Evaluation of reliability of object-oriented systems based on Cohesion and Coupling Fuzzy computing

Samira Kalantari, Masoomeh Alizadeh, Homayoun Motameni
Department of Computer Engineering, Islamic Azad University, Sari, Iran
Kalantarisamira@yahoo.com; Alizadeh_masoomeh@yahoo.com; Motameni@iausari.ac.ir

Keywords: Coupling, Cohesion, Non-Functional Parameters, Complexity, Reliability, Fuzzy logic

1. Introduction

By increasing the use of software in different areas, the demand for software with high reliability has been increased as well. Reliability is one of the most important parameters of software quality to produce the reliable and safe systems. So evaluating reliability software systems after production and before entering the market is an important issue. Reliability means the system, in a specified period of time and in a specific environment, has been worked without failing [1].

Failures caused by many reasons such as defect in design, unpredictable input, failing in specification stage of software and etc. [2]. Many errors that lead to failure are due to complexity. And the complex and imperfect design that is an important error, has been caused by complexity. So, complexity reduces the reliability [2-3]. Therefore, software designing is an efficient factor to determine software quality [4], so that a good design leads to produce software that has such features as: increasing readability, increasing reusability, easy maintenance, control of data access [5-7].
The software design is a basic part of software production and a good design reduces complexity. By designs based on divide and conquer, complexity will be reduced by dividing a problem to smaller problems [7-8].

Design focuses on four main sections: data, architecture, modular, and interfaces. The concept of modular, has taken attention for almost five decades, is not only acceptable in software engineering but is also used in all engineering disciplines [8]. In fact, the software divided in to components that called modules [4,8]. There are three significant concepts in modular design: 1. Functional independence, 2. coupling, and 3. cohesion. Functional independence seeks out a software design in which every modular has particular task and maintains its independence with abstraction and hiding information to other modular [7]. But to evaluate a software solution achieving to the best set of modular is important. As you can see in figure 1, if the number of modular be more or less, the production cost will be increased [7].

![Figure 1. The effect of number of modular on production cost [7]](image)

In modularity, coupling and cohesion are the main concepts. Cohesion measures the operational power of a module and coupling is an operational power or is dependence between two or more modules [4]. Authors in [9] have proposed a hybrid fuzzy model for the prediction of fault proneness in object oriented design based on coupling and cohesion metrics and other relationships between modules. They used 3 metrics of CK and 6 metrics of MOOD in Mamdani and Sugeno Fuzzy controller, respectively.

There are various relations among modules, each of which will lead to dependency. Some researchers computed coupling and cohesion generally and did not consider any type of dependencies [10], while some others have studied and computed different types of dependency [5,11]. Researchers, for example in [11], to compute measures of coupling and cohesion, have applied variety dependencies among components of a service-oriented system.

Since by considering any type of dependencies among modules, computing of coupling and cohesion, measurement accuracy can be increased, in this paper an approach based on fuzzy computing is presented in which all dependencies have been considered. Thus, assessing the reliability and error rate that occur in the system will be calculated more accurately. Also, the approach can shed more light on software
engineering to calculate other Non-Functional parameters such as security, readability, and reusability.

The rest of this paper is organized as follows: In section 2, we explained concept of coupling and cohesion will be explained, and to compute them some approaches will be studied. In section 3, concept of Fuzzy Logic will be introduced. To compute the coupling, cohesion and reliability, in section 4, a new method is proposed. In section 5, applying the proposed method, the relations between coupling and cohesion in object-oriented systems will be analyzed and finally a brief conclusion will be discussed as well.

2. Literature

In this section, first the concepts of coupling and cohesion will be identified, and then the same as related works, to measure these two factors, some methods will be presented.

2.1 Concept of coupling and cohesion:

As mentioned before, cohesion is a factor to measure the operational power of a module; in other words, it can specify the dependency of the internal components of a module and interrelations of modules. The relation of internal components is based on similar features in modules [5-7]. Coupling is a factor to measure dependencies between modules; In other words, it can identify the amount of connections and relations between modules [5,7,8,11]. According to the meaning of coupling and cohesion in modularization, it is clear that their designing is through divide and conquer design. Therefore, the level of complexity is reduced. Their design can also be implemented in languages that are supported by divide and conquer technique [7].

A decade tedious efforts led to a classification of coupling and cohesion that finally Myers presents the following classification [12]:

![Figure 2. a: Coupling classification, b: Cohesion classification](image)

According to the purpose of modularization and achieving the functional independency for decreasing complexity, Software designers looked for modular design with low coupling and high cohesion [7,13].

There are many methods to measure these two parameters. Here, according to table 1 measuring of coupling and cohesion in object-oriented designing is studied [14].
Table 1. Modules and internal components of modules at object-oriented designing

<table>
<thead>
<tr>
<th>Internal components of modules</th>
<th>Modules</th>
</tr>
</thead>
<tbody>
<tr>
<td>Variables and Methods</td>
<td>Class</td>
</tr>
</tbody>
</table>

2.2 Measuring methods of coupling and cohesion

CBO, RC, CF, DAC are Metrics for Object-Oriented Design (MOOD) coupling's metrics that Researchers in [9,15,16] summarized them briefly. CBO (Coupling Between Object classes) shows that how many classes couple together and identify the amount of coupling. RFC (Response For Class) as a response set, includes all methods in a class and other calling classes from the same class (in fact, we'll consider concept of coupling, when a class will be available in another class, in another word, at least some parts of this class to be called by another one). CF (Coupling Factor) to be calculated by equation 1 and equals the number of coupled class pairs, divided by the total number of class pairs. DAC (Data Abstraction Coupling) equals the number of class features (methods and variables) that other classes used as their data.

\[
CF = \frac{\text{number of coupled class pairs}}{\text{total number of class pairs}}
\]

LCOM, LCOM3, RLCOM and TCC are metrics that calculate cohesion [6,9,15,16]. LCOM (Lack of Cohesion in Methods) equals the number of non-similar method pairs in a class (similarity of the methods mean that all methods in a class have one or more similar and common features). LCOM3 is the number of component connections in a graph that use graph model, so that the vertex is the methods and the edge is the link between methods. According to equation 2 and 3, RLCOM and TCC (Tight class cohesion) are number of non-similar method pairs and number of similar method pairs on total number of method pairs in the class, respectively.

\[
RLCOM = \frac{\text{number of non-similar method pairs}}{\text{total number of method pairs in the class}}
\]

\[
TCC = \frac{\text{number of similar method pairs}}{\text{total number of method pairs in the class}}
\]

In [5] Felton and Melton, to measure coupling, present a metric (equation 4) that uses Myers classification. Where N is the number of connection of x and y, and i is equivalent to surface of Myers model (figure 2a).

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

G. Gui proposed a metric in which to measure coupling, equation GroupD is used (equation 5) [10].

\[
coupD(i,j) = \frac{|MV_{j,i}|}{|MV_{j}|+|V_{j}|+|M_{j}|}
\]

In this equation, M is the method and V is equivalent to variables. MV is the set of methods and variables that was called by C class. MV is the set of methods and variables in C class and will be called by C class (MV=UMV.). And since the
denominator is equivalent to \(|MV_i| = |V_i| + |M_i|\), the amount of coupling is calculated in range of \(0 \leq coupD(i, j) \leq 1\). In the other words, this Standardization caused Group D to be independent of class size. We also use equation 6 to measure coupling indirectly.

\[
coupT(i, j, p) = \prod_{e_{s,t} \in P} coupD(s, t) = \prod_{e_{s,t} \in P} \frac{|MV_{s,t}|}{|MV_s|+|V_s|+|M_s|} \tag{6}
\]

When a transitive relation established between modules, indirect coupling occurs in the case that several coupling be in indirect paths. According to equation 6, a path with the most coupling is used to calculate coupling (equation 7).

\[
coup(i, j) = coupT(i, j, p_{max}(i, j)) \tag{7}
\]

Thus, to calculate coupling in a system (Total Coupling), the equations 8 and 9 are used, where \(m\) is the number of classes in system.

\[
WT_{coup} = \frac{\sum_{i=1}^{m} coup(i, j)}{m^2 - m} \tag{8}
\]

\[
WT_{coup} = \frac{\sum_{i=1}^{m} coupD(i, j)}{m^2 - m} \tag{9}
\]

G. Gui in [10] also proposed a method to measure Cohesion (equation 10) in which \(V_i \cap V_j\) are similar variables in i and j methods.

\[
simD(i, j) = \frac{|V_i \cap V_j|}{|V_i \cup V_j|} \tag{10}
\]

So, Gui, to calculate indirectly cohesion, presents equations 11 and 12.

\[
simT(i, j, p) = \prod_{e_{s,t} \in P} simD(s, t) = \prod_{e_{s,t} \in P} \frac{|V_s \cap V_t|}{|V_s \cup V_t|} \tag{11}
\]

\[
sim(i, j) = simT(i, j, p_{max}(i, j)) \tag{12}
\]

Considering \(m\) and \(n\), i.e, number of methods and number of class respectively, equations 13 and 14 show cohesion directly and indirectly in the class, respectively as well and equations 15 and 16 show cohesion in the system directly and indirectly, respectively.

\[
classCohT = \frac{\sum_{i,j=1}^{m} sim(i, j)}{m^2 - m} \tag{13}
\]

\[
classCohD = \frac{\sum_{i,j=1}^{m} simD(i, j)}{m^2 - m} \tag{14}
\]

\[
WTcoh = \frac{\sum_{j=1}^{n} classCohT_j}{n} \tag{15}
\]

\[
WICoh = \frac{\sum_{j=1}^{n} classCohD_j}{n} \tag{16}
\]
In [14] a method is proposed that by using calculation of two matrixes amount of coupling is identified. First matrix use data of elements of each component and do not consider connections among them (Table 2).

**Table 2. Structure of first matrix to calculate coupling by method [14]**

<table>
<thead>
<tr>
<th>Component/Element</th>
<th>E₁</th>
<th>E₂</th>
<th>...</th>
<th>Eₙ</th>
</tr>
</thead>
<tbody>
<tr>
<td>C₁</td>
<td>d₁₁</td>
<td>d₁₂</td>
<td></td>
<td>d₁ₙ</td>
</tr>
<tr>
<td>C₂</td>
<td>d₂₁</td>
<td>d₂₂</td>
<td></td>
<td>d₂ₙ</td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Cₘ</td>
<td>dₘ₁</td>
<td>dₘ₂</td>
<td></td>
<td>dₘₙ</td>
</tr>
</tbody>
</table>

Then, second matrix uses first matrix and is determined by amount of coupling between software components that is calculated through equation 17 (Table 3).

**Table 3. The structure of second matrix to calculate coupling by method [14]**

<table>
<thead>
<tr>
<th>Component</th>
<th>C₁</th>
<th>C₂</th>
<th>...</th>
<th>Cₘ</th>
</tr>
</thead>
<tbody>
<tr>
<td>C₁</td>
<td>C₁₁</td>
<td>C₁₂</td>
<td></td>
<td>C₁ₘ</td>
</tr>
<tr>
<td>C₂</td>
<td>C₂₁</td>
<td>C₂₂</td>
<td></td>
<td>C₂ₘ</td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Cₘ</td>
<td>Cₘ₁</td>
<td>Cₘ₂</td>
<td></td>
<td>Cₘₘ</td>
</tr>
</tbody>
</table>

Where \( \beta_k \) and \( \alpha_i \) are determined by:

\[
C_{ij} = \left( \sum_{k=1}^{n} d_{ik} \times d_{jk} \times \beta_k \right) / \alpha_i
\]

(17)

\[
\alpha_i = \sum_{j=1}^{n} d_{ij} \quad \text{and} \quad \beta_k = 1 / \sum_{j=1}^{m} d_{ik}
\]

(18)

3. Fuzzy Logic

The concept of fuzzy logic was first introduced by Dr. Lotfi zadeh; the Iranian Professor from California University in Brookline. And it was not only a control methodology but as a way of processing data on the basis of authorizing membership in small groups rather than membership in cluster groups [17].

The fuzzy logic is a simple rule based on IF x AND y THEN z. Fuzzy mathematics is Meta sets from Boolean logic that refers to relative accuracy. Although fuzzy systems describe uncertain and indefinite phenomena, the fuzzy theory is still a precise theory. A fuzzy system includes the following structure [18]:

- **Fuzzification**: making something fuzzy.
- **Fuzzy rule base**: in the rule base, the if-then rules are fuzzy rules.
- **Fuzzy inference engine**: produces a map of the fuzzy set in the space entering the fuzzy set and in the space leaving the fuzzy set, according to the rules if-then.
- **Defuzzification**: making something non-fuzzy

![Figure 3. The structure of fuzzy system](image-url)
In the paper, we use fuzzy logic as a method to measure amount of coupling and cohesion between components of software systems.

4. Proposed Method

The proposed method follows a sequential process that is carried out according to figure 4 and this algorithm:

1. Determined type of coupling in software system according to Myers model (Data, Stamp, Control, Common and Content).
2. Computing type of Myers model coupling according to:
   \[
   \text{coupling} = \frac{\sum_{i,j=1}^{\infty} \text{coup}(i,j)}{m^2 - m} = \frac{\sum_{i,j=1}^{\infty} A_{ij}}{m^2 - m}
   \]
3. Computing total coupling according to fuzzy logic.
4. Determined type of cohesion in software system according to Myers model (Logical, Temporal, Procedural, Communicational, Sequential and Functional)
5. Computing type of Myers model cohesion according to:
   \[
   \text{class}_{coh} = \frac{\sum_{i,j=1}^{\infty} \text{sim}(i,j)}{m^2 - m} = \frac{\sum_{i,j=1}^{\infty} A_{ij}}{m^2 - m} \quad \text{&} \quad \text{cohesion} = \frac{\sum_{i,j=1}^{\infty} \text{class}_{coh}}{n}
   \]
6. Computing total cohesion according to fuzzy logic
7. Evaluated system reliability according to total coupling and total cohesion.

4.1. Determining amount of coupling

As mentioned before, computing coupling is done based on communication between modules. As required low coupling to design software with high quality and to reduce complexity, to establish this communication as more accurate measurement is necessary.

There are several couplings in the software system. If we consider and compute every coupling, coupling can be computed more accurate and this means more detail is
studied. In this paper, to compute coupling Myers classification is used. In this classification, coupling includes: Content, Common, Control, Stamp, and Data. All will be defined briefly [13,19]:

- **Content coupling**: content coupling occurs when modules could correct and change internal data directly. In fact, a module refers to another module and can be accessed without restriction.

- **Common coupling**: coupling communicates each other through data sharing. In fact, modules only access to common data or common blocks (global data can be addressed and accessed).

- **Control coupling**: in this method, a module sends a control variable to another module and in this way, they can communicate each other. In fact, it is done by setting a control flag (or control variable) (a module sends a request to another module through the control flag).

- **Stamp coupling**: a variable type of record as a parameter is sent by the first module and second module used only a subset of that record. A module sends data more than the other module’s need and only some part of that data will be used.

- **Data coupling**: a module sends data to another one considering its needs. All data must be homogeneous.

The computing coupling methods in [10] are general and without considering the type of coupling. The proposed method in this paper computes any type of coupling through changing parameters in equation 19.

$$\text{coupling} = \frac{\sum_{i=1}^{m} \text{Coup}(i,j)}{m^2 - m} = \frac{\sum_{i=1}^{m} \frac{A_{ji}}{B_{ji}}}{m^2 - m} \quad (19)$$

As equation 8 is in the range of 0 to 1, proposed equation is also standard and is the range 0 (Zero) to 1 (One).

<table>
<thead>
<tr>
<th>Table 4. Used parameters in coupling formula</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
</tr>
<tr>
<td>Content coupling (j,i)</td>
</tr>
<tr>
<td>Common coupling (j,i)</td>
</tr>
<tr>
<td>Control coupling (j,i)</td>
</tr>
<tr>
<td>Stamp coupling (j,i)</td>
</tr>
<tr>
<td>Data coupling (j,i)</td>
</tr>
</tbody>
</table>

By classifying input and output parameters, total coupling is obtained by fuzzy structure (table 5).
Table 5. Coupling is computed in the system through data fuzzification

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Type</th>
<th>Data</th>
<th>Linguistic variables</th>
</tr>
</thead>
<tbody>
<tr>
<td>Content</td>
<td>Input</td>
<td>[0 1]</td>
<td>low, medium, high</td>
</tr>
<tr>
<td>Common</td>
<td>Input</td>
<td>[0 1]</td>
<td>low, medium, high</td>
</tr>
<tr>
<td>Control</td>
<td>Input</td>
<td>[0 1]</td>
<td>low, medium, high</td>
</tr>
<tr>
<td>Stamp</td>
<td>Input</td>
<td>[0 1]</td>
<td>low, medium, high</td>
</tr>
<tr>
<td>Data</td>
<td>Input</td>
<td>[0 1]</td>
<td>low, medium, high</td>
</tr>
<tr>
<td>Total</td>
<td>Output</td>
<td>[0 1]</td>
<td>v-low, medium low, high, v-high</td>
</tr>
</tbody>
</table>

The used fuzzy inference system is fuzzy Mamdani. It is a simple rule base method that can be computed through if….then…. and without complex computing.

Input data (Table 1) entered the fuzzy system and produced considered output through fuzzy inference (Mamdani inference). The inference is of a set of rules that embedded in the fuzzy system. Figure 5 and 6 show the graphs of membership function of inputs and outputs in this computing system.

![Figure 5. Membership function of input parameters](image1)

![Figure 6. Total coupling output membership function](image2)

### 4.2. Determining cohesion

Since cohesion is defined based on internal communication between modules, there are many ways to be computed. According to coupling classification, Myers presents a different classification for cohesion. To offer a more accurate method this classification is used.

Myers presented cohesion classification as Logical, Temporal, Procedural, Communicational, Sequential and Functional [12].

- **Logical Cohesion**: in logical cohesion, elements that have the same operation are considered as the same method.
- **Temporal Cohesion**: in temporal cohesion, elements that have the same operation are placed in a method and performed simultaneously.
- **Procedural cohesion**: in procedural cohesion, elements from a method connected together through some control flows.
- **Communicational cohesion**: the elements that have procedural cohesion and work on the same data set can be placed in communicational cohesion.
- **Sequential cohesion**: the elements from a method that have communicational cohesion and are connected together by regular control flow have sequential cohesion.
- **Functional cohesion**: the elements of a method that have a sequential cohesion and have the same work or the same goal have functional cohesion.

In previous methods since the types of cohesion were ignored and cohesion was computed in a system generally, it is tried that according to computing coupling, the formula \[10\] to be expanded and all types of cohesion on finally cohesion to be considered.

\[ class_{coh} = \frac{\sum_{i,j=1}^{m} sim(i,j)}{m^2 - m} = \frac{\sum_{i,j=1}^{m} A_{ij}}{m^2 - m} \]  

\[ cohesion = \frac{\sum_{i=1}^{n} class_{coh}}{n} \]  

<table>
<thead>
<tr>
<th>Type of cohesion</th>
<th>( A_i )</th>
<th>( B_i )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Logical cohesion</td>
<td>Variables from ( i,j ) that have the same function</td>
<td>Total number of variable ( i )</td>
</tr>
<tr>
<td>Temporal cohesion</td>
<td>Variables from ( i,j ) that have the same function and perform simultaneously</td>
<td>The number of variable ( i ) that have the same function with ( j )</td>
</tr>
<tr>
<td>Procedural cohesion</td>
<td>Variables from ( i,j ) that connect with a control variable</td>
<td>The numbers of ( i ) and ( j ) that connected together</td>
</tr>
<tr>
<td>Communicational cohesion</td>
<td>Variables from ( i ) that is connected with ( j ) through a control variable and worked on the same data set.</td>
<td>Variables from ( i ) that connected with ( j ) through a control variable</td>
</tr>
<tr>
<td>Sequential cohesion</td>
<td>Variables from ( i ) that is connected with ( j ) through a control variable and worked on the same data set and connected together by a regular control flow.</td>
<td>Variables from ( i ) that connected with ( j ) through a control variable and worked on the same data set</td>
</tr>
<tr>
<td>Functional Cohesion</td>
<td>Variables from ( i ) that connected with ( j ) through a control variable and worked on the same data set and connected together by a regular control flow and do the same work.</td>
<td>Variables from ( i ) that connected with ( j ) through a control variable and worked on the same data set and connect together by a regular control flow</td>
</tr>
</tbody>
</table>

| Table 6. Cohesion parameters |

By classifying input and output parameters, Total cohesion is obtained by fuzzy structure (Table 7)
Table 7. Cohesion is computed in the software system through data fuzzification

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Type</th>
<th>Data</th>
<th>Linguistic variables</th>
</tr>
</thead>
<tbody>
<tr>
<td>Logical</td>
<td>Input</td>
<td>[0 1]</td>
<td>Low, medium, high</td>
</tr>
<tr>
<td>Temporal</td>
<td>Input</td>
<td>[0 1]</td>
<td>Low, medium, high</td>
</tr>
<tr>
<td>Procedural</td>
<td>Input</td>
<td>[0 1]</td>
<td>Low, medium, high</td>
</tr>
<tr>
<td>Communicational</td>
<td>Input</td>
<td>[0 1]</td>
<td>Low, medium, high</td>
</tr>
<tr>
<td>Sequential</td>
<td>Input</td>
<td>[0 1]</td>
<td>Low, medium, high</td>
</tr>
<tr>
<td>Functional</td>
<td>Input</td>
<td>[0 1]</td>
<td>Low, medium, high</td>
</tr>
<tr>
<td>Total cohesion</td>
<td>Output</td>
<td>[0 1]</td>
<td>v-low, low, medium, high, v-high</td>
</tr>
</tbody>
</table>

By fuzzy computing of coupling, to compute cohesion the Mamdani is used by fuzzy computing of coupling. The data that has been identified in table 5, is placed in the fuzzy system and the outputs is specified through Mamdani inference. In figures 5 and 6, the graphs of membership function for cohesion inputs and an output according to coupling method are presented.

4.3. Evaluating reliability in object-oriented systems

Reliable software is required for some functions. Researches show that high complexity is a factor that reduces reliability and makes the system difficult to be realized. If the software system has high cohesion and low coupling, the complexity will be reduced. Thus, the risk of errors will be reduced and the reliability will be increased. As you can see in table 8, cohesion has an inverse relationship with complexity and direct relationship with reliability.

Table 8. Cohesion relation with complexity and reliability [6]

<table>
<thead>
<tr>
<th>Complexity</th>
<th>Reliability</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cohesion</td>
<td>+ve</td>
</tr>
</tbody>
</table>

If the final cohesion and coupling resulting from fuzzy system output have been called Totalcoh and Total_coup, respectively, the reliability of a software system can be computed through table 9 and equation 22.

Table 9. Identifying output of fuzzy systems

<table>
<thead>
<tr>
<th>Fuzzy output</th>
<th>Defuzzifier</th>
<th>Fuzzy output crisp</th>
<th>Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>Totalcoup</td>
<td>Centeroid</td>
<td>T_coup</td>
<td>[0-1]</td>
</tr>
<tr>
<td>Totalcoh</td>
<td>Centeroid</td>
<td>T_coh</td>
<td>[0-1]</td>
</tr>
</tbody>
</table>

\[ R \propto \alpha \times \frac{(T_{coh})}{\beta \times (T_{coup})} \] (22)

Where R is the system reliability, and f=1-R is the system failure rate. \( \alpha \) and \( \beta \) (0 ≤ \( \alpha \), \( \beta \) ≤ 1) the importance of coupling and cohesion in the systems can be specified as an effective factor in terms of different software systems. Thus reliability of software design is computed as follow:
\[ R_{design} = \frac{TotalCohesion + (1-TotalCoupling)}{2} \]

This relationship indicates that whatever Coupling to be lower and Cohesion to be higher, the reliability of the software design will be higher and vice versa.

5. Case Study

The case study in this paper is insurer's policy for people with disabilities [20]. The system has several classes that the relations among them have been shown in UML graph (figure 7). The system's classes include: The Customer class, which represents the information about disabled people. Medical-Officer class that includes the information of medical staff. Main-Branch class is an organization to manage information about disabled people and has several branches. Mobile-System class is a system to transport disabled people. Disability class is used to determine disability. The other classes include New_Business, Bank and Plan that are inherited relation that study the jobs, banking and insurance's pension, respectively.

In terms of Myers model, the relation among different classes can be interpreted as follows: as can be seen in figure 7, the inherited classes from Plan have a type of complete relationship of methods and variables that lead to content coupling. Branch_New class handled acceptance or rejection of disability by using of control variable and using information that certificated by disability class. This process can lead to control coupling. In figure 8, the examples of relationship between methods and variables in the class that represent cohesion are shown.
Figure 7. Coupling relations in the case study system

Figure 8. Cohesion relations in the case study system
6. Conclusion

Reliability is one of the significant parameters of software quality that has an inverse relationship with complexity. To design software, coupling and cohesion are factors that can reduce the software complexity by correct selection. Thus, to compute more accurate, fuzzy computing of coupling and cohesion is used. It will be obtained through computing all types of coupling and cohesion in Myers Model. Finally, reliability of the system is evaluated based on computed coupling and cohesion by fuzzy model. If coupling is being high and cohesion is being low, the failure rate will be decreased and reliability will be increased.

References


