th>
<th>MAIN</th>
<th>RCHECK</th>
<th>AVCALC</th>
<th>GTAVG</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAIN</td>
<td>1.0000</td>
<td>0.3960</td>
<td>0.3423</td>
<td>0.3020</td>
</tr>
<tr>
<td>RCHECK</td>
<td>0.6629</td>
<td>1.0000</td>
<td>0.1011</td>
<td>0.0337</td>
</tr>
<tr>
<td>AVCALC</td>
<td>2.0400</td>
<td>0.3600</td>
<td>1.0000</td>
<td>0.4400</td>
</tr>
<tr>
<td>GTAVG</td>
<td>0.6522</td>
<td>0.0435</td>
<td>0.1594</td>
<td>1.0000</td>
</tr>
</tbody>
</table>

Table 93. Module III Procedure to Procedure Normalized Coupling Matrix After Combining AVCALC and SUMCAL into new AVCALC Procedure.

The cumulative coupling of functions is:

<table>
<thead>
<tr>
<th>Function Name</th>
<th>MAIN</th>
<th>RCHECK</th>
<th>AVCALC</th>
<th>GTAVG</th>
<th>System</th>
</tr>
</thead>
<tbody>
<tr>
<td>Coupling</td>
<td>2.197682</td>
<td>0.798602</td>
<td>1.721413</td>
<td>0.815397</td>
<td>1.383273</td>
</tr>
<tr>
<td>Coupling Inverse</td>
<td>0.455025</td>
<td>1.252188</td>
<td>0.580918</td>
<td>1.226397</td>
<td>0.722923</td>
</tr>
</tbody>
</table>

Table 94. Module III Procedures Overall Coupling After Combining AVCALC and SUMCAL into new AVCALC Procedure.

5.3.2.2 Dhamma Method

<table>
<thead>
<tr>
<th>Function Name</th>
<th>MAIN</th>
<th>RCHECK</th>
<th>AVCALC</th>
<th>GTAVG</th>
<th>System</th>
</tr>
</thead>
<tbody>
<tr>
<td>Coupling (Increasing)</td>
<td>9</td>
<td>8</td>
<td>5</td>
<td>5</td>
<td>6.75</td>
</tr>
<tr>
<td>Coupling (Decreasing)</td>
<td>0.1111</td>
<td>0.1250</td>
<td>0.2000</td>
<td>0.2000</td>
<td>0.1590</td>
</tr>
</tbody>
</table>

Table 95. Module III Procedures Coupling Weights Using Dhamma Method After Combining AVCALC and SUMCAL into new AVCALC Procedure.

The coupling of procedure to procedure is shown in the following table.

<table>
<thead>
<tr>
<th></th>
<th>MAIN</th>
<th>RCHECK</th>
<th>AVCALC</th>
<th>GTAVG</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAIN</td>
<td>0.1111</td>
<td>0.1181</td>
<td>0.1556</td>
<td>0.1556</td>
</tr>
<tr>
<td>RCHECK</td>
<td>0.1181</td>
<td>0.1250</td>
<td>0.1625</td>
<td>0.1625</td>
</tr>
<tr>
<td>AVCALC</td>
<td>0.1556</td>
<td>0.1625</td>
<td>0.2000</td>
<td>0.2000</td>
</tr>
<tr>
<td>GTAVG</td>
<td>0.1556</td>
<td>0.1625</td>
<td>0.2000</td>
<td>0.2000</td>
</tr>
</tbody>
</table>

Table 96. Module III Procedure to Procedure Coupling Weights Using Dhamma Method After Combining AVCALC and SUMCAL into new AVCALC Procedure.
5.3.2.3  Fenton and Melton Method

<table>
<thead>
<tr>
<th>Function Pair Coupled</th>
<th>MAIN, RCHECK</th>
<th>MAIN, AVCALC</th>
<th>MAIN, GTAVG</th>
<th>System</th>
</tr>
</thead>
</table>

*Table 97. Module III Procedure to Procedure Coupling Weights Using Fenton and Melton Method After Combining AVCALC and SUMCAL into new AVCALC Procedure.*

5.3.2.4  Offut, Harrold and Kotle Method

<table>
<thead>
<tr>
<th>Function Pair Coupled</th>
<th>MAIN, RCHECK</th>
<th>MAIN, AVCALC</th>
<th>MAIN, GTAVG</th>
<th>System</th>
</tr>
</thead>
<tbody>
<tr>
<td>Calling to Called Coupling Level</td>
<td>2.7000</td>
<td>0.7000</td>
<td>2.7000</td>
<td>2.0333</td>
</tr>
<tr>
<td>Called to Calling Coupling Level</td>
<td>2.7000</td>
<td>0.7000</td>
<td>2.7000</td>
<td></td>
</tr>
</tbody>
</table>

*Table 98. Module III Procedure to Procedure Coupling Weights Using Offut, Harrold and Kotle Method After Combining AVCALC and SUMCAL into new AVCALC Procedure.*

5.3.2.5  Chen and Lu Method

<table>
<thead>
<tr>
<th>Function Name</th>
<th>MAIN</th>
<th>RCHECK</th>
<th>AVCALC</th>
<th>GTAVG</th>
<th>System</th>
</tr>
</thead>
<tbody>
<tr>
<td>Coupling</td>
<td>7</td>
<td>1</td>
<td>3</td>
<td>1</td>
<td>3</td>
</tr>
</tbody>
</table>

*Table 99. Module III Procedures Coupling Weights Using Chen and Lu Method After Combining AVCALC and SUMCAL into new AVCALC Procedure.*
The procedure to procedure coupling values are shown in Table 100.

<table>
<thead>
<tr>
<th></th>
<th>MAIN</th>
<th>RCHECK</th>
<th>AVCALC</th>
<th>GTAVG</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAIN</td>
<td>7.0000</td>
<td>4.0000</td>
<td>5.0000</td>
<td>4.0000</td>
</tr>
<tr>
<td>RCHECK</td>
<td>4.0000</td>
<td>1.0000</td>
<td>2.0000</td>
<td>1.0000</td>
</tr>
<tr>
<td>AVCALC</td>
<td>5.0000</td>
<td>2.0000</td>
<td>3.0000</td>
<td>2.0000</td>
</tr>
<tr>
<td>GTAVG</td>
<td>4.0000</td>
<td>1.0000</td>
<td>2.0000</td>
<td>1.0000</td>
</tr>
</tbody>
</table>

*Table 100. Module III Procedure to Procedure Coupling Weight Using Chen and Lu Method After Combining AVCALC and SUMCAL into new AVCALC Procedure.*

5.3.2.6 Analysis

The effect of combining AVCALC and SUMCAL into a new AVCALC procedure starts from the definition matrix. The row and column containing the SUMCAL function are deleted. Also, NUMEL variable weight increases from 2 to 3 while X and NDIM variables weights decrease from 2 to 1. The new method preserves the relation because all procedures and system coupling has decreased.

Dharma measure does not capture the relation because when the SUMCAL procedure is deleted the system coupling has increased. Fenton and Melton measure does recognize the relation because when the AVCALC to SUMCAL coupling is deleted the system coupling has increased. Offut, Harrold and Kotle measure does not satisfy the relation because when the AVCALC to SUMCAL and SUMCAL to AVCALC coupling is deleted and the system coupling, the coupling of MAIN to AVCALC and the coupling of AVCALC to MAIN have increased. Chen and Lu measure does not agree with the relation because the coupling of AVCALC and the overall system coupling have increased.
5.3.3 Replace LEAP in Module IV

In this impact analysis, LEAP procedure that was included with VALDAY procedure is eliminated and VALDAY procedure will use the LEAP procedure included with DAYS procedure by making it external routine in Module III to see how this change is affecting the different methods.

5.3.3.1 New Method

The definition matrix is

<table>
<thead>
<tr>
<th></th>
<th>S D A T E</th>
<th>E D A T E</th>
<th>N D A Y</th>
<th>V A L D A Y</th>
<th>L E A P</th>
<th>Y E S</th>
<th>I E S</th>
<th>Y E S</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAIN</td>
<td>1 1 1 0 0 0 0 0 0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DAYS</td>
<td>6 6 1 2 21 4 1 1 0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VALDAY</td>
<td>3 3 0 2 0 0 0 0 1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LEAP</td>
<td>0 0 0 7 12 3 3 3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

*Table 101. Module IV Definition Matrix After Removing One Leap Procedure and Let VALDAY Procedure Use the Other LEAP Procedure.*

The coupling matrix is

<table>
<thead>
<tr>
<th></th>
<th>MAIN</th>
<th>DAYS</th>
<th>VALDAY</th>
<th>LEAP</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAIN</td>
<td>0.2333</td>
<td>0.5667</td>
<td>0.2000</td>
<td>0.0000</td>
</tr>
<tr>
<td>DAYS</td>
<td>0.0405</td>
<td>0.6179</td>
<td>0.1095</td>
<td>0.2321</td>
</tr>
<tr>
<td>VALDAY</td>
<td>0.0667</td>
<td>0.5111</td>
<td>0.3389</td>
<td>0.0833</td>
</tr>
<tr>
<td>LEAP</td>
<td>0.0000</td>
<td>0.3482</td>
<td>0.0268</td>
<td>0.6250</td>
</tr>
</tbody>
</table>

*Table 102. Module IV Procedure to Procedure Coupling Matrix After Removing One Leap Procedure and Let VALDAY Procedure Use the Other LEAP Procedure.*
The normalized coupling matrix: (coupling of functions i & j where $i \neq j$) is

<table>
<thead>
<tr>
<th></th>
<th>MAIN</th>
<th>DAYS</th>
<th>VALDAY</th>
<th>LEAP</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAIN</td>
<td>1.0000</td>
<td>2.4286</td>
<td>0.8571</td>
<td>0.0000</td>
</tr>
<tr>
<td>DAYS</td>
<td>0.0655</td>
<td>1.0000</td>
<td>0.1773</td>
<td>0.3757</td>
</tr>
<tr>
<td>VALDAY</td>
<td>0.1967</td>
<td>1.5082</td>
<td>1.0000</td>
<td>0.2459</td>
</tr>
<tr>
<td>LEAP</td>
<td>0.0000</td>
<td>0.5571</td>
<td>0.0429</td>
<td>1.0000</td>
</tr>
</tbody>
</table>

*Table 103. Module IV Procedure to Procedure Normalized Coupling Matrix After Removing One Leap Procedure and Let VALDAY Procedure Use the Other LEAP Procedure.*

The cumulative coupling of functions is:

<table>
<thead>
<tr>
<th>Function Name</th>
<th>MAIN</th>
<th>DAYS</th>
<th>VALDAY</th>
<th>LEAP</th>
<th>System</th>
</tr>
</thead>
<tbody>
<tr>
<td>Coupling</td>
<td>1.773973</td>
<td>2.556204</td>
<td>1.514042</td>
<td>0.610812</td>
<td>1.613758</td>
</tr>
<tr>
<td>Coupling Inverse</td>
<td>0.563706</td>
<td>0.391205</td>
<td>0.660484</td>
<td>1.637165</td>
<td>0.619672</td>
</tr>
</tbody>
</table>

*Table 104. Module IV Procedures Overall Coupling After Removing One Leap Procedure and Let VALDAY Procedure Use the Other LEAP Procedure.*

### 5.3.3.2 Dhamma Method

<table>
<thead>
<tr>
<th>Function Name</th>
<th>MAIN</th>
<th>DAYS</th>
<th>VALDAY</th>
<th>LEAP</th>
<th>System</th>
</tr>
</thead>
<tbody>
<tr>
<td>Coupling</td>
<td>7</td>
<td>7</td>
<td>5</td>
<td>10</td>
<td>7.2500</td>
</tr>
<tr>
<td>Coupling Inverse</td>
<td>0.1428</td>
<td>0.1428</td>
<td>0.2000</td>
<td>0.1000</td>
<td>0.1464</td>
</tr>
</tbody>
</table>

*Table 105. Module IV Procedures Coupling Weights Using Dhamma Method After Removing One Leap Procedure and Let VALDAY Procedure Use the Other LEAP Procedure.*

The coupling of procedure to procedure is shown in the following table.

<table>
<thead>
<tr>
<th></th>
<th>MAIN</th>
<th>DAYS</th>
<th>VALDAY</th>
<th>LEAP</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAIN</td>
<td>0.1428</td>
<td>0.1428</td>
<td>0.1714</td>
<td>0.1214</td>
</tr>
<tr>
<td>DAYS</td>
<td>0.1428</td>
<td>0.1428</td>
<td>0.1714</td>
<td>0.1214</td>
</tr>
<tr>
<td>VALDAY</td>
<td>0.1714</td>
<td>0.1714</td>
<td>0.2000</td>
<td>0.1500</td>
</tr>
<tr>
<td>LEAP</td>
<td>0.1214</td>
<td>0.1214</td>
<td>0.1500</td>
<td>0.1000</td>
</tr>
</tbody>
</table>

*Table 106. Module IV Procedure to Procedure Coupling Weights Using Dhamma Method After Removing One Leap Procedure and Let VALDAY Procedure Use the Other LEAP Procedure.*
5.3.3.3 Fenton and Melton Method

<table>
<thead>
<tr>
<th>Function Pair Coupled</th>
<th>MAIN, DAYS</th>
<th>DAYS, VALDAY</th>
<th>DAYS, LEAP</th>
<th>VALDAY, LEAP</th>
<th>System</th>
</tr>
</thead>
<tbody>
<tr>
<td>Coupling Level</td>
<td>3.2500</td>
<td>3.3333</td>
<td>3.3333</td>
<td>3.3333</td>
<td>3.3333</td>
</tr>
</tbody>
</table>

Table 107. Module IV Procedure to Procedure Coupling Weights Using Fenton and Melton Method After Removing One Leap Procedure and Let VALDAY Procedure Use the Other LEAP Procedure.

5.3.3.4 Offut, Harrold and Kotle Method

<table>
<thead>
<tr>
<th>Function Pair Coupled</th>
<th>MAIN, DAYS</th>
<th>DAYS, VALDAY</th>
<th>DAYS, LEAP</th>
<th>VALDAY, LEAP</th>
<th>System</th>
</tr>
</thead>
<tbody>
<tr>
<td>Calling to Called Coupling Level</td>
<td>2.7500</td>
<td>2.8333</td>
<td>2.8333</td>
<td>2.8333</td>
<td>2.0624</td>
</tr>
<tr>
<td>Calling to Called Coupling Level</td>
<td>0.7500</td>
<td>2.8333</td>
<td>0.8333</td>
<td>0.8333</td>
<td></td>
</tr>
</tbody>
</table>

Table 108. Module IV Procedure to Procedure Coupling Weights Using Offut, Harrold and Kotle Method After Removing One Leap Procedure and Let VALDAY Procedure Use the Other LEAP Procedure.

5.3.3.5 Chen and Lu Method

<table>
<thead>
<tr>
<th>Function Name</th>
<th>MAIN</th>
<th>DAYS</th>
<th>VALDAY</th>
<th>LEAP</th>
<th>System</th>
</tr>
</thead>
<tbody>
<tr>
<td>Coupling</td>
<td>6</td>
<td>6</td>
<td>1</td>
<td>4</td>
<td>4.25</td>
</tr>
</tbody>
</table>

Table 109. Module IV Procedures Coupling Weights Using Chen and Lu Method After Removing One Leap Procedure and Let VALDAY Procedure Use the Other LEAP Procedure.

The procedure to procedure coupling values are shown in Table 110.
<table>
<thead>
<tr>
<th></th>
<th>MAIN</th>
<th>DAYS</th>
<th>VALDAY</th>
<th>LEAP</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAIN</td>
<td>6.0000</td>
<td>6.0000</td>
<td>3.5000</td>
<td>5.0000</td>
</tr>
<tr>
<td>DAYS</td>
<td>6.0000</td>
<td>6.0000</td>
<td>3.5000</td>
<td>5.0000</td>
</tr>
<tr>
<td>VALDAY</td>
<td>3.5000</td>
<td>3.5000</td>
<td>1.0000</td>
<td>2.5000</td>
</tr>
<tr>
<td>LEAP</td>
<td>5.0000</td>
<td>5.0000</td>
<td>2.5000</td>
<td>4.0000</td>
</tr>
</tbody>
</table>

*Table 110. Module IV Procedure to Procedure Coupling Weight Using Chen and Lu Method After Removing One Leap Procedure and Let VALDAY Procedure Use the Other LEAP Procedure.*

### 5.3.3.6 Analysis

The effect of removing one LEAP and using the other starts from the definition matrix. The row and the column containing the LEAP function are deleted. Also, the weight of YS variable against LEAP has increased from 0 to 3. Moreover, the weight of LEAP variable against LEAP procedure has increased from 6 to 7. Finally, the weight of LEAP variable against DAYS procedure has increased from 18 to 21. The new method satisfies the relation because the coupling of the other LEAP and DAYS procedures has increased and system coupling has increased. The coupling of VALDAY procedure has decreased because it loses the weight of its LEAP procedure. The coupling of MAIN procedure increased slightly because MAIN procedure does not use the variables modified by the change operation.

Dhama measure does not preserve the relation because when the LEAP procedure is deleted, MAIN and DAYS procedures coupling have increased. Moreover, the coupling of LEAP procedure and system coupling has decreased while the coupling of VALDAY procedure remains the same. Fenton and Melton measure, Offut, Harrold and Kotle measure and Chen and Lu measure do not satisfy the relation because no coupling is changed in any of their procedures or system.
5.3.4 Increase Element Weight in Module V

Suppose that the player wants to know before each game how many times he played so far and how many of these he won. This can be achieved in the following code fragment that needs to be included in CRAPS procedure.

\[
\begin{align*}
\text{INTEGER NOPLAY, NOWON;}
\end{align*}
\]

\[
\begin{align*}
\text{NOPLAY} & := \text{NOPLAY} + 1; \\
\text{IF} \ \text{WON} \ \text{THEN} \\
& \quad \text{NOWON} := \text{NOWON} + 1; \\
\text{END IF}
\end{align*}
\]

Now we will see the effect of this change on the methods.

5.3.4.1 New Method

The definition matrix is

\[
\begin{array}{cccccccc}
\text{SEED} & \text{AMOUNT} & \text{BET} & \text{WON} & \text{FINISHED} & \text{VALUE} & \text{RANDOM} & \text{N} \\
\hline
\text{CRAPS} & 5 & 3 & 2 & 6 & 3 & 0 & 0 & 0 \\
\text{INITIALIZE} & 1 & 3 & 0 & 0 & 0 & 0 & 0 & 0 \\
\text{ACCEPTBET} & 0 & 3 & 3 & 0 & 0 & 0 & 0 & 0 \\
\text{PLAY} & 0 & 0 & 0 & 3 & 0 & 6 & 0 & 0 \\
\text{ROLLDICE} & 0 & 0 & 0 & 0 & 0 & 2 & 4 & 4 \\
\text{RANDOM} & 20 & 0 & 0 & 0 & 0 & 0 & 4 & 4 \\
\text{UPDATE} & 0 & 3 & 1 & 3 & 3 & 0 & 0 & 0 \\
\end{array}
\]

Table 111. Module V Description Matrix After Increasing the weight of WON in Main Procedure From 3 to 6.
The coupling matrix is

<table>
<thead>
<tr>
<th></th>
<th>CRAPS</th>
<th>INITIALIZE</th>
<th>ACCEPT BET</th>
<th>PLAY</th>
<th>ROLL DICE</th>
<th>RANDOM</th>
<th>UP DATE</th>
</tr>
</thead>
<tbody>
<tr>
<td>CRAPS</td>
<td>0.3620</td>
<td>0.0496</td>
<td>0.0921</td>
<td>0.0789</td>
<td>0.0000</td>
<td>0.2024</td>
<td>0.2149</td>
</tr>
<tr>
<td>INITIALIZE</td>
<td>0.2356</td>
<td>0.1971</td>
<td>0.1875</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.1923</td>
<td>0.1875</td>
</tr>
<tr>
<td>ACCEPTBET</td>
<td>0.2917</td>
<td>0.1250</td>
<td>0.3750</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.2083</td>
</tr>
<tr>
<td>PLAY</td>
<td>0.1667</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.5833</td>
<td>0.1667</td>
<td>0.0000</td>
<td>0.0833</td>
</tr>
<tr>
<td>ROLLDICE</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.1500</td>
<td>0.4500</td>
<td>0.4000</td>
<td>0.0000</td>
</tr>
<tr>
<td>RANDOM</td>
<td>0.1374</td>
<td>0.0275</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.1429</td>
<td>0.6923</td>
<td>0.0000</td>
</tr>
<tr>
<td>UPDATE</td>
<td>0.4083</td>
<td>0.0750</td>
<td>0.1250</td>
<td>0.0750</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.3167</td>
</tr>
</tbody>
</table>

Table 112. Module V Procedure to Procedure Coupling Matrix After Increasing the weight of WON in Main Procedure From 3 to 6.

The normalized coupling matrix: (coupling of functions i & j where ≠ j) is

<table>
<thead>
<tr>
<th></th>
<th>CRAPS</th>
<th>INITIALIZE</th>
<th>ACCEPT BET</th>
<th>PLAY</th>
<th>ROLL DICE</th>
<th>RANDOM</th>
<th>UP DATE</th>
</tr>
</thead>
<tbody>
<tr>
<td>CRAPS</td>
<td>1.0000</td>
<td>0.1370</td>
<td>0.2544</td>
<td>0.2181</td>
<td>0.0000</td>
<td>0.5592</td>
<td>0.5937</td>
</tr>
<tr>
<td>INITIALIZE</td>
<td>1.1951</td>
<td>1.0000</td>
<td>0.9512</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.9756</td>
<td>0.9512</td>
</tr>
<tr>
<td>ACCEPTBET</td>
<td>0.7778</td>
<td>0.3333</td>
<td>1.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.5556</td>
</tr>
<tr>
<td>PLAY</td>
<td>0.2857</td>
<td>0.0000</td>
<td>0.0000</td>
<td>1.0000</td>
<td>0.2857</td>
<td>0.0000</td>
<td>0.1429</td>
</tr>
<tr>
<td>ROLLDICE</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.3333</td>
<td>1.0000</td>
<td>0.8889</td>
<td>0.0000</td>
</tr>
<tr>
<td>RANDOM</td>
<td>0.1984</td>
<td>0.0397</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.2063</td>
<td>1.0000</td>
<td>0.0000</td>
</tr>
<tr>
<td>UPDATE</td>
<td>1.2895</td>
<td>0.2368</td>
<td>0.3947</td>
<td>0.2368</td>
<td>0.0000</td>
<td>0.0000</td>
<td>1.0000</td>
</tr>
</tbody>
</table>

Table 113. Module V Procedure to Procedure Normalized Coupling Matrix After Increasing the weight of WON in Main Procedure From 3 to 6.

The cumulative coupling of functions is:

<table>
<thead>
<tr>
<th>Function Name</th>
<th>CRAPS</th>
<th>INITIALIZE</th>
<th>ACCEPT BET</th>
<th>PLAY</th>
<th>ROLL DICE</th>
<th>RANDOM</th>
<th>UP DATE</th>
<th>System</th>
</tr>
</thead>
<tbody>
<tr>
<td>Coupling</td>
<td>1.633525</td>
<td>0.751271</td>
<td>0.857143</td>
<td>1.434062</td>
<td>2.200595</td>
<td>1.720148</td>
<td>2.543268</td>
<td>1.827970</td>
</tr>
<tr>
<td>Coupling Inverse</td>
<td>0.612173</td>
<td>1.331078</td>
<td>1.166667</td>
<td>0.697320</td>
<td>0.454423</td>
<td>0.581345</td>
<td>0.393195</td>
<td>0.547055</td>
</tr>
</tbody>
</table>

Table 114. Module V Procedures Overall Coupling After Increasing the weight of WON in Main procedure From 3 to 6.
5.3.4.2 Dhama Method

<table>
<thead>
<tr>
<th>Function Name</th>
<th>CRAPS</th>
<th>INITIALIZE</th>
<th>ACCEPT BET</th>
<th>PLAY</th>
<th>ROLL DICE</th>
<th>RANDOM</th>
<th>UP DATE</th>
<th>System</th>
</tr>
</thead>
<tbody>
<tr>
<td>Coupling Increasing</td>
<td>9</td>
<td>4</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>8</td>
<td>5.7143</td>
</tr>
<tr>
<td>Coupling Decreasing</td>
<td>0.1111</td>
<td>0.2500</td>
<td>0.2000</td>
<td>0.2000</td>
<td>0.2000</td>
<td>0.1250</td>
<td>0.1837</td>
<td></td>
</tr>
</tbody>
</table>

*Table 115. Module V Procedures Coupling Weights Using Dhama Method After Increasing the weight of WON in Main Procedure From 3 to 6.*

The coupling of procedure to procedure is shown in the following table.

<table>
<thead>
<tr>
<th></th>
<th>CRAPS</th>
<th>INITIALIZE</th>
<th>ACCEPT BET</th>
<th>PLAY</th>
<th>ROLL DICE</th>
<th>RANDOM</th>
<th>UP DATE</th>
</tr>
</thead>
<tbody>
<tr>
<td>CRAPS</td>
<td>0.1111</td>
<td>0.1806</td>
<td>0.1556</td>
<td>0.1556</td>
<td>0.1556</td>
<td>0.1556</td>
<td>0.1181</td>
</tr>
<tr>
<td>INITIALIZE</td>
<td>0.1806</td>
<td>0.2500</td>
<td>0.2250</td>
<td>0.2250</td>
<td>0.2250</td>
<td>0.2250</td>
<td>0.1875</td>
</tr>
<tr>
<td>ACCEPTBET</td>
<td>0.1556</td>
<td>0.2250</td>
<td>0.2000</td>
<td>0.2000</td>
<td>0.2000</td>
<td>0.2000</td>
<td>0.1625</td>
</tr>
<tr>
<td>PLAY</td>
<td>0.1556</td>
<td>0.2250</td>
<td>0.2000</td>
<td>0.2000</td>
<td>0.2000</td>
<td>0.2000</td>
<td>0.1625</td>
</tr>
<tr>
<td>ROLLDICE</td>
<td>0.1556</td>
<td>0.2250</td>
<td>0.2000</td>
<td>0.2000</td>
<td>0.2000</td>
<td>0.2000</td>
<td>0.1625</td>
</tr>
<tr>
<td>RANDOM</td>
<td>0.1556</td>
<td>0.2250</td>
<td>0.2000</td>
<td>0.2000</td>
<td>0.2000</td>
<td>0.2000</td>
<td>0.1625</td>
</tr>
<tr>
<td>UPDATE</td>
<td>0.1181</td>
<td>0.1875</td>
<td>0.1625</td>
<td>0.1625</td>
<td>0.1625</td>
<td>0.1625</td>
<td>0.1250</td>
</tr>
</tbody>
</table>

*Table 116. Module V Procedure to Procedure Coupling Weights Using Dhama Method After Increasing the weight of WON in Main Procedure From 3 to 6.*

5.3.4.3 Fenton and Melton Method

<table>
<thead>
<tr>
<th>Function Pair Coupled</th>
<th>CRAPS, INITIALIZE</th>
<th>CRAPS, ACCEPT BET</th>
<th>CRAPS, PLAY</th>
<th>CRAPS, UP DATE</th>
<th>CRAPS, RANDOM</th>
<th>PLAY, ROLL DICE</th>
<th>ROLL DICE, RANDOM</th>
<th>System</th>
</tr>
</thead>
</table>

*Table 117. Module V Procedure to Procedure Coupling Weights Using Fenton and Melton Method After Increasing the weight of WON in Main Procedure From 3 to 6.*
5.3.4.4 Offut, Harrold and Kote Method

<table>
<thead>
<tr>
<th>Function Pair Coupled</th>
<th>CRAPS, INITIALIZE</th>
<th>CRAPS, ACCEPT BET</th>
<th>CRAPS, PLAY</th>
<th>CRAPS, UPDATE</th>
<th>CRAPS, RANDOM</th>
<th>PLAY, ROLL DICE</th>
<th>ROLL DICE, RANDOM</th>
<th>System</th>
</tr>
</thead>
<tbody>
<tr>
<td>Calling to Called Coupling</td>
<td>2.8333</td>
<td>2.8333</td>
<td>3.0000</td>
<td>2.7000</td>
<td>4.8333</td>
<td>1.0000</td>
<td>1.0000</td>
<td>2.4571</td>
</tr>
<tr>
<td>Called to Calling Coupling</td>
<td>0.8333</td>
<td>2.8333</td>
<td>1.0000</td>
<td>2.7000</td>
<td>4.8333</td>
<td>3.0000</td>
<td>1.0000</td>
<td></td>
</tr>
</tbody>
</table>

*Table 118. Module V Procedure to Procedure Coupling Weights Using Offut, Harrold and Kote Method After Increasing the weight of WON in Main Procedure From 3 to 6.*

5.3.4.5 Chen and Lu Method

<table>
<thead>
<tr>
<th>Function Name</th>
<th>CRAPS</th>
<th>INITIALIZE</th>
<th>ACCEPT BET</th>
<th>PLAY</th>
<th>ROLL DICE</th>
<th>RANDOM</th>
<th>UPDATE</th>
<th>System</th>
</tr>
</thead>
<tbody>
<tr>
<td>Coupling</td>
<td>11</td>
<td>1</td>
<td>1</td>
<td>4</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>3.1428</td>
</tr>
</tbody>
</table>

*Table 119. Module V Procedure Coupling Weights Using Chen and Lu Method After Increasing the weight of WON in Main Procedure From 3 to 6.*

The procedure to procedure coupling values as in the following table.
Table 120. Module V Procedure by Procedure Coupling Values Using Chen and Lu Method After Increasing the weight of WON in Main Procedure From 3 to 6.

<table>
<thead>
<tr>
<th></th>
<th>CRAPS</th>
<th>INITIALIZE</th>
<th>ACCEPTBET</th>
<th>PLAY</th>
<th>ROLLDICE</th>
<th>RANDOM</th>
<th>UPDATE</th>
</tr>
</thead>
<tbody>
<tr>
<td>CRAPS</td>
<td>11.0000</td>
<td>6.0000</td>
<td>6.0000</td>
<td>7.5000</td>
<td>7.0000</td>
<td>6.0000</td>
<td>6.0000</td>
</tr>
<tr>
<td>INITIALIZE</td>
<td>6.0000</td>
<td>1.0000</td>
<td>1.0000</td>
<td>2.5000</td>
<td>2.0000</td>
<td>1.0000</td>
<td>1.0000</td>
</tr>
<tr>
<td>ACCEPTBET</td>
<td>6.0000</td>
<td>1.0000</td>
<td>1.0000</td>
<td>2.5000</td>
<td>2.0000</td>
<td>1.0000</td>
<td>1.0000</td>
</tr>
<tr>
<td>PLAY</td>
<td>7.5000</td>
<td>2.5000</td>
<td>2.5000</td>
<td>4.0000</td>
<td>3.5000</td>
<td>2.5000</td>
<td>2.5000</td>
</tr>
<tr>
<td>ROLLDICE</td>
<td>7.0000</td>
<td>2.0000</td>
<td>2.0000</td>
<td>3.5000</td>
<td>3.0000</td>
<td>2.0000</td>
<td>2.0000</td>
</tr>
<tr>
<td>RANDOM</td>
<td>6.0000</td>
<td>1.0000</td>
<td>1.0000</td>
<td>2.5000</td>
<td>2.0000</td>
<td>1.0000</td>
<td>1.0000</td>
</tr>
<tr>
<td>UPDATE</td>
<td>6.0000</td>
<td>1.0000</td>
<td>1.0000</td>
<td>2.5000</td>
<td>2.0000</td>
<td>1.0000</td>
<td>1.0000</td>
</tr>
</tbody>
</table>

5.3.4.6 Analysis

The effect of increasing the weight of WON variable against CRAPS procedure from 2 to 6 starts from the coupling matrix. The new method does not agree with the relation because the coupling of CRAPS has decreased while the coupling of RANDOM, ROLLDICE and system have increased. In addition, the coupling of the other procedures have decreased. The reason for that the proportion of change from 2 to 6 in MAIN is not low compared with others like RANDOM.

Dharma measure does not preserve the relation because CRAPS and system coupling have decreased. Fenton and Melton measure, Offut, Harrold and Kotle measure and Chen and Lu measure do not satisfy the relation because no coupling is changed in any of their procedures or system.
5.3.5 Reduce Element Weight Module VI

Reduce the value of the outstat variable by making it a parameter instead of a global variable in writestat function and let do_stats function calls writestat with outstat as a parameter to see how can this effect the methods.
### 5.3.5.1 New Method

The definition matrix is

|            | ndocs | iinode | ioinare | parareas | bletters | aorh | whrds | sentences | pchars | statline | eticle | stlline | ctitle | staat | staat | staat | staat | staat | staat | staat | staat | staat | staat | staat | staat | staat | staat |
|------------|-------|--------|---------|----------|----------|------|-------|-----------|--------|----------|--------|--------|--------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|
| main       | 3 2   | 5 0    | 2 5 5   | 5 5 5    | 5 5 5    | 5 5 5| 5 5 5 | 5 1 5     | 5 5 2 1| 1 1 0 10 |       |        |        |       |       |       |       |       |       |       |       |       |       |       |       |       |
| initialize | 0 0   | 0 0 0 0| 0 0 0 0 | 0 0 0 0 0| 0 0 0 0 0| 0 0 0 0 0| 0 0 0 0 0| 0 0 0 5 5| 0 0 0 0 0| 8 8 0 0 0|       |        |        |       |       |       |       |       |       |       |       |       |       |       |       |       |
| open_files | 8 0   | 0 0 0 0| 0 0 0 0 | 0 0 0 0 0| 0 0 0 0 0| 0 0 0 0 0| 0 0 0 0 0| 0 0 0 0 0| 0 0 0 0 0| 8 8 0 0 0|       |        |        |       |       |       |       |       |       |       |       |       |       |       |       |       |
| readdoc    | 1 6   | 0 1 4 | 0 0 0 0 | 0 0 0 0 0| 0 0 0 0 0| 0 0 0 0 0| 0 0 0 0 0| 0 0 0 0 0| 0 0 0 0 0| 6 0 0 0 0|       |        |        |       |       |       |       |       |       |       |       |       |       |       |       |       |
| wriedoc    | 0 0   | 1 0 4 | 0 0 0 0 | 0 0 0 0 0| 0 0 0 0 0| 0 0 0 0 0| 0 0 0 0 0| 0 0 0 1 0| 0 1 0 1 0| 0 1 0 1 0|       |        |        |       |       |       |       |       |       |       |       |       |       |       |       |       |
| do_stats   | 0 5   | 0 5 5 | 7 5 5 5 | 5 5 5 5 5| 5 5 5 5 5| 5 5 5 5 5| 0 0 0 5 5| 0 5 5 0 0| 0 0 0 0 0| 0 0 0 0 0|       |        |        |       |       |       |       |       |       |       |       |       |       |       |       |       |
| write_stats| 0 0   | 0 0 0 0| 0 0 0 0 | 0 0 0 0 0| 0 0 0 0 0| 0 0 0 0 0| 0 0 0 0 0| 0 0 0 0 0| 0 0 2 7 8| 0 0 0 0 0|       |        |        |       |       |       |       |       |       |       |       |       |       |       |       |       |
| close_files| 8 0   | 0 0 0 0| 0 0 0 0 | 0 0 0 0 0| 0 0 0 0 0| 0 0 0 0 0| 0 0 0 0 0| 0 0 0 0 0| 0 0 8 8 0| 0 0 0 0 0|       |        |        |       |       |       |       |       |       |       |       |       |       |       |       |       |

*Table 121. Module VI Definition Matrix After Decreasing the weight of outstat in write_stat Procedure From 60 to 20.*

The coupling matrix is
<table>
<thead>
<tr>
<th></th>
<th>main</th>
<th>initialize</th>
<th>open_files</th>
<th>read_doc</th>
<th>write_doc</th>
<th>do_stats</th>
<th>write_stats</th>
<th>close_files</th>
</tr>
</thead>
<tbody>
<tr>
<td>main</td>
<td>0.4451</td>
<td>0.0259</td>
<td>0.0449</td>
<td>0.0911</td>
<td>0.0899</td>
<td>0.2203</td>
<td>0.0379</td>
<td>0.0449</td>
</tr>
<tr>
<td>initialize</td>
<td>0.5000</td>
<td>0.3333</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.1667</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
</tr>
<tr>
<td>open_files</td>
<td>0.3613</td>
<td>0.0000</td>
<td>0.1202</td>
<td>0.0833</td>
<td>0.0769</td>
<td>0.1701</td>
<td>0.0680</td>
<td>0.1202</td>
</tr>
<tr>
<td>readdoc</td>
<td>0.4884</td>
<td>0.0000</td>
<td>0.0556</td>
<td>0.2870</td>
<td>0.1134</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0556</td>
</tr>
<tr>
<td>writedoc</td>
<td>0.3469</td>
<td>0.0333</td>
<td>0.0369</td>
<td>0.0817</td>
<td>0.2309</td>
<td>0.0972</td>
<td>0.1361</td>
<td>0.0369</td>
</tr>
<tr>
<td>do_stats</td>
<td>0.1959</td>
<td>0.0000</td>
<td>0.0188</td>
<td>0.0000</td>
<td>0.0224</td>
<td>0.4433</td>
<td>0.3008</td>
<td>0.0188</td>
</tr>
<tr>
<td>write_stats</td>
<td>0.0430</td>
<td>0.0000</td>
<td>0.0096</td>
<td>0.0000</td>
<td>0.0400</td>
<td>0.3840</td>
<td>0.5138</td>
<td>0.0096</td>
</tr>
<tr>
<td>close_files</td>
<td>0.3613</td>
<td>0.0000</td>
<td>0.1202</td>
<td>0.0833</td>
<td>0.0769</td>
<td>0.1701</td>
<td>0.0680</td>
<td>0.1202</td>
</tr>
</tbody>
</table>

Table 122. Module VI Procedure to Procedure Coupling Matrix After Decreasing the weight of outstat in write_stat Procedure From 60 to 20.

The normalized coupling matrix: (coupling of functions i & j where $i \neq j$) is

<table>
<thead>
<tr>
<th></th>
<th>main</th>
<th>initialize</th>
<th>open_files</th>
<th>read_doc</th>
<th>write_doc</th>
<th>do_stats</th>
<th>write_stats</th>
<th>close_files</th>
</tr>
</thead>
<tbody>
<tr>
<td>main</td>
<td>1.0000</td>
<td>0.0582</td>
<td>0.1009</td>
<td>0.2047</td>
<td>0.2019</td>
<td>0.4949</td>
<td>0.0851</td>
<td>0.1009</td>
</tr>
<tr>
<td>initialize</td>
<td>1.5000</td>
<td>1.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.5000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
</tr>
<tr>
<td>open_files</td>
<td>3.0071</td>
<td>0.0000</td>
<td>1.0000</td>
<td>0.6935</td>
<td>0.6402</td>
<td>1.4154</td>
<td>0.5661</td>
<td>1.0000</td>
</tr>
<tr>
<td>readdoc</td>
<td>1.7016</td>
<td>0.0000</td>
<td>0.1935</td>
<td>1.0000</td>
<td>0.3952</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.1935</td>
</tr>
<tr>
<td>writedoc</td>
<td>1.5020</td>
<td>0.1443</td>
<td>0.1599</td>
<td>0.3536</td>
<td>1.0000</td>
<td>0.4210</td>
<td>0.5894</td>
<td>0.1599</td>
</tr>
<tr>
<td>do_stats</td>
<td>0.4420</td>
<td>0.0000</td>
<td>0.0424</td>
<td>0.0000</td>
<td>0.0505</td>
<td>1.0000</td>
<td>0.6787</td>
<td>0.0424</td>
</tr>
<tr>
<td>write_stats</td>
<td>0.0837</td>
<td>0.0000</td>
<td>0.0187</td>
<td>0.0000</td>
<td>0.0779</td>
<td>0.7474</td>
<td>1.0000</td>
<td>0.0187</td>
</tr>
<tr>
<td>close_files</td>
<td>3.0071</td>
<td>0.0000</td>
<td>1.0000</td>
<td>0.6935</td>
<td>0.6402</td>
<td>1.4154</td>
<td>0.5661</td>
<td>1.0000</td>
</tr>
</tbody>
</table>

Table 123. Module VI Procedure to Procedure Normalized Coupling Matrix After Decreasing the weight of outstat in write_stat Procedure From 60 to 20.

The cumulative coupling of functions is:

<table>
<thead>
<tr>
<th>Function _Name</th>
<th>main</th>
<th>initialize</th>
<th>open_files</th>
<th>read_doc</th>
<th>write_doc</th>
<th>do_stats</th>
<th>write_stats</th>
<th>close_files</th>
<th>System</th>
</tr>
</thead>
<tbody>
<tr>
<td>Coupling</td>
<td>6.245038</td>
<td>1.101270</td>
<td>4.418884</td>
<td>2.214610</td>
<td>2.917993</td>
<td>2.875903</td>
<td>1.715921</td>
<td>4.418883</td>
<td>3.238450</td>
</tr>
<tr>
<td>Coupling _Inverse</td>
<td>0.160127</td>
<td>0.908043</td>
<td>0.226301</td>
<td>0.461547</td>
<td>0.342701</td>
<td>0.347826</td>
<td>0.582778</td>
<td>0.226302</td>
<td>0.308790</td>
</tr>
</tbody>
</table>

Table 124. Module VI Procedures Overall Coupling After Decreasing the weight of outstat in write_stat Procedure From 60 to 20.
### 5.3.5.2 Dhama Method

<table>
<thead>
<tr>
<th>Function Name</th>
<th>main</th>
<th>init-</th>
<th>open_</th>
<th>read</th>
<th>write</th>
<th>do_</th>
<th>write</th>
<th>close</th>
<th>System</th>
</tr>
</thead>
<tbody>
<tr>
<td>Coupling</td>
<td>26</td>
<td>5</td>
<td>7</td>
<td>8</td>
<td>19</td>
<td>7</td>
<td>16</td>
<td>7</td>
<td>10.5555</td>
</tr>
<tr>
<td>Coupling Inverse</td>
<td>0.0384</td>
<td>0.2000</td>
<td>0.1428</td>
<td>0.1250</td>
<td>0.0526</td>
<td>0.1428</td>
<td>0.0625</td>
<td>0.1428</td>
<td>0.1133</td>
</tr>
</tbody>
</table>

*Table 125. Module VI Procedures Coupling Weights Using Dhama Method After Decreasing the weight of outstat in write_stat Procedure From 60 to 20.*

The coupling of procedure to procedure is shown in the following table.

<table>
<thead>
<tr>
<th></th>
<th>main</th>
<th>init-</th>
<th>open_</th>
<th>read</th>
<th>write</th>
<th>do_</th>
<th>write</th>
<th>close</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>main</td>
<td>0.0384</td>
<td>0.1192</td>
<td>0.1812</td>
<td>0.0817</td>
<td>0.0455</td>
<td>0.1812</td>
<td>0.0505</td>
<td>0.1812</td>
<td></td>
</tr>
<tr>
<td>initialize</td>
<td>0.1192</td>
<td>0.2000</td>
<td>0.1740</td>
<td>0.1625</td>
<td>0.1263</td>
<td>0.1740</td>
<td>0.1313</td>
<td>0.1740</td>
<td></td>
</tr>
<tr>
<td>open_files</td>
<td>0.1812</td>
<td>0.1740</td>
<td>0.1428</td>
<td>0.1339</td>
<td>0.0977</td>
<td>0.1428</td>
<td>0.1027</td>
<td>0.1428</td>
<td></td>
</tr>
<tr>
<td>readdoc</td>
<td>0.0817</td>
<td>0.1625</td>
<td>0.1339</td>
<td>0.1250</td>
<td>0.0888</td>
<td>0.1339</td>
<td>0.0938</td>
<td>0.1339</td>
<td></td>
</tr>
<tr>
<td>writedoc</td>
<td>0.0455</td>
<td>0.1263</td>
<td>0.0977</td>
<td>0.0888</td>
<td>0.0526</td>
<td>0.0977</td>
<td>0.0576</td>
<td>0.0977</td>
<td></td>
</tr>
<tr>
<td>do_stats</td>
<td>0.1812</td>
<td>0.1740</td>
<td>0.1428</td>
<td>0.1339</td>
<td>0.0977</td>
<td>0.1428</td>
<td>0.1027</td>
<td>0.1428</td>
<td></td>
</tr>
<tr>
<td>write_stats</td>
<td>0.0505</td>
<td>0.1313</td>
<td>0.1027</td>
<td>0.0938</td>
<td>0.0576</td>
<td>0.1027</td>
<td>0.0625</td>
<td>0.1027</td>
<td></td>
</tr>
<tr>
<td>close_files</td>
<td>0.1812</td>
<td>0.1740</td>
<td>0.1428</td>
<td>0.1339</td>
<td>0.0977</td>
<td>0.1428</td>
<td>0.1027</td>
<td>0.1428</td>
<td></td>
</tr>
</tbody>
</table>

*Table 126. Module VI Procedure to Procedure Coupling Weights Using Dhama Method After Decreasing the weight of outstat in write_stat Procedure From 60 to 20.*

### 5.3.5.3 Fenton and Melton Method

<table>
<thead>
<tr>
<th>Function Pair Coupled</th>
<th>main, init-</th>
<th>main, open_</th>
<th>main, read</th>
<th>main, write</th>
<th>main, do_</th>
<th>main, close</th>
<th>do_stats, write</th>
<th>System</th>
</tr>
</thead>
</table>

*Table 127. Module VI Procedure to Procedure Coupling Weights Using Fenton and Melton Method After Decreasing the weight of outstat in write_stat Procedure From 60 to 20.*
5.3.5.4 Offut, Harrold and Kotle Method

<table>
<thead>
<tr>
<th>Function Pair Coupled</th>
<th>main, initialize</th>
<th>main, open_files</th>
<th>main, read doc</th>
<th>main, write doc</th>
<th>main, do_stats</th>
<th>main, close files</th>
<th>do_stats, write_stats</th>
<th>System</th>
</tr>
</thead>
<tbody>
<tr>
<td>Calling to Called Coupling Level</td>
<td>3.7500</td>
<td>3.7500</td>
<td>3.8333</td>
<td>3.6000</td>
<td>3.7000</td>
<td>3.7000</td>
<td>3.8333</td>
<td>3.3780</td>
</tr>
<tr>
<td>Called to Calling Coupling Level</td>
<td>3.7500</td>
<td>3.7500</td>
<td>3.8333</td>
<td>3.6000</td>
<td>3.7000</td>
<td>3.7000</td>
<td>3.8333</td>
<td></td>
</tr>
</tbody>
</table>

*Table 128. Module VI Procedure to Procedure Coupling Weights Using Offut, Harrold and Kotle Method After Decreasing the weight of outstat in write_stat Procedure From 60 to 20.*

5.3.5.5 Chen and Lu Method

<table>
<thead>
<tr>
<th>Function Name</th>
<th>main</th>
<th>initialize</th>
<th>open_files</th>
<th>read doc</th>
<th>write doc</th>
<th>do_stats</th>
<th>write_stats</th>
<th>close_files</th>
<th>System</th>
</tr>
</thead>
<tbody>
<tr>
<td>Coupling</td>
<td>16</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>3.125</td>
</tr>
</tbody>
</table>

*Table 129. Module VI Procedures Coupling Values Using Chen and Lu Method After Decreasing the weight of outstat in write_stat Procedure From 60 to 20.*

The procedure to procedure coupling values are shown in the following table.
<table>
<thead>
<tr>
<th></th>
<th>main</th>
<th>initialize</th>
<th>open_files</th>
<th>read_doc</th>
<th>write_doc</th>
<th>do_stats</th>
<th>write_stats</th>
<th>close_files</th>
</tr>
</thead>
<tbody>
<tr>
<td>main</td>
<td>16.0000</td>
<td>8.5000</td>
<td>8.5000</td>
<td>8.5000</td>
<td>8.5000</td>
<td>9.5000</td>
<td>8.5000</td>
<td>8.5000</td>
</tr>
<tr>
<td>initialize</td>
<td>8.5000</td>
<td>1.0000</td>
<td>1.0000</td>
<td>1.0000</td>
<td>1.0000</td>
<td>2.0000</td>
<td>1.0000</td>
<td>1.0000</td>
</tr>
<tr>
<td>open_files</td>
<td>8.5000</td>
<td>1.0000</td>
<td>1.0000</td>
<td>1.0000</td>
<td>1.0000</td>
<td>2.0000</td>
<td>1.0000</td>
<td>1.0000</td>
</tr>
<tr>
<td>readdoc</td>
<td>8.5000</td>
<td>1.0000</td>
<td>1.0000</td>
<td>1.0000</td>
<td>1.0000</td>
<td>2.0000</td>
<td>1.0000</td>
<td>1.0000</td>
</tr>
<tr>
<td>writedoc</td>
<td>8.5000</td>
<td>1.0000</td>
<td>1.0000</td>
<td>1.0000</td>
<td>1.0000</td>
<td>2.0000</td>
<td>1.0000</td>
<td>1.0000</td>
</tr>
<tr>
<td>do_stats</td>
<td>9.5000</td>
<td>2.0000</td>
<td>2.0000</td>
<td>2.0000</td>
<td>2.0000</td>
<td>3.0000</td>
<td>2.0000</td>
<td>2.0000</td>
</tr>
<tr>
<td>write_stats</td>
<td>8.5000</td>
<td>1.0000</td>
<td>1.0000</td>
<td>1.0000</td>
<td>1.0000</td>
<td>2.0000</td>
<td>1.0000</td>
<td>1.0000</td>
</tr>
<tr>
<td>close_files</td>
<td>8.5000</td>
<td>1.0000</td>
<td>1.0000</td>
<td>1.0000</td>
<td>1.0000</td>
<td>2.0000</td>
<td>1.0000</td>
<td>1.0000</td>
</tr>
</tbody>
</table>

Table 130. Module VI Procedure by Procedure Coupling Values Using Chen and Lu Method After Decreasing the weight of outstat in write_stat Procedure From 60 to 20.

5.3.5.6 Analysis

The effect of reducing the weight of outstat variable against write_stats procedure from 60 to 20 starts from the coupling matrix. The new method recognizes the relation because because all procedures and system couplings have decreased except do_stats procedure coupling has increased.

Dhama measure, Fenton and Melton measure, Offut, Harrold and Kotle measure and Chen and Lu measure do not capture the relation because no coupling is changed in any of thier procedures or system.

5.3.6 Use Parameters in Module VI

Make module VI communicate through parameter passing instead of global variables. So, the effect is to reduce the weights of the elements.
### 5.3.6.1 New Method

The definition matrix is

<table>
<thead>
<tr>
<th></th>
<th>node1</th>
<th>node2</th>
<th>node3</th>
<th>node4</th>
<th>node5</th>
<th>node6</th>
<th>node7</th>
<th>node8</th>
<th>node9</th>
<th>node10</th>
<th>node11</th>
<th>node12</th>
<th>node13</th>
<th>node14</th>
<th>node15</th>
<th>node16</th>
<th>node17</th>
</tr>
</thead>
<tbody>
<tr>
<td>main</td>
<td>1</td>
<td>2</td>
<td>9</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>init</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>open</td>
<td>4</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>4</td>
</tr>
<tr>
<td>read</td>
<td>4</td>
<td>0</td>
<td>6</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>0</td>
<td>4</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>write</td>
<td>0</td>
<td>0</td>
<td>6</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>do</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>write</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>close</td>
<td>4</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>4</td>
</tr>
</tbody>
</table>

*Table 13. Module VI Definition Matrix After Making It Uses Parameters.*

The coupling matrix is
<table>
<thead>
<tr>
<th></th>
<th>main</th>
<th>init-</th>
<th>open_</th>
<th>read</th>
<th>write</th>
<th>do_</th>
<th>write</th>
<th>close</th>
</tr>
</thead>
<tbody>
<tr>
<td>main</td>
<td>0.4147</td>
<td>0.0217</td>
<td>0.0602</td>
<td>0.1320</td>
<td>0.0955</td>
<td>0.1667</td>
<td>0.0489</td>
<td>0.0602</td>
</tr>
<tr>
<td>initialize</td>
<td>0.5000</td>
<td>0.3333</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.1667</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
</tr>
<tr>
<td>open_files</td>
<td>0.2310</td>
<td>0.0000</td>
<td>0.1841</td>
<td>0.0556</td>
<td>0.0952</td>
<td>0.0833</td>
<td>0.1667</td>
<td>0.1841</td>
</tr>
<tr>
<td>readdoc</td>
<td>0.3795</td>
<td>0.0000</td>
<td>0.0417</td>
<td>0.4301</td>
<td>0.1071</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0417</td>
</tr>
<tr>
<td>writedoc</td>
<td>0.2440</td>
<td>0.0185</td>
<td>0.0635</td>
<td>0.0952</td>
<td>0.2374</td>
<td>0.0694</td>
<td>0.2083</td>
<td>0.0635</td>
</tr>
<tr>
<td>do_stats</td>
<td>0.1704</td>
<td>0.0000</td>
<td>0.0222</td>
<td>0.0000</td>
<td>0.0278</td>
<td>0.3296</td>
<td>0.4278</td>
<td>0.0222</td>
</tr>
<tr>
<td>write_stats</td>
<td>0.0250</td>
<td>0.0000</td>
<td>0.0222</td>
<td>0.0000</td>
<td>0.0417</td>
<td>0.2139</td>
<td>0.6750</td>
<td>0.0222</td>
</tr>
<tr>
<td>close_files</td>
<td>0.2310</td>
<td>0.0000</td>
<td>0.1841</td>
<td>0.0556</td>
<td>0.0952</td>
<td>0.0833</td>
<td>0.1667</td>
<td>0.1841</td>
</tr>
</tbody>
</table>

Table 132. Module VI Procedure to Procedure Coupling Matrix After Making It Communicate through Parameter Passing.

The normalized coupling matrix: (coupling of functions i & j where i≠j) is

<table>
<thead>
<tr>
<th></th>
<th>main</th>
<th>init-</th>
<th>open_</th>
<th>read</th>
<th>write</th>
<th>do_</th>
<th>write</th>
<th>close</th>
</tr>
</thead>
<tbody>
<tr>
<td>main</td>
<td>1.0000</td>
<td>0.0524</td>
<td>0.1453</td>
<td>0.3183</td>
<td>0.2303</td>
<td>0.4019</td>
<td>0.1179</td>
<td>0.1453</td>
</tr>
<tr>
<td>initialize</td>
<td>1.5000</td>
<td>1.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.5000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
</tr>
<tr>
<td>open_files</td>
<td>1.2543</td>
<td>0.0000</td>
<td>1.0000</td>
<td>0.3017</td>
<td>0.5172</td>
<td>0.4526</td>
<td>0.9052</td>
<td>1.0000</td>
</tr>
<tr>
<td>readdoc</td>
<td>0.8824</td>
<td>0.0000</td>
<td>0.0969</td>
<td>1.0000</td>
<td>0.2491</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0969</td>
</tr>
<tr>
<td>writedoc</td>
<td>1.0279</td>
<td>0.0780</td>
<td>0.2674</td>
<td>0.4011</td>
<td>1.0000</td>
<td>0.2925</td>
<td>0.8774</td>
<td>0.2674</td>
</tr>
<tr>
<td>do_stats</td>
<td>0.5169</td>
<td>0.0000</td>
<td>0.0674</td>
<td>0.0000</td>
<td>0.0843</td>
<td>1.0000</td>
<td>1.2978</td>
<td>0.0674</td>
</tr>
<tr>
<td>write_stats</td>
<td>0.0370</td>
<td>0.0000</td>
<td>0.0329</td>
<td>0.0000</td>
<td>0.0617</td>
<td>0.3169</td>
<td>1.0000</td>
<td>0.0329</td>
</tr>
<tr>
<td>close_files</td>
<td>1.2543</td>
<td>0.0000</td>
<td>1.0000</td>
<td>0.3017</td>
<td>0.5172</td>
<td>0.4526</td>
<td>0.9052</td>
<td>1.0000</td>
</tr>
</tbody>
</table>

Table 133. Module VI Procedure to Procedure Normalized Coupling Matrix After Making It Communicate through Parameter Passing.

The cumulative coupling of functions is:

<table>
<thead>
<tr>
<th>Function Name</th>
<th>main</th>
<th>init-</th>
<th>open_</th>
<th>read</th>
<th>write</th>
<th>do_</th>
<th>write</th>
<th>close</th>
<th>System</th>
</tr>
</thead>
<tbody>
<tr>
<td>Coupling</td>
<td>3.942051</td>
<td>1.0652983</td>
<td>3.030475</td>
<td>1.324047</td>
<td>2.685797</td>
<td>1.975064</td>
<td>2.292482</td>
<td>3.820475</td>
<td>2.415700</td>
</tr>
<tr>
<td>Inverse</td>
<td>0.253675</td>
<td>0.3938784</td>
<td>0.331074</td>
<td>0.755260</td>
<td>0.372329</td>
<td>0.506313</td>
<td>0.436208</td>
<td>0.331074</td>
<td>0.413959</td>
</tr>
</tbody>
</table>

Table 134. Module VI Procedures Overall Coupling After Making It Communicate through Parameter Passing.
5.3.6.2 Dhama Method

<table>
<thead>
<tr>
<th>Function Name</th>
<th>main</th>
<th>initialize</th>
<th>open_files</th>
<th>read_doc</th>
<th>write_doc</th>
<th>do_stats</th>
<th>write_stats</th>
<th>close_files</th>
<th>System</th>
</tr>
</thead>
<tbody>
<tr>
<td>Coupling</td>
<td>26</td>
<td>5</td>
<td>7</td>
<td>8</td>
<td>19</td>
<td>7</td>
<td>16</td>
<td>7</td>
<td>10.5555</td>
</tr>
<tr>
<td>Coupling Inverse</td>
<td>0.0384</td>
<td>0.2000</td>
<td>0.1428</td>
<td>0.1250</td>
<td>0.0526</td>
<td>0.1428</td>
<td>0.0625</td>
<td>0.1428</td>
<td>0.1133</td>
</tr>
</tbody>
</table>

Table 135. Module VI Procedures Coupling Weights Using Dhama Method After Making It Communicate through Parameter Passing.

The coupling of procedure to procedure is shown in the following table.

<table>
<thead>
<tr>
<th></th>
<th>main</th>
<th>initialize</th>
<th>open_files</th>
<th>read_doc</th>
<th>write_doc</th>
<th>do_stats</th>
<th>write_stats</th>
<th>close_files</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>main</td>
<td>0.0384</td>
<td>0.1192</td>
<td>0.1812</td>
<td>0.0817</td>
<td>0.0455</td>
<td>0.1812</td>
<td>0.0505</td>
<td>0.1812</td>
<td></td>
</tr>
<tr>
<td>initialize</td>
<td>0.1192</td>
<td>0.2000</td>
<td>0.1740</td>
<td>0.1625</td>
<td>0.1263</td>
<td>0.1740</td>
<td>0.1313</td>
<td>0.1740</td>
<td></td>
</tr>
<tr>
<td>open_files</td>
<td>0.1812</td>
<td>0.1740</td>
<td>0.1428</td>
<td>0.1339</td>
<td>0.0977</td>
<td>0.1428</td>
<td>0.1027</td>
<td>0.1428</td>
<td></td>
</tr>
<tr>
<td>readdoc</td>
<td>0.0817</td>
<td>0.1625</td>
<td>0.1339</td>
<td>0.1250</td>
<td>0.0888</td>
<td>0.1339</td>
<td>0.0938</td>
<td>0.1339</td>
<td></td>
</tr>
<tr>
<td>wriotedoc</td>
<td>0.0455</td>
<td>0.1263</td>
<td>0.0977</td>
<td>0.0888</td>
<td>0.0526</td>
<td>0.0977</td>
<td>0.0576</td>
<td>0.0977</td>
<td></td>
</tr>
<tr>
<td>do_stats</td>
<td>0.1812</td>
<td>0.1740</td>
<td>0.1428</td>
<td>0.1339</td>
<td>0.0977</td>
<td>0.1428</td>
<td>0.1027</td>
<td>0.1428</td>
<td></td>
</tr>
<tr>
<td>write_stats</td>
<td>0.0505</td>
<td>0.1313</td>
<td>0.1027</td>
<td>0.0938</td>
<td>0.0576</td>
<td>0.1027</td>
<td>0.0625</td>
<td>0.1027</td>
<td></td>
</tr>
<tr>
<td>close_files</td>
<td>0.1812</td>
<td>0.1740</td>
<td>0.1428</td>
<td>0.1339</td>
<td>0.0977</td>
<td>0.1428</td>
<td>0.1027</td>
<td>0.1428</td>
<td></td>
</tr>
</tbody>
</table>

Table 136. Module VI Procedure to Procedure Coupling Weights Using Dhama Method After Making It Communicate through Parameter Passing.

5.3.6.3 Fenton and Melton Method

<table>
<thead>
<tr>
<th>Function Pair Coupled</th>
<th>main, initialize</th>
<th>main, open_files</th>
<th>main, read_doc</th>
<th>main, write_doc</th>
<th>main, do_stats</th>
<th>main, close_files</th>
<th>do_stats, write_stats</th>
<th>System</th>
</tr>
</thead>
</table>

Table 137. Module VI Procedure to Procedure Coupling Weights Using Fenton and Melton Method After Making It Communicate through Parameter Passing.
5.3.6.4 Offut, Harrold and Kotle Method

<table>
<thead>
<tr>
<th>Function Pair Coupled</th>
<th>main, initialize</th>
<th>main, open_files</th>
<th>main, read doc</th>
<th>main, write doc</th>
<th>main, do_stats</th>
<th>main, close_files</th>
<th>do_stats, write_stats</th>
<th>System</th>
</tr>
</thead>
<tbody>
<tr>
<td>Calling to Called Coupling Level</td>
<td>3.7500</td>
<td>3.7500</td>
<td>3.8333</td>
<td>3.6000</td>
<td>3.7000</td>
<td>3.7000</td>
<td>3.8333</td>
<td>3.3780</td>
</tr>
<tr>
<td>Called to Calling Coupling Level</td>
<td>3.7500</td>
<td>3.7500</td>
<td>3.8333</td>
<td>3.6000</td>
<td>3.7000</td>
<td>3.7000</td>
<td>3.8333</td>
<td></td>
</tr>
</tbody>
</table>

*Table 138. Module VI Procedure to Procedure Coupling Weights Using Offut, Harrold and Kotle Method After Making It Communicate through Parameter Passing.*

5.3.6.5 Chen and Lu Method

<table>
<thead>
<tr>
<th>Function Name</th>
<th>main</th>
<th>initialize</th>
<th>open_files</th>
<th>read doc</th>
<th>write doc</th>
<th>do_stats</th>
<th>write stats</th>
<th>close_files</th>
<th>System</th>
</tr>
</thead>
<tbody>
<tr>
<td>Coupling</td>
<td>16</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>3.125</td>
</tr>
</tbody>
</table>

*Table 139. Module VI Procedures Coupling Values Using Chen and Lu Method After Making It Communicate through Parameter Passing.*

The procedure to procedure coupling values are shown in the following table.
<table>
<thead>
<tr>
<th></th>
<th>main</th>
<th>initialize</th>
<th>open_files</th>
<th>read_doc</th>
<th>write_doc</th>
<th>do_stats</th>
<th>write_stats</th>
<th>close_files</th>
</tr>
</thead>
<tbody>
<tr>
<td>main</td>
<td>16.0000</td>
<td>8.5000</td>
<td>8.5000</td>
<td>8.5000</td>
<td>8.5000</td>
<td>9.5000</td>
<td>8.5000</td>
<td>8.5000</td>
</tr>
<tr>
<td>initialize</td>
<td>8.5000</td>
<td>1.0000</td>
<td>1.0000</td>
<td>1.0000</td>
<td>1.0000</td>
<td>2.0000</td>
<td>1.0000</td>
<td>1.0000</td>
</tr>
<tr>
<td>open_files</td>
<td>8.5000</td>
<td>1.0000</td>
<td>1.0000</td>
<td>1.0000</td>
<td>1.0000</td>
<td>2.0000</td>
<td>1.0000</td>
<td>1.0000</td>
</tr>
<tr>
<td>readdoc</td>
<td>8.5000</td>
<td>1.0000</td>
<td>1.0000</td>
<td>1.0000</td>
<td>1.0000</td>
<td>2.0000</td>
<td>1.0000</td>
<td>1.0000</td>
</tr>
<tr>
<td>writedoc</td>
<td>8.5000</td>
<td>1.0000</td>
<td>1.0000</td>
<td>1.0000</td>
<td>1.0000</td>
<td>2.0000</td>
<td>1.0000</td>
<td>1.0000</td>
</tr>
<tr>
<td>do_stats</td>
<td>9.5000</td>
<td>2.0000</td>
<td>2.0000</td>
<td>2.0000</td>
<td>2.0000</td>
<td>3.0000</td>
<td>2.0000</td>
<td>2.0000</td>
</tr>
<tr>
<td>write_stats</td>
<td>8.5000</td>
<td>1.0000</td>
<td>1.0000</td>
<td>1.0000</td>
<td>1.0000</td>
<td>2.0000</td>
<td>1.0000</td>
<td>1.0000</td>
</tr>
<tr>
<td>close_files</td>
<td>8.5000</td>
<td>1.0000</td>
<td>1.0000</td>
<td>1.0000</td>
<td>1.0000</td>
<td>2.0000</td>
<td>1.0000</td>
<td>1.0000</td>
</tr>
</tbody>
</table>

*Table 140. Module VI Procedure by Procedure Coupling Values Using Chen and Lu Method After Making It Communicate through Parameter Passing.*

5.3.6.6 Analysis

The effect of making module VI communicates through parameters instead of globals reduce the weights of these variables in the definition matrix. The new method agrees with the relation because the couplings of all procedures and system have decreased.

Dhana measure, Fenton and Melton measure, Offut, Harrold and Kotle measure and Chen and Lu measure do not satisfy the relation because no coupling is changed in any of thier procedures or system.

5.3.7 Analysis Summary

Table 141 summarizes the changes on each method where:

+ : The coupling of the preceding has increased
- : The coupling of the preceding has decreased and

↔ : The coupling of left procedure to the right and vice versa

<table>
<thead>
<tr>
<th>Action Description</th>
<th>New</th>
<th>Dharna</th>
<th>Fenton and Melton</th>
<th>Offut, Harrold and Kotte</th>
<th>Chen and Lu</th>
</tr>
</thead>
<tbody>
<tr>
<td>Remove A Procedure</td>
<td>All procedures(-), System (-)</td>
<td>System (-)</td>
<td>System (-)</td>
<td>System (+)</td>
<td>Sort2 (-), System (-)</td>
</tr>
<tr>
<td>Combine two Procedures</td>
<td>All procedures (-), System (-)</td>
<td>System (+)</td>
<td>System (+)</td>
<td>System (+), AVGCAL ↔ SUMCAL (+)</td>
<td>AVCALC (+), System (+)</td>
</tr>
<tr>
<td>Eliminate One Procedure and Use Equivalent One</td>
<td>LEAP (+), DAYS (+), MAIN (+), VALDAY (-), System (-)</td>
<td>MAIN (+), DAYS (+), LEAP (-), System (-)</td>
<td>None</td>
<td>None</td>
<td>LEAP (+), System (+)</td>
</tr>
<tr>
<td>Increase the weight of an element</td>
<td>PLAY (+), RANDOM (+), ROLLDICE (+), System (+), All others (-)</td>
<td>CRAPS (-), System (-)</td>
<td>None</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>Decrease the weight of an element</td>
<td>do_stats (+), System (-), All others (-)</td>
<td>None</td>
<td>None</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>Make module VI communicate through parameters</td>
<td>All procedure (-), System (-)</td>
<td>None</td>
<td>None</td>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

**Table 141. Comparisons of New and Recent Measures on Capturing Relations on Modules for the Six Atomic modifications.**

From the above table and the corresponding analysis on each change, the new measure preserves five out of the six relations. All other measures do not recognize any relation. However, the new measure is sensitive to all of these six changes. Dharna measure at least captures some response to the first four changes. Fenton and Melton measure, Offut, Harrold and Kotle measure and Chen and Lu measure
have some response to the first two changes. The following table shows how sensitive the new measure and the four recent measures to the atomic modifications.

<table>
<thead>
<tr>
<th></th>
<th>Remove Proc (II)</th>
<th>Combine Proc (III)</th>
<th>Replace Proc (IV)</th>
<th>Inc Elm Weight (V)</th>
<th>Dec Elm Weight (VI)</th>
<th>Use Param (VI)</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>New</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>6</td>
</tr>
<tr>
<td>Fenton &amp; Melton</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>Chen &amp; Lu</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>3</td>
</tr>
<tr>
<td>Offut, Harrold &amp; Kote</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>Dhama</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>4</td>
</tr>
</tbody>
</table>

Table 142. Comparisons between New and Recent Measures on their Sensitivity to Atomic Modifications
Chapter 6

Conclusion

Coupling is one of two attributes that has a great impact on software quality. A total of 8 out of 13 software quality factors depend on coupling and cohesion [Dham95]. In this thesis, the notion of coupling is introduced, the factors affecting it are discussed and its existing measures are surveyed. A new coupling measure is developed to capture most of coupling factors. The new coupling measure is based on a general model. It is validated empirically and on the effect of atomic change. This chapter contains research accomplishments, contributions and future work.

6.1 Accomplishments

The following accomplishments were listed in the previous sections and summarized as the followings:

- The new method is based on the combination of the last four Myers coupling classifications.
- It captures most of the factors affecting the coupling between subprograms in the traditional high level languages.
- The simple assignment of numbers to the elements contributing in coupling shown in table 5 is theoretically proved to be an ordinal scale because:
1. It captures the ranking between the elements that affect on coupling as evident in table 5.

2. It is a homorphism, i.e., knowing the assigned number enables to know the type of coupling the element holds and knowing the type of coupling an element captures enables to know the assigned number.

- This ordinal scale was extended by introducing a binary concatenation operation which was defined to be the sum of contribution of two elements. This operation is used to define the weight of two elements when a procedure is called using these elements. This extended method is the new method and it is shown to be a ratio scale.

- The new method is validated empirically on capturing the assumed relations on the procedures of six sample modules. It is found to capture all of those assumptions across the six modules. Moreover, the new method was compared with the four recent measures on preserving the implied relations. The new method is found to outperform the other measures because none of them satisfies all the relations.

- The new method is validated on preserving the implied relations when six atomic modifications are implemented on the last five modules. It is found to capture all of those assumptions across the six modifications. Moreover, the new method was compared with the four recent measures on preserving the implied relations. The new method is found to outperform the other measures because none of them satisfies one relations.

- The new method was compared with the four recent measures on their sensitivity to the six atomic relations. It was found to be sensitive to all of the atomic modifications. Dhama method was sensitive to four modifications where Chen and Lu method was sensitive only to three. Fenton and Melton, and Offut, Harrold, and Kotive measures were sensitive to two of these modifications.
6.2 Contributions

The thesis has the following contributions to the research:

- A general coupling metric that can be used in all imperative higher level languages was developed and validated.
- It is based on a general model that measures the coupling of a procedure to procedure, a set of procedures and system coupling.
- This metric requires the modules source but it can be used in the design using structure charts.

6.3 Future Works

The following future works may be surveyed:

- The new method can be extended to capture content coupling when such coupling exists in some programs in the same manner it was developed to capture Myers last four coupling classes.
- The new method can be extended by investigating the following factors on the coupling of an element:
  - The element size: Investigate if the size of the element has a contribution to its coupling.
  - The number of times the element is accessed as control or data or both: The new method differentiates between data and control only. When an element has both of these access types, it assumes it is a control access type. Do we need to capture the combination of access types?
  - The number of times the element is written or read or both: The new method does not
incorporate this factor. Do we need to capture this factor to give us better prediction of the coupling of procedures?

- The element access direction as input or output or both: The new method regards the effect of direction types the same. Investigate the impact of the element direction on the coupling.

- The new method could be automated for any higher level language like FORTRAN, PL/I, C, ADA and PASCAL. This automation allows the analysis of large systems to get a better picture of the new method performance.

- An appropriate clustering algorithm can be used to aid the design process by gathering the procedures that are intensively coupled in a subsystem, in such a way, to maximize the coupling of subsystem procedures and minimize the subsystem procedures to others not included in the subsystem.

- Abstract data types are looked at as a good encapsulation of an object. Some attempts was considered to extend Myers classes with abstract data types. Do we need to consider a special class for abstract data types or just treat it as an ordinary set of procedures with their non-local variables.

- The new method is a static method. Trying to use the new method dynamically could be investigated. The procedures and associated variables are known in advance from the compiler stage and their weights could be stored on a weight table for each procedure. When a procedure is called, its elements weights are added to the definition matrix until the system halts. The generated definition matrix captures the dynamic calling sequence and could be fed to the model to get the corresponding coupling matrix. Repeating this process on a number of different input datasets will enable the system analyst to compare the coupling matrices for each input dataset and discover the most used procedure. He probably might discover that a procedure is tightly coupled to others
dynamically. He might change the design to reduce such tight coupling.
Appendix A

This appendix contains six figures listing the source code of the six modules used in the new measure experimental Validation.

Figure 2

-- Package Specification
package Sort1 is
  type array_type is array (1..1000) of integer;

-- Procedure specification for global usage
procedure sort1 (n : in integer;
              to_be_sorted: in out array_type;
              a_or_d : in character);
end sort1;

-- Package implementation
package body sort1 is

  -- procedure sort1 private implementation
  procedure sort1 (n : in integer;
                   to_be_sorted: in out array_type;
                   a_or_d : in character) is
    location, temp : integer;
    begin
      for start in 1..n loop
        location := start;
        -- loop to get min and max each time
        for i in (start + 1)..n loop
          if a_or_d = 'd' then
            if to_be_sorted(i) > to_be_sorted(location) then
              location := i;
            end if;
          end if;
        end loop;
        to_be_sorted(location) := to_be_sorted(start);
        to_be_sorted(start) := location;
      end loop;
    end sort1;
elseif to_be_sorted(i) < to_be_sorted(location) then
    location := i;
end if;
end loop;

-- The exchange
    temp := to_be_sorted(start);
    to_be_sorted(start) := to_be_sorted(location);
    to_be_sorted(location) := temp;
end loop;
end sort1;
end sort1;

Figure 2. Module 1 Source Code
Figure 3

-- Package Specification
package Sort2 is

type array_type is array (1..1000) of integer;

-- Procedure specification for global usage
procedure sort2 (n: in integer;
    to_be_sorted: in out array_type;
    a_or_d : in character);
procedure find_max (n : in integer;
    start : in integer;
    to_be_sorted: in out array_type;
    location: in out integer);
procedure find_min (n : in integer;
    start : in integer;
    to_be_sorted: in out array_type;
    location: in out integer);
procedure exchange (start : in integer;
    to_be_sorted: in out array_type;
    location: in out integer);

end sort2;

-- Package implementation
package body sort2 is

-- procedures sort2 private implementation
procedure find_max (n : in integer;
    start : in integer;
    to_be_sorted: in out array_type;
    location: in out integer) is

begin
    location := start;
for i in (start + 1)..n loop
    if to_be_sorted(i) > to_be_sorted(location) then
        location := i;
    end if;
end loop;
end find_max;

procedure find_min (n : in integer;
    start : in integer;
    to_be_sorted: in out array_type;
    location: in out integer) is
begin
    location := start;
    for i in (start + 1)..n loop
        if to_be_sorted(i) < to_be_sorted(location) then
            location := i;
        end if;
    end loop;
end find_min;

procedure exchange (start : in integer;
    to_be_sorted: in out array_type;
    location: in out integer) is
    temp: integer;
begin
    temp := to_be_sorted(start);
    to_be_sorted(start) := to_be_sorted(location);
    to_be_sorted(location) := temp;
end exchange;

procedure sort2 (n : in integer;
    to_be_sorted: in out array_type;
    a_or_d : in character) is
location: integer;
begin
    for start in 1..n loop
        if a_or_d = 'd' then
            find_max (n, start, to_be_sorted, location);
        else
            find_min (n, start, to_be_sorted, location);
        end if;
    end loop;
end sort2;
end if;
exchange (start, to_be_sorted, location);
end loop;
end sort2;
end sort2;

---

*Figure 3. Module II Source Code*
Figure 4

C THIS PROGRAM CALLS RCHECK, AVCALC AND GTAVG PROCEDURES

INTEGER NUMEL, NABOVE, I
REAL X(50), AVG
CHARACTER *50 PROMPT
INTEGER KBIN, CRTOUT

REAL AVCALC
INTEGER GTAVG

PARAMETER (KBIN = 5, CRTOUT = 5)

PROMPT = 'ENTER NO. OF ARRAY ELEMENTS (1 - 50)'
CALL RCHECK (PROMPT, NUMEL, 1, 50)

WRITE(CRTOUT, *) 'ENTER', NUMEL, ' REAL NUMBERS, ONE PER LINE'
DO 10 I = 1, NUMEL
   READ(KBIN, *) X(I)
10 CONTINUE

AVG = AVCALC (X, NUMEL, 50)

NABOVE = GTAVG (X, AVG, NUMEL, 50)

WRITE (CRTOUT, *) 'AVERAGE = ', AVG
WRITE (CRTOUT, *) 'NUMBER ABOVE AVERAGE = ', NABOVE

STOP
END

SUBROUTINE RCHECK(PROMPT, INPUT, LOWBND, UPBND)
CHARACTER * (*) PROMPT
INTEGER INPUT, LOWBND, UPBND
LOGICAL OK
INTEGER KBIN, CRTOUT

PARAMETER (KBIN = 5, CRTOUT = 5)

OK = .FALSE.

10 IF (.NOT. OK) THEN
    WRITE(CRTOUT, *) 'INPUT OUT OF RANGE', LOWBND, ' TO ', UPBND
    ELSE
    ENDIF
    GO TO 10
ENDIF

RETURN
END

REAL FUNCTION AVCALC (A, N, NDIM)
INTEGER N, NDIM
REAL A(NDIM)

REAL SUM

REAL SUMCAL

SUM = SUMCAL (A, N, NDIM)
AVCALC = SUM / N

RETURN
END

REAL FUNCTION SUMCAL(A, N, NDIM)
INTEGER N, NDIM
REAL A(NDIM)

INTEGER I
SUMCAL = 0
DO 10 I = 1, N
   SUMCAL = SUMCAL + A(I)
10 CONTINUE
RETURN
END

INTEGER FUNCTION GTAVG (A, AVG, N, NDIM)
INTEGER N, NDIM
REAL A(NDIM), AVG

INTEGER I

GTAVG = 0
DO 10 I = 1, N
   IF (A(I) .GT. AVG ) THEN
      GTAVG = GTAVG + 1
   ENDIF
10 CONTINUE
RETURN
END

Figure 4. Module III Source Code
Figure 5

/* DAYS - CALCULATE THE NUMBER OF DAYS BETWEEN SDATE AND EDATE */

DAYS: PROC (SDATE, EDATE, NDAY) OPTIONS(FORTRAN) REORDER;

DCL SDATE FIXED BIN(31),
      EDATE FIXED BIN(31),
      NDAY FIXED BIN(31),
      YS FIXED BIN(31) INIT(0),
      MS FIXED BIN(31) INIT(0),
      DS FIXED BIN(31) INIT(0),
      YE FIXED BIN(31) INIT(0),
      ME FIXED BIN(31) INIT(0),
      DE FIXED BIN(31) INIT(0),
      ERROR_CODE FIXED BIN(31) INIT(0),
      ERROR_CODE1 FIXED BIN(31) INIT(0),
      I FIXED BIN(31) INIT(0),
      DEND CHARACTER(8) INIT ("'"),
      HOLD CHARACTER(8) INIT ("'"),
      START CHARACTER(8) INIT ("'"),
      VALDAY EXTERNAL ENTRY (FIXED BIN(31))
      RETURNS (FIXED BIN(31));

DCL DAYS_IN_MONTH(12) FIXED BIN (31) INIT (
    31, 28, 31, 30, 31, 30, 31, 30, 31, 30, 31, 30 );

******************************************************************************
/* VERIFY THAT THE DATES ARE WITHIN ACCEPTABLE RANGE. */
******************************************************************************

ERROR_CODE = VALDAY(SDATE);
ERROR_CODE1 = VALDAY(EDATE);

IF (ERROR_CODE = 0 | ERROR_CODE1 = 0) THEN NDAY = -1;
ELSE DO;
    NDAY = 0;

/**************************************************************************/
/* IF START DATE IS GREATER THAN END DATE, THEN SWITCH THEM. */
/**************************************************************************/

PUT STRING (START) EDIT (SDATE) (P'999999999');
PUT STRING (DEND ) EDIT (EDATE) (P'999999999');

IF (SDATE > EDATE) THEN DO;
    HOLD = START;
    START = DEND;
    DEND = HOLD;
END;

GET STRING (START) EDIT (YS, MS, DS ) (F(4), (2)(F(2)));
GET STRING (DEND ) EDIT (YE, ME, DE ) (F(4), (2)(F(2)));

/**************************************************************************/
/* CALCULATE FOR DAYS IN THE SAME MONTH. */
/**************************************************************************/

SELECT;
    WHEN ((YS = YE) & (MS = ME)) NDAY = DE - DS;

/**************************************************************************/
/* CALCULATE FOR MONTHS IN THE SAME YEAR. */
/**************************************************************************/

WHEN ((YS = YE) & (MS < ME)) DO;
    NDAY = NDAY + DAYS_IN_MONTH(MS) - DS;

    IF ( LEAP (YS) & MS = 2) THEN NDAY = NDAY + 1;

    DO I = MS + 1 TO ME - 1;
        NDAY = NDAY + DAYS_IN_MONTH(I);
        IF (LEAP(YS) & I = 2) THEN NDAY = NDAY + 1;
    END;

    NDAY = NDAY + DE;
END;

/**************************************************************************/
/* CALCULATE FOR DIFFERENT YEARS. */

WHEN (YS < YE) DO;

NDAY = NDAY + DAYS_IN_MONTH(MS) - DS;

IF (LEAP(YS) & MS = 2) THEN NDAY = NDAY + 1;

DO I = MS + 1 TO 12;
   NDAY = NDAY + DAYS_IN_MONTH(I);
   IF (LEAP(YS) & I = 2) THEN NDAY = NDAY + 1;
END;

DO I = YS + 1 TO YE - 1;
   NDAY = NDAY + 365;
   IF (LEAP(I)) THEN NDAY = NDAY + 1;
END;

DO I = 1 TO ME - 1;
   NDAY = NDAY + DAYS_IN_MONTH(I);
   IF (LEAP(YE) & I = 2) THEN NDAY = NDAY + 1;
END;

/* CALCULATE TO THE TARGET DAY. */
NDAY = NDAY + DE;
END;

OTHERWISE;
END;
END;

 /**************************************************************************
 * LEAP - RETURNS A BIT VALUE TRUE IF A LEAP YEAR FALSE OTHERWISE  */
 */
 %PAGE;

 LEAP: PROC(Y) RETURNS(BIT(1));

 DCL Y FIXED BIN(31),
     LY BIT(1) INIT('0B'),
     MOD BUILTIN;

 IF (MOD(Y,4) ^= 0) THEN LY = '0B';
 ELSE DO;
     IF (MOD(Y,400) = 0 | MOD(Y,100) ^= 0) THEN LY = '1B';
 END;

 RETURN (LY);

 END LEAP;

 END DAYS;

/**************************************************************************
 * VALDAY - VALIDATE A DATE.  */
 */
 %PAGE;

 VALDAY: PROC (SDATE) RETURNS (FIXED BIN(31)) REORDER;

 DCL SDATE FIXED BIN(31),
     YS FIXED BIN(31) INIT(0),
     MS FIXED BIN(31) INIT(0),
     DS FIXED BIN(31) INIT(0),
     ERROR_CODE FIXED BIN(31) INIT(0),
     START CHARACTER(8) INIT(' '),
     ERR BIT(1) INIT('0B');

 DCL DAYS_IN_MONTH(12) FIXED BIN(31) INIT (31,28,31,30,31,30,
                                         31,30,31,30,31,31);
/***********************
/* VERIFY THAT THE DATES ARE WITHIN ACCEPTABLE RANGE.   */
***********************

IF (SDATE > 99991231 l SDATE < 17530101) THEN ERR = '1'B;

/*****************************
/* EXTRACT THE YEAR, MONTH AND DAY.          */
/*****************************/

IF (^ERR) THEN DO;
   PUT STRING (START) EDIT (SDATE) (P'999999999');
   GET STRING (START) EDIT (YS, MS, DS) (F(4), (2)(F(2)));
END;

/*****************************
/* VERIFY THAT THE MONTH IS VALID.          */
/*****************************/

IF (MS < 1 l MS > 12) THEN ERR = '1'B;
END;

/*****************************
/* VERIFY THAT THE DAY IN THE MONTH IS VALID.*/
/*****************************/

IF (^ERR) THEN DO;
   IF (MS = 2) THEN
      IF (LEAP (YS)) THEN
         IF (DS < 1 l DS > 29) THEN ERR = '1'B;
         ELSE;
            ELSE IF (DS < 1 l DS > 28) THEN ERR = '1'B;
            ELSE;
            ELSE IF (DS < 1 l DS > DAYS_IN_MONTH(MS)) THEN ERR = '1'B;
            END;
      END;
   IF (^ERR) THEN ERROR_CODE = 1;
   ELSE ERROR_CODE = 0;
   RETURN (ERROR_CODE);

/******************
/* LEAP - RETURNS A BIT VALUE TRUE IF A LEAP YEAR FALSE OTHERWISE */
******************/
%PAGE;
LEAP:PROC(Y) RETURNS(BIT(1));

DCL Y FIXED BIN(31),
    LY BIT (1) INIT ('0'B),
    MOD BUILTIN;

IF (MOD(Y,4) ^= 0) THEN LY = '0'B;
ELSE DO;
    IF (MOD(Y,400) = 0 | MOD(Y,100) ^= 0) THEN LY = '1'B;
END;

RETURN (LY);

END LEAP;

END VALDAY;

Figure 5. Module IV Source Code
Figure 6

PROGRAM CRAPS
(* PLAY CRAPS WITH THE USER *)
VAR
  AMOUNT, BET, SEED: REAL;
  WON, FINISHED: BOOLEAN;

FUNCTION RANDOM (N: INTEGER): INTEGER;
(* RETURNS PSEUDORANDOM INTEGER IN RANGE 1-N
  REFERS TO AND MODIFIES GLOBAL VARIABLE SEED *)
BEGIN
  SEED := SQR (SEED + 3.1415297);
  SEED := SEED - TRUNC (SEED);
  RANDOM := TRUNC (N * SEED);
END; (* RANDOM *)

PROCEDURE ROLLDICE (VAR VALUE: INTEGER); (* SIMULATE ROLLING OF TWO DICE *)
VAR
  DIE1, DIE2: INTEGER;
BEGIN
  DIE1 := RANDOM (6);
  DIE2 := RANDOM (6);
  VALUE := DIE1 + DIE2;
  WRITELN ('YOU ROLLED A', DIE1, ' AND A', DIE2);
END; (* ROLLDICE *)

PROCEDURE INITIALIZE (VAR SEED, AMOUNT: REAL); (* GET SEED FOR PSEUDORANDOM NUMBER GENERATOR AND AMOUNT USER HAS TO PLAY WITH *)
BEGIN
  WRITE ('ENTER NUMBER BETWEEN 0 AND 1: ');

READLN (SEED);
WRITE ('HOW MUCH MONEY DO YOU HAVE TO PLAY WITH? ');
READLN (AMOUNT);
WHILE AMOUNT <= 0.0 DO
  BEGIN
    WRITELN ('YOU CAN NOT PLAY WITH A ZERO OR NEGATIVE');
    WRITELN ('AMOUNT OF MONEY. PLEASE ENTER A POSITIVE');
    WRITE ('AMOUNT: ');
    READLN (AMOUNT);
  END
END; (* INITIALIZE *)

PROCEDURE ACCEPTBET (VAR BET, AMOUNT: RAIL);
(* FIND OUT HOW MUCH USER WANTS TO BET ON THIS GAME *)
BEGIN
  WRITE ('HOW MUCH DO YOU WANT TO BET ON THIS GAME? ');
  READLN (BET);
  WHILE (BET < 0.0) OR (BET > AMOUNT) DO
    BEGIN
      IF BET < 0.0 THEN
        WRITELN ('YOU CAN NOT BET A NEGATIVE AMOUNT');
      ELSE IF BET > AMOUNT THEN
        WRITELN ('YOU CAN NOT BET MORE THAN YOU HAVE');
      WRITE('PLEASE ENTER ANOTHER BET: ');
      READLN (BET);
    END
  END; (* ACCEPT BET *)

PROCEDURE PLAY (VAR WON: BOOLEAN);
(* PLAY ONE GAME *)

VAR
  VALUE, POINT : INTEGER;
BEGIN
  ROLLDICE (VALUE);
  IF (VALUE = 7) OR (VALUE = 11) THEN
    WON := TRUE;
  ELSE IF (VALUE = 2) OR (VALUE = 3) OR (VALUE = 12) THEN
    WON := FALSE;
  ELSE
      BEGIN
POINT := VALUE;
REPEAT
  ROLLDICE (VALUE);
  UNTIL (VALUE = POINT) OR (VALUE = 7);
  WON := VALUE = POINT (* BOOLEAN EXPRESSION *)
END
END; (* PLAY *)

PROCEDURE UPDATE (VAR BET, AMOUNT : REAL;
  VAR WON, FINISHED: BOOLEAN);
(* UPDATE PLAYER'S BANKROLL, DETERMINES WHETHER OR NOT PLAYER
  WANTS TO CONTINUE *)

VAR
  ANSWER : STRING;
BEGIN
  IF WON THEN
    BEGIN
      WRITELN ('CONGRADULATIONS, YOU WIN!');
      AMOUNT := AMOUNT + BET;
    END
  ELSE
    BEGIN
      WRITELN ('TOO BAD, YOU LOSE!');
      AMOUNT := AMOUNT - BET;
    END;
  WRITELN ('YOU NOW HAVE ', AMOUNT:8:2, ' DOLLARS');
  IF AMOUNT < 0.01 THEN
    FINISHED := TRUE;
  ELSE
    BEGIN
      WRITE ('DO YOU WANT TO PLAY ANOTHER GAME ’);
      WRITE ('(YES OR NO)? ’);
      READLN (ANSWER);
      FINISHED := ANSWER <> 'YES' (* BOOLEAN EXPRESSION *)
    END
END
END; (* UPDATE *)

BEGIN (* MAIN PROGRAM *)
INITIALIZE (SEED, AMOUNT);
REPEAT
  WRITELN;
  ACCEPTBET (BET, AMOUNT);
  PLAY (WON);
  UPDATE (BET, AMOUNT, WON, FINISHED)
UNTIL FINISHED;
WRITELN ('I ENJOYED PLAYING WITH YOU');
WRITELN ('LET'S PLAY AGAIN SOON')
END.

Figure 6. Module V Source Code
/* This program counts document statistics in a file of text * /
   and produce them in a file */

#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <ctype.h>

/* MACRO DEFINITIONS */

#define CHECK_REC_IO(a, b, c) if(rc != sizeof((a)) - 1)\
   {
      perror(strcat((b), " failed"));\
      exit((c));\
   }

#define CHECKIO(a, b, c) if (ferror((a)))\
   {
      perror(strcat((b), " failed ");\
      exit((c));\
   }

/* FILE POINTERS */

FILE *indoc;
FILE *outstat;
FILE *outdoc;

/* CHARACTER VARIABLES */

char ioarea[81];
char blankline[81];
char alpha[26] = 'abcdefghijklmnopqrstuvwxyz';

struct
{  
  char ctclas;
  char title[34];
  char count[9];
  char filler[17];
  } statline = {'1', {'""'}, {0}, {'""'}};

/* SIGEND INTEGER VARIABLES */

signed int result = 0;
signed int rc = 0;
signed int lines = 0;
signed int letters = 0;
signed int chars = 0;
signed int sentences = 0;
signed int phrases = 0;
signed int words = 0;
signed int paras = 0;
signed int blines_ins = 0;
signed int word_len = 0;

signed int alpha_ct[26] = {0};

/* FUNCTIONS */

void open_files(void);
void initialize(void);
void readdoc(void);
void writedoc(char);
void do_stats(void);
void write_stats(void);
void close_files(void);

main()
{
  register int i = 0, j = 0;
  register char this_char = 0, smtce_flag = 1, blank_line = 0;

  printf("Start of Processing \n\n");

  initialize();
  open_files();
  readdoc();
  while (!feof(indoc))

{
    lines++;  
    if ((i = strspn(ioarea, " "))) == 80)  
    blank_line = 1;  
else  
{
    if (i > 1 && sntnce_flag)  
        {  
            paras++;  
            if (blank_line == 0)  
                {  
                    writedoc(2);  
                    blines_ins++;  
                }  
            blank_line = 0;  
        }  
    for ( ; i < 80; i++)  
    {  
        this_char = *(ioarea + i);  
        if (this_char != ' ')  
            {  
                if (isalpha(this_char))  
                    {  
                        if (isupper(this_char))  
                            this_char = *(ioarea + 1) = tolower(this_char);  
                        letters++;  
                        alpha_ct[strlen(alpha, this_char) - alpha]++;  
                    }  
                if (i == 79 || *(ioarea + i + 1) == ' ')  
                    {  
                        words++;  
                        switch(this_char)  
                            {  
                                case ":":  
                                case ":":  
                                case "?": sentences++;  
                                    sntnce_flag = 1;  
                                case ":":  
                                case ":":  
                                case ":": phrases++;  
                            }  
            }  
        }
}
void do_stats()
{
    register int i = 0;
    float avwordsz = 0;

    if (letters == 0)
    {
        return;
    }

    avwordsz = (float)(chars - phrases) / (float)words;
    printf(statline.title, "%s", "Document Analysis Figures");
    writestat();
    statline.cflags = '0';
    printf(statline.title, "%s", "NUMEBR OF LINES ...............");
    printf(statline.count, "%5d", lines);
    writestat();
    printf(statline.title, "%s", "NUMEBR OF LETTERS............");
    printf(statline.count, "%5d", letters);
    writestat();
    printf(statline.title, "%s", "NUMEBR OF CHARACTERS...........");
    printf(statline.count, "%5d", chars);
    writestat();
    printf(statline.title, "%s", "NUMEBR OF PHRASES............");
    printf(statline.count, "%5d", phrases);
    writestat();
    printf(statline.title, "%s", "NUMEBR OF SENTENCES............");
    printf(statline.count, "%5d", sentences);
    writestat();
    printf(statline.title, "%s", "NUMEBR OF PARAGRAPHS............");
    printf(statline.count, "%5d", lparas);
    writestat();
    printf(statline.title, "%s", "NUMEBR OF BLANK LINES INSERTED...");
    printf(statline.count, "%5d", blines_ins);
    writestat();
    printf(statline.title, "%s", "AVERAGE WORD SIZE.............");
}
sprintf(statline.count, "%5d", avwordsz);
writestat();

for (i = 0; i < 26; i++)
{
    sprintf(statline.title, "%s", "% of letter " is ..
    *(statline.title + 13) = alpha[i];
    sprintf(statline.count, "%8.2f",
        (float) alpha_ct[i] / (float) letters * 100);
    writestat();
    statline.ctlasa = ' ';
}

void initialize()
{
    memset(blankline, '*', sizeof(blankline) - 1);
    memset(statline.filler, ' ', sizeof(statline.filler) - 1);
}

void open_files()
{
    if ((indoc = fopen("dd:INDOC", "rb"))) == NULL)
        CHECKIO(indoc, "fopen of INDOC", 60);

    if ((outdoc = fopen("dd:OUTDOC", "rb"))) == NULL)
        CHECKIO(outdoc, "fopen of OUTDOC", 70);

    if ((outstat = fopen("dd:OUTSTAT", "rb"))) == NULL)
        CHECKIO(outstat, "fopen of OUTSTAT", 60);
}

void readdoc()
{
    rc = fread (ioarea, 1, sizeof(ioarea) - 1, indoc);
    if (!feof(indoc))
        CHECKIO(ioarea, "fread of INDOC", 20);
}

void writedoc(register char io_num)
{
  if (io_num == 1)
  {
    rc = fwrite(ioarea, 1, sizeof(ioarea) - 1, outdoc);
    CHECK_REC_IO(ioarea, "fwrite of OUTDOC", 30);
    memset(ioarea, '0', sizeof(ioarea) - 1);
  }
  else
  {
    rc = fwrite(blankline, 1, sizeof(blankline) - 1, outdoc);
    CHECK_REC_IO(blankline, "fwrite of OUTDOC");
  }
}

void writestat()
{
  rc = fwrite(&statline, 1, sizeof(statline) - 1, outstat);
  CHECK_REC_IO(statline, "fwrite of OUTSTAT", 50);
}

void close_files()
{
  if (fclose(indoc))
    CHECKIO(indoc, "fclose of INDOC", 90);
  if (fclose(outdoc))
    CHECKIO(outdoc, "fclose of OUTDOC", 100);

  if (fclose(outstat))
    CHECKIO(outstat, "fclose of OUTSTAT", 110);
}

---------------------------------------------------------

Figure 7. Module VI Source Code
References


[Hutc85] D. H. Hutches, and V. R. Basili, System Structure Analysis: Clustering with Data


[Pick95] M. M. Pickard, and B. D. Carter, A Field Study of the relationship of Information Flow and


