Evaluating Data Flow Diagrams

by

Qian Huang

B.S. Shanghai University of Technology, 1982

A MASTER'S THESIS

submitted in partial fulfillment of the

requirements for the degree

MASTER OF SCIENCE

Department of Computing and Information Sciences

KANSAS STATE UNIVERSITY

Manhattan, Kansas

1989

Approved by:

David A. Gustafson

Dr. David A. Gustafson
Major Professor

Table of Contents

Acknowlegements

It would still be an unrealistic dream for me to get my M.S. degree if now were 10 years ago. However, time changes and everything is possible once we try. I did with the help of many kind people.

I want to express my sincere and special thanks to my major professor Dr. Gustafson. The academic advice, help, and guidance are not the only things he gave at the regular meetings every week, the encouragement and support of free thinking from him provide me with another important way to learn. I will always remember that in my future academic work.

I would also like to thank my committee members, Dr. Unger and Dr. Lee, not only for the help and guidance to me as the members in the committee but also for the precious knowledge I learned from them at Kansas State University.

Thanks also go to my friends, Richard Courtney and Eric J. Byrne. Many discussions with you have assisted my thesis work.

I appreciate the people who participated in the survey of this research and thank them for the cooperation.

Finally, I want to tell my family in China : thank you. Even though they are at the other side of the world, I know they can hear me.

Chapter 1

Introduction

In the past few decades, the computer industry has been revolutionized by a number of new philosophies and techniques. Some of them have been successful and some of them are still in the stage of their developments. Software engineering is one of the later.

Software engineering is the application of a disciplined, systematic, quantifiable approach to the development, operation, and maintenance of software [STAND88]. The purpose of software engineering is to improve not only the productivity but also the reliability, maintainability, and the controllability of the software and software design process.

In order to achieve this goal, various kinds of methodologies and principles in software design have been proposed since 1960s, such as structured programming design, top-down design and bottom-up design. Different methodologies use different representations: hierarchy diagrams, flowcharts, structure charts, or data flow diagrams. These tools were developed from different perspectives of software design to catch different characteristics in software design. For example, hierarchy diagrams are used to describe the control structure of the software system while data flow diagrams are used to show how the data flows among the modules in the system without concerning the issue of control.

One very important problem of software engineering technologies is that they are usually qualitative [DEMAR82]. The saying "*You can not control what you can not measure*" [DEMAR82] still laughs at us like a ghost. The consequence is that we can not utilize the software engineering techniques as reliable means in software production management without solving this problem. This is why in recent decade, people started to put effort in software measures research. They tried to provide a sound mathematical basis for quantifying software engineering tools so that in the long run, software qualities will be able to be judged according to the quantitative evaluation of their designs. McCabe's and Holstead's measures are examples of quantifying software designs such as flowcharts or even code. Henry Kafura's measure is designed to evaluate the complexity of data processing in the software based on data flow diagram. Since they are calculated based on different representations of software design, they do reflect different characteristics of the software system.

Compared to other engineering disciplines, software engineering is still in the need of building sound foundation of measurement as a basis of real scientific discipline of its own. Software engineers demand good mathematical formalism in system design. This is one of the motivations of the thesis.

1.1. The Goals of the Thesis

The purpose of this research is to quantify the expertise of evaluating DFDs and provide DFD designers useful guidance based on the classified evaluation of DFDs.

The goals of this research are clearly stated in figure 1.1. Figure 1.1 is the scheme of this research and several major tasks are involved in the scheme. First task is to linearize the two dimensional DFDs by introducing another representation of DFDs - TR which will provide a good basis for the later tasks. The second task is to develop valid measures for DFDs to reflect their design qualities so that DFDs can be judged in a proper way. The third task is to classify the categories of evaluation criteria based on the valid measures and the poll result from a survey conducted in this research. The theoretical background of

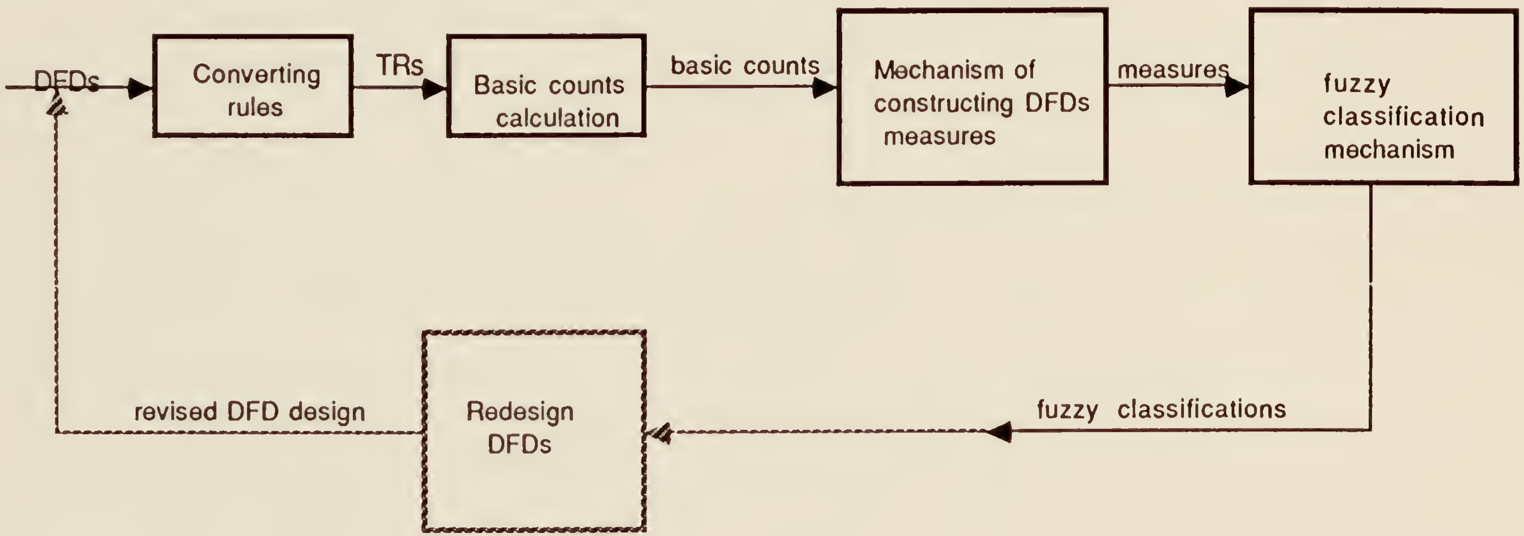


Fig. 1.1 The scheme of this research

this task is fuzzy set theory and the methodology is to build fuzzy membership functions by using linguistic approach. Finally, the fuzzy classifications of DFDs' evaluation are used to guide the DFD design.

The theoretical basis is built upon both conventional mathematics and fuzzy set theory. Fuzzy membership functions are set up to classify measures to reflect the evaluations of DFDs from experts in a survey. The reason we adopt fuzzy set theory is because we want to provide a more intuitive evaluation opinion, such as 'good', 'fairly complex', which are closer to human saying and give people better feeling about the quality of the software design. Especially, many measures now days don't consider the unit of the measure, therefore the claim "the measure is 570" usually will not provide much information to users. By using fuzzy membership functions, we can group measures into several levels over their ranges so that a relative level of a specific DFD can always be obtained. In this work, the membership function will be derived based on the empirical evaluation by the experts.

In order to develop a intuitive way to derive measures, we, first of all, linearize DFDs, i.e., we map a DFD into a linear, textual representation. This representation must keep all the characteristics of the original DFD. Then a number of measures are developed. The methodology guarantees that these measures can be obtained easily and directly from the textual representations.

1.2. Terminology

Researchers of software engineering have been using terms such as complexity, interconnections, width of the graph, to refer to different things. Therefore, it may be beneficial to define the meanings of some terms used in this thesis before we continue. Some terms are defined according to "Software Engineering Standars [STAND87] and some of them according to the definitions in this research.

Complexity :

Complexity is the degree to which a system or component has a design or implementation that is complicated or difficult to understand. In this research, we will emphasize structural complexity issues of DFDs -- the complexity that arises from a software document itself.

Modularity :

Modularity is the extent to which software is composed of discrete components such that a change to one component has minimal impact on other components.

Cohesion :

Cohesion is the degree to which the tasks performed by a single program module are functionally related.

Interconnection :

The connections among two or more components for the purpose of passing information from one to the other.

Token :

A token is a data item that need not be subdivided within a module when it is passed to this module to be processed.

Path :

A path in a DFD is a unique sequence of data/module names that goes from an external input of the DFD to an external output of the DFD. We can also refer it as LI-path, meaning linear independent path.

Width of the data usage :

The width of a specific independent data's usage is the number of occurrences of this data in the DFD.

Burden :

Burden means the degree of loading to a certain structure that can be a module in a DFD or even the whole DFD diagram. Loading can refer to different things such as token or path. For example, token burden of a module means the number of tokens this module process, including both input and output tokens. Path burden of a module can be the number of paths that go through this specific module. Interconnection burden of a whole DFD diagram stands for the number of interconnections this graph produces.

There are some other terms used in the research and we will give their definitions in the chapters they appear due to the need of using real graph examples to explain them.

1.3. Hypothesis

Considering the natures of structural measures of software tools, we adopted following hypothesis throughout this research. The beliefs behind these hypothesis is stated clearly in report "A mathematical Perspective for Software Measures Research" written by Austin C. Melton, Albert L. Baker, James M. Bieman, and David A. Gustafson [AUSTI88].

Suppose we have a set of similar software documents D, and let M be a software document measure defined on D and C be a quantifiable criterion that is an intuitive feeling of a specific nature of the software documents in D. Here, M is actually a quantitative predict of C [AUSTI88]. Now, we restate the two assumptions as below with slight change in the first one (compare to the correspondent assumption in the report) :

Assumption 1 :

There is an *order* on the documents in D; the order is based on the relative (complexity) degree of the measure.

Assumption 2 :

A *valid measure* M preserves or carries the "correct order" on D into R, here R is real number range. That is, if two documents d1 and d2 are related by the "correct order", then the images M(d1) and M(d2) are related in the same relative order in R.

Here, the "correct order" is determined by the criterion that is used to reflect a specific nature of the software document, e.g., complexity or cohesion.

1.4. Contents of The Thesis

The contents of the thesis layout in the following sequences. The first chapter of the thesis (this chapter) introduces the background of the research, some of the terminologies used in the thesis, hypothesis of the research, and the contents of the thesis.

Chapter two concerns with the development of linear representation of DFDs, covering the discussion of Adler's approach of decomposing DFDs, the necessity of developing new representation of DFDs, and, most importantly, the constructive rules for our linear representations of DFDs through examples and its strike characteristics.

Based on the linear representation developed in chapter two, chapter three will furthermore build structural measures of DFDs under the mathematical hypothesis stated in introduction part of this thesis. Examples are also given to illustrate the calculation and the usage of the measures.

Chapter four will deal with the survey that we have done in this research, explaining the survey environment and the data obtained from the survey, introducing the theoretical background and mathematical techniques used in data analysis, and summarize the conclusions of the analysis.

The fifth chapter talks about how to use fuzzy set theory to set up fuzzy membership functions for the DFDs' evaluation expertise that should (ideally) match the certain categories of the measurements of the linear representations of DFDs. In the meantime, the basic concepts of fuzzy set theory and the principles of constructing membership functions are given.

The last chapter, chapter 6 will give the overall conclusions of this research and what the author think about the possible future works in this area.

Chapter 2

Textual Representation of DFDs

The data flow diagram is a very important tool in software design. It gives the system implementors a dynamic overview of the data flow in the system. The decomposition of DFDs is a top-down method that takes a process as well as its inputs and outputs, and logically describes the process as a set of smaller processes. But even today, the decomposition is still performed in the manner in which analysts need to apply heuristics and expertise to the problem itself [ADLER88]. Because of this, good DFD decompositions will depend on the intuitive feelings of the software designers. Therefore, a systematic methodology is really needed in order to produce quality DFD designs.

Mike Adler developed an algebra to formalize the process of decomposition based on DeMarco's representation schema [ADLER88]. It can be used as a guideline or a tool in DFD decompositions. To prove its efficiency, he also proposed some measures to try to evaluate the quality of the resultant decomposed DFDs. There are still some problems in this approach that will be listed as following :

Problem 1 :

By using this approach, many DFD decomposition problems become trivial. The extreme situation is when all the inputs contribute to all the outputs. No decomposition can be done in this case. Unfortunately, that is really fairly common in the real world. We have used some data flow designs from real projects in this research to apply Adler's approach and a fair amount of them became trivial. This means that his approach is still not general enough to be applied to real design problem.

Problem 2 :

The measure used in this approach to judge the quality of the decomposed DFDs has not taken some important criteria that are usually crucial to DFD designs such as complexity, interconnections, or cohesion into considerations. There is no measurement involved in this evaluation measure. For example, his criterion for a decomposed DFD to be "optimal" is "equivalent to initial sentence and non- trivial". Here, first of all, he didn't mention from which sense the "optimal" is defined. Secondly, since he didn't prove the uniqueness of the decomposition (actually, there is no uniqueness because of the lack of uniqueness of some of the operators), there can be more than one decompositions that satisfy the "optimal" criterion.

Problem 3 :

When the decomposition process can not go further because no operator can be applied, a special type of operator (for example, the weak substitution) can be used to add extra data flow to continue the process. But obviously, the semantics of the data flow information has been changed in this process and the resultant decomposition of this algebra will not actually be optimal because extra or redundant data flow will be included. Adler admitted in his article that "if those elements are not already in the matrix [The matrix is a table used to denote the relationship of inputs and outputs of the DFD], the graph interpretation of the transform could change" [ADLER88].

Problem 4 :

The resultant DFDs have so called "local flows" and they are actually interconnections of DFDs. But in his approach, these interconnections have been produced without knowing their meanings during the process of decompositions. The burden of finding out their meanings is left to designers. The problem is that it is even harder for the designers to add the meanings to a given graph correctly than to produce a meaningful decomposition manually.

The author thinks that the evaluation of decompositions should rely on a reliable methodology of evaluating DFDs. The existing problems of Adler's approach force us to start to develop textual representation for DFDs with the purposes of 1) getting a more general abstract form of DFDs, 2) reflecting characteristics of DFDs through our developed representation, and 3) providing a really useful basis for formalizing the evaluation of DFDs. These goals can be stated more specifically as requirements of the textual representation of DFDs in the next section.

2.1. Requirements of The Textual Representation

Bearing in mind the purposes of developing this textual representation for DFDs, it should satisfy the following :

One-to-One Correspondence :

One-to-one correspondence should exist in the sense that the textual representation(s) obtained from a specific DFD correspond to only one recovery DFD.

Monotonicity :

The meaning of monotonicity here is that adding to a DFD will make the textual representation stay the same or increase. Monotonicity has to hold for textual representation because of our hypothesis of this research. The representation is another form (or a projection) of a DFD so it will be valid if and only if it preserves the nature of the DFD. Therefore, whenever a DFD becomes more complex, the corresponding representation should reflect the change, i.e., it should also become more complex.

Preserving Information :

The representation must be able to preserve all the important information in the DFDs. For example, the number of modules in the DFD, the number of possible data flow paths in the whole graph, or the number of interconnections among modules of the DFD, etc. Most importantly, the semantics of the DFD such as the relationship among data flows.

Ease of converting :

The components of the representation and the components of the DFD should have fairly simple correspondences, i.e., it should not be hard to convert a DFD into its textual form and vise versa.

If we can successfully develop a representation satisfying the above requirements, then it will be a good basis for systematically deriving measures directly from the representation. We give the rules of converting from DFD into textual representation in section 2.2.

2.2. Converting Rules

We start to build the textual representation (TR) of a DFD by choosing one of its paths and then constructing the TR from the inputs and the module of this path. From choosing different path, we will get different TRs. But as we stated in One-to-One correspondence in above section, they will all recover the same DFD. In appendix G, we give several examples to show this.

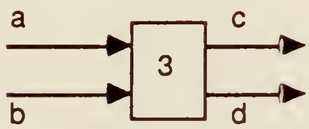
We will give the syntax of TR in the form of EBNF after some fundamental symbols used in TR are first introduced. Then we will explain the semantics of some more complicated symbols used in this grammar separately. Examples will also be given to help understanding better. Figure 2.1 gives an example of rule (1) through rule (5).

1) Each module in the DFD corresponds to an arrow with its name on the top of the arrow in TR. See the following example.

```
           ____
          |    |                        A
module    | A  |   is transformed to   --->
          |____|
```

2) Each input to a module appears to the left of the arrow for the module and different inputs are separated by comma ',' in TR.

DFD :



TR:

'a','b' ---> "c" | "d"

Figure 2.1 Example of converting rules (1) -- (5)

3) Each output from a module appears to the right of the arrow for the module and different outputs are separated by vertical bar '|' in TR.

4) Each external input of the DFD is quoted by "'" in TR.

5) Each external output of the DFD is quoted by '"' in TR.

Based on the fundamental symbols we developed above, we can now turn to introduce the syntax of TR before other rules can be clearly explained. The syntax will be expressed in the form of EBNF as following :

```
<TR> ::= <INPUT> <INTERNAL> <OUTPUT>
```

```
                  <MNAME>
<INTERNAL> ::= "------>"  |  <SHARE>
```

```
<SHARE>  ::=  "{"  <TR>  <MORE_SHARE1>  "}"

<MORE_SHARE1> ::=  ":"  <TR>  <MORE_SHARE2>

<MORE_SHARE2> ::=  <MORE_SHARE1>  |  e

<INPUT>  ::=  <PARALLEL_DATA> | ( "'" <MNAME> "'" |
          <NAME> | <NAME> "*" | "[" <NAME> "]" )
          <MORE_INPUT>* |  e

<MORE_INPUT> ::=  "," ( "'" <MNAME> "'" | <NAME> |
            "[" <NAME> "]" | <NAME> "*" |
            <NEST_TR>  )

<PARALLEL_DATA> ::=  <PARALLEL_ITEM>  <MORE_PARALLEL1>

<MORE_PARALLEL1> ::= ";" <PARALLEL_ITEM> <MORE_PARALLEL2>

<MORE_PARALLEL2> ::= <MORE_PARALLEL1> | e

<PARALLEL_ITEM> ::=  <NEST_TR>  |  <NAME>

<NEST_TR>      ::=  "("  <TR>  ")"

<OUTPUT> ::= (""" <MNAME> """ | <NAME> "*" | <NEST_TR> )
          <MORE_OUTPUT1>*  <MORE_OUTPUT2>* | e

<MORE_OUTPUT1> ::=  "|" ( """ <MNAME> """ | <NAME> "*" |
               <NEST_TR> | e )
```

<MORE_OUTPUT2> ::= "||" (""" <MNAME> """| <NEST_TR> | e)

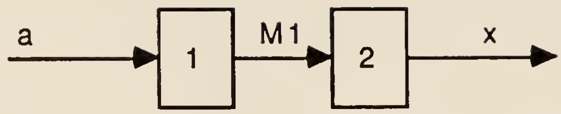
<NAME> ::= $\overrightarrow{\text{string}}$ | $\overleftarrow{\text{string}}$ | <MNAME>

<MNAME> ::= string

More symbols get involved in this grammar such as "[ ]", "*", and "||". We will explain their meanings one by one in the following rules and will also show examples about how to use them.

6) This rule concerns with the meaning of "( )" symbol. For the output that is not an external output element but a middle result or interconnection of the DFD, we enclose it and its continuations (i.e., another textual representation) by a pair of parenthesis. One example of this rule is shown in figure 2.2. In the example, 'a' is an external input, element "x" is an external output, and M1 is a middle result. Therefore, according this rule, we put M1 with its continuation, i.e., M1 --> "x", into a pair of parenthesis as (M1 --> "x"). The whole thing will be considered as the output of module 1. Or, constructively, for this DFD, we have a --> M1, and M1 --> x, then we connect two parts through the common item M1. The result is 'a' --> (M1 --> "x").

DFD :



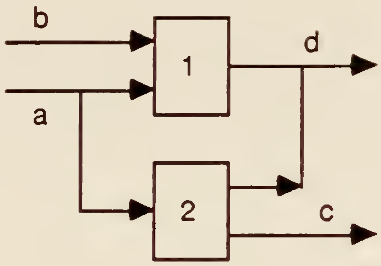
TR :

$$\text{'a'} \xrightarrow{1} (\text{M1} \xrightarrow{2} \text{"x"})$$

Figure 2.2 Example of rule (6)

7) Multiple data usage at the same level will be handled by using braces. In data flow diagrams, sometimes more than one modules will accept the same data as inputs or produce the same outputs. Several examples are shown in figure 2.3. In example (a), two modules accept the same input 'a' and produce the same output 'd'. Example (b) shows that two modules produce the same output but accept different set of input, while (c) illustrates the opposite situation with the same input but different outputs. The braces '{ }' quote the parallel modules that either share same inputs or produce same outputs at the same level of the DFD. Colon ":" is used to separate these modules. The shared data is put outside of the braces, inputs at the LHS and outputs at the RHS. In example (a), 'a' and "d" are input and output of both module 1 and 2. But 'b' is the input of only module 1 and "c" is the output of only module 2. The TR for (a) shows these facts and the vertical bar after output "c" means that the output list has not finished and the continuation is the outsider of the right brace, "d". Example (b) and (c) show how to develop the TRs when only either input or output is shared but not both.

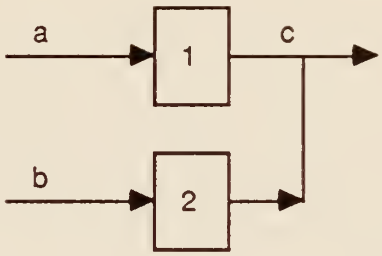
DFD :



TR : 'a' {'b' $\xrightarrow{1}$: $\xrightarrow{2}$ "c" | } "d"

(a) Mixed situation

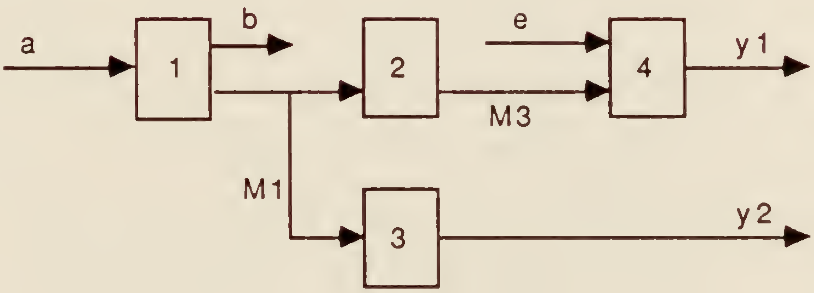
DFD :



{'a' $\xrightarrow{1}$: 'b' $\xrightarrow{2}$ } "c"

(b) Only share output

DFD :



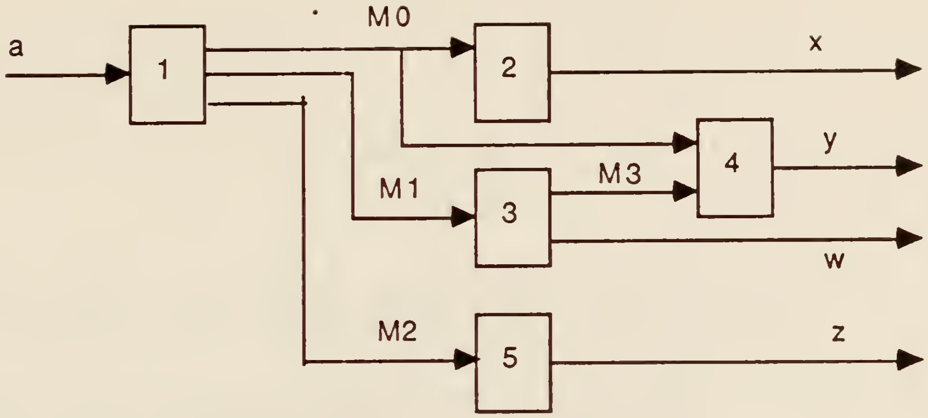
TR : 'a' $\xrightarrow{1}$ "b" | (M1 $\overset{3}{\{}$ ---> "y2" $\overset{2}{:}$ ---> (M3, 'e' $\xrightarrow{4}$ "y1") })

(c) Only share input

Figure 2.3 Examples of converting rule (7)
data sharing at the same level

8) Multiple data usage at different levels can not be expressed by using rule 7. In this case, we repeat each usage in the TR and then use left upper arrow and right upper arrow over data names to indicate the direction of the source data. Arrow '<--' means that the source data is at the lower level or the left direction. While '-->' indicates the higher level or the right direction of the source data. Without the upper arrow, we will not be able to find exactly where the data is from. Upper arrows always refer to where the data is produced so that there will be only one data that is not be upper arrowed and that is exactly the place where this data is produced. In the example shown in figure 2.4, M0 is used by both module 2 and module 4 but at different levels of the DFD. Therefore, M0 not only appears twice in the TR but also the M0 that gets into module 4 has an upper left arrow on its top, indicating that the data comes from the first left appearance of M0 (where M0 is produced by module 1) but not the output of module 3.

DFD :



TR :

```
 1            5                  2                  3
'a' ---> ( M2 ---> "z") | ( M0 ---> "x") | ( M1 ---> "w" |

(M3,  M0  ---> "y" ))

           <--  4
```

Figure 2.4 Example of converting rule (8)
data sharing at different levels

9) The symbol ';' in the grammar is used to express the parallel data flows in DFDs. We first present its definition and then discuss its usages by giving two examples in figure 2.5 and figure 2.6. In the grammar, symbol ";" appears :

<PARALLEL_DATA> ::= <PARALLEL_ITEM>
<MORE_PARALLEL1>

<MORE_PARALLEL1> ::= ";" <PARALLEL_ITEM>
<MORE_PARALLEL2>

<MORE_PARALLEL2> ::= <MORE_PARALLEL1> | e

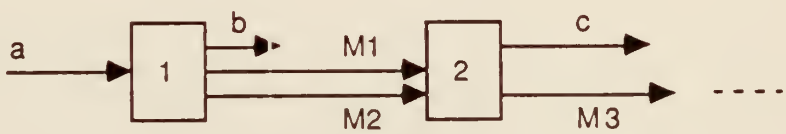
where non-terminal PARALLEL_ITEM can either be a single data name (<MNAME>) or another nested TR (<NEST_TR>). Suppose we have A ; B, then the meaning of this expression can be stated as :

- The most left element of A (if A is <NEST_TR>) or A itself (if A is <MNAME>) as well as the most left element of B (if B is <NEST_TR>) or B itself (if B is <MNAME>) are the outputs of the nearest arrow on their left
- The most right element of A (if A is <NEST>TR>) or A itself (if A is <MNAME>) as well as the most right element of B (if B is <NEST_TR>) or B itself (if B is <MNAME>) are the inputs of the nearest arrow on their right.

We will furthermore give two examples to clearly show how it can be used in expressing parallel data flows :

i) See figure 2.5 where M1 and M2 are parallel data flows between module 1 and module 2, i.e., they are both the outputs of module 1 and also they are both the inputs of module 2. The corresponding TR is shown below the DFD. According to the definition, since both M1 and M2 are single data names then M1 and M2 are both the outputs of the nearest arrow on their left (that is module 1). Also, they are both the inputs of the nearest arrow on their right (that is module 2). Actually, their nearest left arrow is the module producing them, and their nearest right arrow is the module accepting both of them as inputs. This is a typical parallel data flow between two consecutive modules, different data flows from the same resource and to the same destination.

DFD :



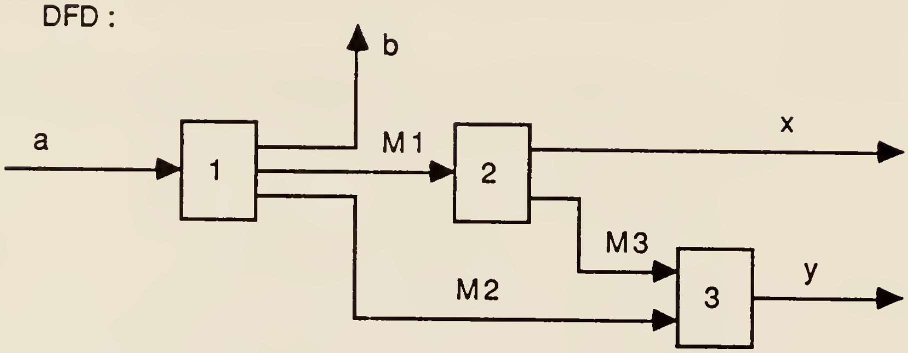
TR :

1 2
'a' ---> "b" | (M1 ; M2 ---> "c" | (M3 .....

Figure 2.5 Example of converting rule (9), case (i)

ii) Second example shows the usage of ";" when not all of A and B are single data names. figure 2.6 gives a DFD with parallel data flows among different levels. Both M1 and M2 are produced by module 1 but they are sent to different modules at different levels. M1 gets into module 2 and it results M3 which is used with M2 as the inputs of module 3. Here, we say parallel data flow in the sense that M1 and M2 are from the same source and then the consequence (M3) of one of them (M1) is used together with the other one (M2) as the inputs of one module (module 3). In the case of figure 2.6, A is M2 and B is a TR (M1 -> "x" | M3). Again, according to the definition, M2 and the most left element of B (M1) will be the outputs of the nearest arrow on their left (module 1). M2 and the most right element of B (M3) will be the inputs of the nearest arrow on their right (module 3). This is exactly the situation in figure 2.6. The TR using ";" does reflect the semantics of the corresponding DFD.

DFD :



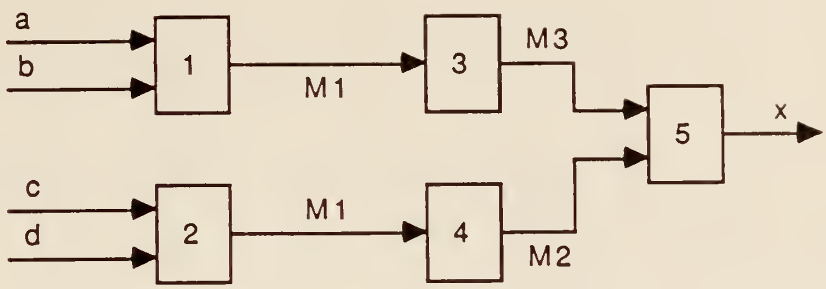
TR :

1 2 3
'a' ---> "b" | (M2 ; (M1 ---> "x" | M3) ---> "y")

Figure 2.6 Example of converting rule (9), case (ii)

10) Rule 10 deals with the data that have the same name but different semantics. It is quite often in DFDs that different modules will produce data with the same name that finally go to different places. The situation here is different from the data sharing. Data sharing means that several data names are from the same resource. Actually, the data from different modules are semantically different even though they have the same name. For example, different modules update different attributes of symbol table in compiler design. In this case, we need to distinguish those data in TR in order to keep the correct semantics of the corresponding DFD. The strategy of doing this is simple. We just repeat the name without referencing (in data sharing, all data except the original one is upper arrowed to refer to the resource). Figure 2.7 clearly shows how it works.

DFD :



TR :

```
     1          3                          2
'a','b' ---> (M1 ---> ([M3], ('c','d' --->
    4              5
(M1 --> (M2, M3 ---> "x" )))))
```

Figure 2.7 Example of converting rule (10)

The meaning of '[ ]' will be explained later. In this example, both module 1 and module 2 produce data M1 but they are semantically different. In the corresponding TR, we have two appearances of M1 and, obviously, the first M1 is from module 1 and the second M1 is from module 2 (not from the same resource !).

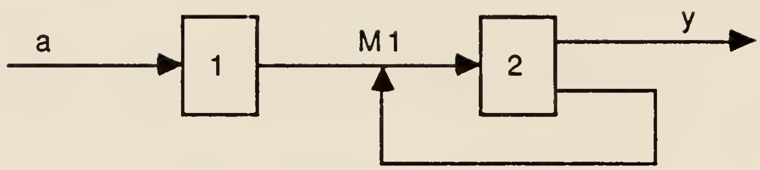
11) Loop is another hard thing to be represented in TR, especially the nested loops. In our grammar, we use two symbols, "||" and "*" to represent data loops in DFDs. Before we introduce their meanings, it will be helpful to discuss the natures of loops first. If we have a loop shown in figure 2.8, several facts need to be explicitly represented in our TR :

i) The M1 from module 1 is semantically different from the M1 from module 2

ii) The loop itself which is shown in figure 2.9 (a)

iii) The final exit of the loop which is shown in figure 2.9 (b)

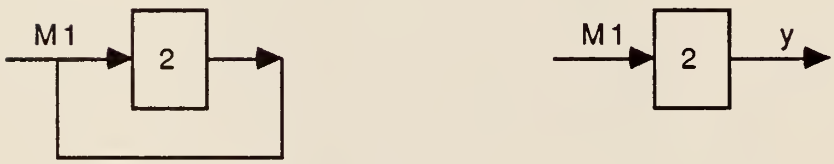
DFD :



TR :

$$'a' \xrightarrow{1} (M1 \xrightarrow{2} M1^{*} \;\; || \;\; "y")$$

Figure 2.8 Example of converting rule (11)



(a) The loop body (b) The final exit of the loop

Figure 2.9 Example of conveting rule (11)

The two symbols are used to explicitly describe above semantics of the DFDs :

a) Symbol '||' means that the following element is the final exit of a loop

b) Symbol '*' will be used to mark the data which form the loop.

By using those two symbols, we can get the TR for the DFD of the example in figure 2.8.

In the TR, the first M1 is the output of module 1 and the second M1 is the output of module 2. Then, the second M1 is linked back (by the meaning of symbol '*') to the place where the first M1 appears to complete the loop. For module 2, it initially gets the output of module 1 as input and later accepts the M1 produced by itself as input. The loop process repeats till the exit 'y' is reached.

12) We start to build a TR by choosing one of the paths in the DFD. In the example in figure 2.7, we start from module 1 and this path can continue till module 5 is reached because another input M2 of module 5 is from the other path. Therefore, when we approach the output M3 of module 3, we have to start to process the other path in order to get data M2 as another input together with M3 to module 5. So, we need to postpone the usage of M3 at this point till M2 is produced. Symbol '[ ]' is used for this purpose and it will postpone the usage of the data quoted by '[' and ']' till the next appearance of the same data name. In the example shown by figure 2.7, the usage of the first appearance of M3 (quoted by '[]') is postponed till the next appearance of M3 which, together with M2, is the input of module 5. If we choose module 2 to start, then we will need to postpone the usage of M2 to wait for data M3. Although we will get different TR in this case, we proved in Appendix G that they will recover the same DFD.

2.3. Example and Conclusions

So far, we have introduced 13 converting rules. With them, different kinds of DFDs can be easily transferred into their TR forms. Figure 2.10 is another fairly complicated DFD example and we will use the rules developed above to give its TR representation.

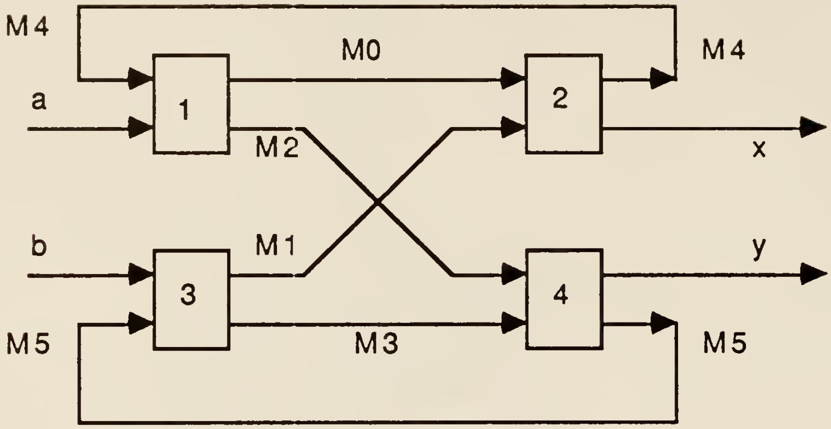


Figure 2.10 Example of using converting rules

This is a symmetric data flow diagram with loops and parallel data flows. The corresponding textual representation could be :

```
          1                        3                  <-- 2
 'a',M4 ---> ([M0],('b',M5 ---> [M3] | (M1,M0 --->

                              <-- 4
     M4* || "x" )) | (M2,M3 ---> M5*||"y")
```

As we discussed in One-to-One correspondence, symmetric DFD can get different TRs but they will turn to the same recovery DFD. In appendix G, we developed two different TRs for this example and used them to recover the same DFD.

In order to test the effectiveness of TRs, with the help of I/O matrix, we can prove that TRs can recover the same matrix as the original one upon which the corresponding DFD is developed (see Appendix G).

A TR is a linear representation of a two dimensional DFD. But it preserves all the structural information of a DFD, including the number of modules, number of interconnections, the interfacing patterns of them, the possible paths, and the relationships among data flows. The converting rules are developed with the purposes of keeping those information in TRs and building a good basis of calculating structural measures from TRs. Next chapter will discuss the measurement issue of TRs.

Chapter 3

Structural Measures of Textural Representation

Another important task of this research is to provide some useful measures to aid the evaluation of DFD designs. The measures developed in this chapter can either be used as the indications of the design quality of DFDs or, furthermore, serve as a quantitative guide in the process of DFD decompositions.

Much of the software measures research has concentrated on source programs and has provided quantitative means of assessing the complexity, cost, and reliability of the resultant code. But this research emphasizes the early phase of software design because we think that the early evaluation of the software designs can lead to significant improvements in software quality and a significant decrease in development cost.

The data flow information of a program has been used for measuring program characteristics in two different approaches. One is to use the data flows among modules to define the nature of the interrelationships of these modules ([HENRY811], [HENRY812]). Another one is to use the data flows within a module to describe the nature of the program ([OVIED80], [IYENG82]). In this research, measures will be developed according to the data flows among modules (the first approach) so that the evaluation can be done based on the structure of the whole diagram instead of just the individual modules.

In addition, the research puts attention to the structural measures of DFDs without concerning any criterion which is related to the semantics of the project. The reason for that is that DFDs themselves only depict the flow of data rather than flow of control. They treat data as information but do not distinguish what kind of data is being used [TROY81]. Therefore, the measurements developed in this chapter will not be concerned with the type of the data but only with the data flow pattern, i.e., we don't care about how to control the data flow in the DFDs but only care about where the data flows come from and where they go in the DFDs.

Because of the strategy we adopted in the research, we want to measure whatever we see from the *structure* of the DFDs but not from the *semantics* of the DFD. The textural representation of a DFD we developed in chapter two is a good basis of calculating structural measures because it is exactly a structural projection of a DFD and keeps all the structural characteristics of the DFDs.

Basically, we try to develop the measures which will be useful in evaluating DFDs according to some criteria such as the structural complexity of the DFDs, interconnections, etc. The later sections will introduce the basic measures counted from TRs and some advanced measures built to reflect the specific natures of the data flow diagram design. These measures will also be developed under the second hypothesis stated in chapter one.

3.1. Basic Measures Counted From TRs

Before we start to introduce the basic measures obtained from TRs, we will show examples of several terms. This will help to avoid confusion and help to explicitly explain the background.

The first one is the "width of the data usage'. The definition given in chapter one is "the number of the occurrences of an independent variable in the DFD". Here, attention must be paid to the difference between this concept and 'fan-out' concept of Henry-Kafura's. The example in figure 3.1 shows the differences.

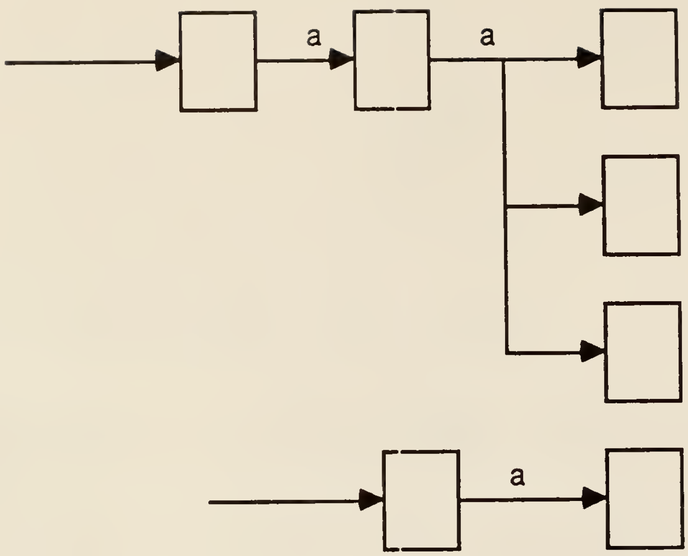


Figure 3.1 Example of 'width of the data usage'

In figure 3.1, 'a' is an independent variable. According to the definition, the width of a's usage will be 5 since it appears five time in the graph. Henry-Kafura's fan-out would only be three in this case.

The second is concerning with term "path". The example in figure 3.2 more clearly explains the meaning of the definition for a path. For the DFD, the unique sequence of variable/module names that goes from an external input to an external output will be " a X b Y c ". Here, a, b, and c are variables, and X and Y are module names. The sequence defines the order in which a specific data flow passes through different components of the DFD.

Corresponding path is : a X b Y c

Figure 3.2 Example of a path

Following is the list of basic measures we can obtain directly from TR :

1) UM --

Number of unique arrows in the representation or number of unique boxes (modules) in the graph

2) SM --

Number of modules that share a common input data with other modules

3) LM --

Number of modules that are in a loop

4) UI --

Number of unique external inputs

5) CIi --

Number of occurrences of the ith unique external input. The attention has to be paid to both symbol '{}' and '[ ]' when we count this measure from TRs. Each modules in '{}' share the input outside of the brace so that the occurrences of the outside input should be the number of modules within the braces. The same thing for the count of occurrences of outside outputs. Any data which is quoted by '[ ]' should not be counted because symbol '[ ]' postpones the data usage. The occurrence will be counted when it is really used.

6) UO --

Number of unique external outputs

7) COi --

Number of occurrences of the ith unique external output (see (5))

8) CTMi --

Number of tokens related to the ith module, including input tokens and output tokens

9) UIC --

Number of independent interconnections (the unique variables that are not quoted either by ' ' or " ")

10) CICi --

Number of occurrences of the ith interconnection variable (see (5))

11) CIC --

Total number of occurrences of all interconnections $\sum_i (CICi)$

12) UIV --

Number of independent variables in the graph, including inputs, output, and interconnections. It is calculated by UI + UO + UIC

13) CIV --

Total number of occurrences of independent variables which is $\sum_i (CIi) + \sum_i (COi) + CIC$

14) P --

Number of possible data flow paths in the graph (from every external input to every external output)

15) CW --

Conceptual width of the graph (The biggest degree of data usages). It is Max(CICi, CIi, COi)

16) NPMi --

Number of paths that go through the ith module (box)

17) DPi --

The length of the ith path (the number of components of the DFD the ith path has to pass through)

18) DP --

The conceptual length of the graph (The longest path in the graph). It is Max (DPi).

19) CL --

Number of loops in the graph

20) LLi --

The depth of the ith loop (the number of modules that are involved in the ith loop. For example, for a self-loop, LLi = 1)

21) CSIZE --

Conceptual size of the graph which is the product of the conceptual width and conceptual depth of the graph CW * DP

All above measures can be easily obtained from textural representations of DFDs. These measures describe different characteristics of DFDs and form a good foundation for designing advanced measures that are the indications of software design qualities. The following section discusses the criteria we will use in evaluating DFDs, the factors that will influence the criteria, in the way they will affect the quality of DFDs, and how we build measures to reflect all the above.

3.2. Criteria in Evaluating DFDs

Several criteria will be used to evaluate data flow diagrams : complexity, interconnections, modularity, cohesion, and ease of implementation. These criteria are actually related to or influence each other in the sense that an increase of one will often cause the same/different direction of change in the others. Therefore, when we consider the proper measures for the criteria, we will also take this fact into count in order to correctly model the real situations. We will first introduce each criterion in detail and then try to use appropriate measures to model it.

Interconnections :

For ease of understanding, we discuss interconnections first. This criterion can be considered as an indication of the strength of coupling among modules. The strength of coupling can be affected by following reasons :

a) Average token burden. The number of tokens through each module in the diagram is related to the coupling strength. The larger the average number of tokens that are processed by each module in the DFD, the stronger the coupling. This can be formulated as : the total number of tokens processed in the DFD divided by the number of modules in the DFD, i.e.,

$$ATB = \sum_i (CTMi)/UM$$

b) Average connection burden. The coupling increases with the number of interface connections in the DFDs [TROY82]. The larger the average number of interconnections each module creates in the DFD, the stronger the coupling. This can be calculated by the following formula, the total number of interconnections in the DFD divided by the total number of modules in the diagram :

$$ACB = CIC / UM$$

c) The interfacing pattern of these interconnections. Again, coupling strength increases with the increasing complexity of the interface among modules in a DFD [TROY82]. It can be furthermore considered from several perspectives :

- Average path burden. The way the interconnections are related in a DFD determines the number of possible paths in the DFD. The larger the average number of paths that go through each module in the DFD, the stronger the coupling. This can be measured by :

$$APB = P / UM$$

  that is, the total number of possible paths in a DFD divided by the total number of modules in the DFD.

- Data sharing degree. Coupling also increases when more than one module interfaces with the same data, i.e., share a common environment [TROY82]. We define this degree based on two considerations. One is the percentage of the total number of modules in a DFD that share common data environment (MDSR). The other one is the percentage of the independent variables that are shared in the DFD (DSR). These two factors are defined by the following equations, separately :

$$MDSR = (SM + UM) / UM$$

$$DSR = CIV / UIV$$

  Then, the data sharing degree is simply the summation of these two

:

$$DS = MDSR + DSR$$

- Loop density. This density is viewed also from two aspects, average loop length :

$$ALEN = ((\sum_i (LLi)) / CL) + 1$$

and loop frequency shown as below :

$$LF = (LM + UM) / UM$$

Finally, the loop density can be defined as following :

$$LD = LF * ALEN$$

All of the above factors proportionally influence the coupling strength of the DFD. Therefore, we use the summation of them to reflect the nature of interconnection of DFDs :

$$INTER = ATB + ACB + APB + DS + LD$$

Complexity :

Measuring whatever we see from the DFDs is one of our goals. Therefore, complexity of DFDs are affected by their size and interconnection characteristics since they are the only things we can see and measure from DFDs. We have discussed the measure of interconnection. In the last section, we introduced the measurement for conceptual size of the DFDs which is the product of conceptual width and length of the DFDs. We here furthermore define the complexity measure for a DFD to be the product of its conceptual size and its interconnection measure which is formulated as following :

$$COMP = CSIZE * INTER$$

Modularity :

Modularity is the extent to which software is composed of discrete components such that a change to one component has minimal impact on other components [STAND87]. Therefore, modularity actually is a measure of the relative independence among modules. Usually, fewer connections among modules indicates a better module independence and thus a better modularity. It could be measured by the average number of variable occurrences over modules which is formulated as the following :

$$MOD = CIV / UM$$

Cohesion :

In this study, we are only concerned with two kinds of cohesions, one is functional cohesion and the other one is logical cohesion. We again will consider them separately.

a) Functionally cohesive modules should (ideally) do just one task, i.e., it should have singularity of tasks. This is similar to the concept of measuring dependence among modules, cohesive modules are more independent with necessary connections to other modules. Therefore, we think that this criterion should be somehow related to the evaluation of modularity. Instead of defining a measure for this criterion, we will test this relationship by statistical analysis later stated in chapter four. If we can confirm the relationship, the measure can be designed based on the conclusion from there.

b) In this research, we define the logical cohesion as a logical strength of the whole diagram, i.e., see how strongly the modules are related to each other.

Low logically cohesive diagrams do unrelated tasks.

Again, we feel that this criterion might be affected by interconnection. Unrelated modules in a DFD usually means that the DFD carries out unrelated tasks. Also, the high interconnections usually indicates that modules are tightly related. We intend to explore the relationship between logical cohesion and the other criteria such as interconnections. How to design this measure will depend on the data analysis results explained in chapter four.

Ease of Implementation :

We believe that criterion 'ease of implementation' is actually the proportional function of complexity. The more complex a DFD is, the harder the implementation will be. The relationship will be assessed through the experiment reported in chapter four.

3.3. Examples of Calculated Measures

In order to present how to calculate the measures designed in previous section, we choose six DFDs (they are the ones used as the objects in the experiment introduced in chapter four), calculate their corresponding basic counts, measures, and show them in Appendix A. The rows are the lists of external inputs/outputs, interconnections, modules, paths, and loops. The columns are the corresponding basic counts, such as unique modules UM, the sharing modules SM, etc. Those rows and columns form a table for each DFD. Some of the tables are too big to put on one page, so they are shown on different sheets. From the numbers marked for rows and columns, it is easy to recognize each part of a table.

Based on the basic counts of the six DFDs, various measures are calculated for different criteria. They are shown in Table 3.1. The rows correspond to six DFDs and the columns are the measures. For example, for the first DFD, its ATB is 2.71, ACB is 0.43, APB is 1.86, COM is 194.84, and INTER is 9.74.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N
1			ATB	ACB	APB	MSDR	DSR	DS	ALEN	LF	LD	COM	INTER	MOD
2														
3	DFD-1		2.71	0.43	1.86	1.857	1.889	3.746	1	1	1	194.84	9.742	2.429
4	DFD-2		3.25	0.75	1.5	1.75	1.667	3.417	1	1	1	148.755	9.917	2.5
5	DFD-3		2	0.67	0.33	1	1	2	1	1	1	42	6	1.33
6	DFD-4		4.67	1.33	10	1.667	1.294	2.961	4	1.667	6.668	1947.8	25.629	3.667
7	DFD-5		4	1	4.4	1.8	1.667	3.467	3	2	6	679.212	18.867	3
8	DFD-6		4	1.25	3	1.5	1.2	2.7	3.5	1.75	6.125	443.95	17.075	3

Table 3.2 Measures for 6 DFDs

Chapter 4

Survey in this research

In chapter three, we developed structural measures for DFDs in order to evaluate their quality. The question is how effective these measures will be when we use them to evaluate real DFDs. In order to test their validities, we conduct a survey to empirically prove their significance.

There are two goals in this survey. They are :

1) Empirically validate the structural measures developed in chapter three by using statistical analysis techniques

2) Build up fuzzy membership functions for linguistic concepts (terms) used in evaluating DFDs based on the survey data and the validation result from 1)

The hypothesis of the first step is that a valid measure will preserve the "correct order" of the DFDs (see assumption 2 in chapter one). While the hypothesis of the second step is that the meanings of all terms in a natural language are to a lesser or greater degree vague, such that, the boundary of the application of a term is never a point but a region where the term gradually moves from being applicable to being nonapplicable [HERSH76].

This chapter will give detail discussions of the first goal. It will introduce the survey environment, data collection, data analysis, and results from the data analysis. Chapter five will discuss the issues related to fuzzy set theory and its application in this research, and present the results of the second step.

4.1. Survey Environment

This survey was done at Kansas State University, Department of Computing and Information Sciences. It was performed by choosing a set of objects, presenting them to the chosen subjects, and asking the subjects to evaluate the objects according to some predefined criteria. The subjects and the objects were selected in the following situations :

Subjects :

The resource of the survey data was two software engineering classes of Computing and Information Sciences Department at KSU, one was graduate level class CIS 740 and the other one was undergraduate level class CIS 541. The reason we chose these classes was because the students in these classes were familiar with DFDs and the terminologies related to this research. The total number of subjects was 53.

Objects :

The objects were 6 sets of parent boxes and the expanded data flow diagrams. The DFDs chosen for the survey had different degrees of complexity within each measurement category such as interconnection, complexity, etc. Also, their semantics should be hidden from the subjects because a) we are only concerned the structural evaluation, and b) the pilot study done at the early stage of this research showed that people who know the semantics of the DFDs can not structurally evaluate them properly. Six objects are selected in this way. They are presented in appendix B.

4.2. Data Collection

Data collection includes both data-collection tool and collection procedure. They will be discussed in the following :

Data-collection tool :

The data-collection tool used in the survey is the data-collection form. The form addresses 9 criteria for evaluating DFDs and has 10 questions in it. The questions are designed to satisfy the following two conditions :

i) Explicitly define the meaning of each criterion before the questions are asked to avoid misunderstandings

ii) In the definitions of the criteria, avoid giving hints about how the related measurements would possibly be constructed so that the answer will not be led by the questioner

In the data-collection form, for each question, there are 6 possible answers. Each answer represents a evaluative adjective phrase which indicates a certain degree of complexity of the related criterion. The six answers are ordered along a favorable-neutral-unfavorable continuum. For example, for complexity criterion, the six answers can be 'very complex','fairly complex','more complex than simple',.., till 'very simple'. In the survey, criterion complexity is asked twice, the first and the last. The purpose of this is to give subjects another chance at the end to adjust their answer based on their overall feeling of the objects (see appendix C).

Data Collection Procedure :

We presented the survey during class time. After presenting the purpose of the survey, we asked students to read the definitions of the criteria and to ask questions about them if any. Again, when we answered questions, we tried to avoid giving hints about how to choose an answer or how possibly the measure would be built. The answer session began when there was no question left about the definitions. We showed each chosen object (DFD) by slide and asked students to answer all the 10 questions about the current slide. The process continued till all objects were examined. Every student signed his/her name on the data-collection form in case we needed to ask specific questions about the answers.

After the survey, the result were entered into a spread sheet under the software Excel. One of the answer forms was dropped due to too many missing values in it. Therefore, the final valid number of subjects was 52. Next section will examine the data processing process and present the results from the statistical analysis.

4.3. Data Analysis and Results

In Appendix D, there are 5 spread sheets corresponding to 5 questions that we are interested in the research. The rows represent subjects and the columns represent objects. The values in the spread sheet are the symbols for the answers. For example, in the first spread sheet, value 1 stands for answer 'very complex' and 2 stands for answer 'fairly complex'. Here, these values keep the order of the original phrases. This fact makes statistics a possible tool in the later data analysis.

Several analysis techniques were used in data processing for different purposes. They are separately discussed in the following sections.

Simple Statistics :

In order to check the consistency of the answers and the bias of each subject, several simple statistics were calculated such as average, standard deviation, and frequencies. For each row in the spread sheet, the average was calculated to show the basic attitude of the corresponding subject. Some people are always more optimistic than others and some people are always more pessimistic than others. This average will provide us a good review of their original bias and it is the base for normalizing the answers. In this research, we emphasize evaluating the relative complexity levels of DFDs hence we finally need to adjust all people's basic attitudes to the same ground. This is the normalization problem and we will talk about it in the next section.

Another average was calculated based on each column. It represents the average evaluation of a criterion of the corresponding object. If different columns have different average values, it means that different objects get different evaluations on the criterion scale.

This average is shown on row 71 on the spread sheet.

Also based on each column (each object), frequencies were calculated to show the distribution of the answers. They are shown from row 62 till row 67. This is good for checking the consistency among the answers. For example, on the first spread sheet (addressing criterion 'complexity'), the number 22 in cell C-64 means that 22 people (out of 52) responsed '3' (more complex than simple) for the first object. Also, if different columns have different peaks on their distribution, it means that different objects get different evaluations on this criterion.

Normalization :

In order to adjust all the answers to the same base, we normalized the raw data. We subtracted from every answer the average value of the corresponding row, i.e., we adjusted the original values to the distances from their mean values. In this way, we diminished the bias but still preserved the same order of the evaluations which is more important.

The normalized data is shown in Appendix E. Frequencies were also calculated for the normalized data based on the six subranges defined on the range of the normalized answers. In Appendix E, for each question spread sheet, row 73 and 74 indicate the maximum and minimum values of the corresponding columns and then the Max value on row 59 is the maximum value of the whole spread sheet (the maximun value of row 73) and the Min value indicated on row 59 is the minimum value of the whole spread sheet (the minimum value of row 74). Then the whole range of the answer will be [Min, Max]. The length of the whole range is indicated by Total on row 59. Dividing the whole length by six will be the length of each subrange. The value on row 60 is this length. In this way, we get six subranges representing the six evaluative adjective phrases. Based on these subranges, frequencies were calculated again, counting the number of normalized answers that fall in a certain subrange.

The normalized data shows slightly different distribution than the raw data. But again, the rank of the evaluations is preserved.

Statistical Analysis :

Based on the averages across the subjects, statistical analysis was applied by using statistics package SAS. The reason of using averages (but not the original data) is that the averages are more smooth. Even though using averages will lead to a small sample size, we adopted it because they are calculated from fairly big sample size (52) so that they are stable enough for the statistical analysis.

The data shown on page F-1 and F-2 is the data file used by SAS. The columns in the file correspond to the addressed questions in this research as well as some measures calculated from the chosen objects according to the definitions in chapter three. The first two rows indicate the relationship between SAS variable names and their meanings. For example, second column is 'complexity' whose variable name in SAS file will be V1. The rows in the SAS file correspond to the objects used in the survey.

We need to validate 1) the effectiveness of the measures we developed in chapter three and 2) the relationship among the different criteria. Several hypothesis related to these purposes are stated as following :

a) complexity measure is valid according to the definition in assumption (2) in chapter one

b) interconnection measure is valid under assumption (2)

c) modularity measure is valid under assumption (2)

d) ease of implementation is proportionally related to complexity and/or to modularity

e) complexity is somehow related to modularity and/or cohesion

Two basic techniques were used, one was correlation analysis and the other was regression analysis. They are discussed below.

Correlations :

A complete correlation matrix across all the variables in SAS file was built to show the correlations between different pairs of variables. It is on page F-3 and F-4. From the matrix, we can see that V1 (complexity) and V2 (interconnection) have strong correlation -0.94, V1 and V13 (complexity measure) have correlation -0.91, V2 and V14 (interconnection measure) have correlation 0.93, and V3 (modularity) and V16 (modularity measure) only have correlation 0.25.

One interesting thing is that some correlations verified our guesses obtained after we reviewed the match between survey result and the calculated measures. For example, for complexity, we calculated complexity measures for all the objects and then arranged the objects in the order of the complexity measures, from the lowest (least complex) to the highest (most complex). The order is : obj-3, obj-2, obj-1, obj-6, obj-5, obj-4. This can be seen from Table 4.1. The first row of Table 4.1 shows this order and the second row lists the corresponding measures. In order to check the match between measures and the survey data, we also included the column averages for all the objects in the third row of Table 4.1 and they are arranged in the same order. These averages are from the normalized data. The column averages should have the reverse order (from biggest average to the smallest) because in the original answer, 1 stands for 'very complex' while 6 stands for answer 'very simple'.

By checking the consistency of the two rows, obj-5 is the only one that doesn't match.

Objects	obj-3	obj-2	obj-1	obj-6	obj-5	obj-4
Measure	42	148.8	194.8	443.9	679.2	1947.8
Average	1.17	0.69	0.19	-0.46	0.19	-1.79

Table 4.1 Complexity evaluations & complexity measures

We rechecked object 5, compared it with other objects, and checked how the complexity measure was constructed (complexity measure is calculated from interconnection measure and the size of the DFD. One component in interconnection measure is data sharing degree DS). We thought that the reason for that was that the subjects underestimated or even ignored data sharing issues when they judged the complexity of the DFD. The object 5 has very strong data sharing, four modules share both the same input and output and the shared output is linked back to complete a data loop. This is a fairly complicated pattern of interconnections. From maintenance point of view, this structure will also definitely increases the complexity of the DFD. However, visually, the object 5 doesn't seem to show a messy connections among modules that might mislead the subjects to underestimate the complexity of the DFD.

From the correlation matrix on F-4, V2 (interconnection) and V22 (data sharing measure DS) have correlation -0.06. It gave us the partial confidence about our conclusion. In order to furthermore prove the conclusion, we later on carried out regression analysis again to show that there is no evidence in the survey data about the relationship between interconnection evaluation and the data sharing degree.

Regression analysis :

Regression analysis was carried out among different groups of variables. Those different groups and their regression analysis results from SAS are listed from page F-4 of Appendix F. For the ease of discussion, some useful information is extracted from there and put into Table 4.2.

Regression Results Summary			
Dep.	Indep.	R-squ.	P
V11	V12	0.9889	0.000
V11	V9	0.7683	0.023
V11	V13	0.7205	0.033
V9	V16	0.0626	0.634
V8	V14	0.8564	0.009
V8	V17	0.8070	0.016
V8	V18	0.8482	0.010
V8	V19	0.8156	0.015
V8	V22	0.0037	0.874
V8	V25	0.7178	0.034
V8	V17-V25	1.0000	*****
V7	V8	0.8883	0.006
V7	V15	0.8716	0.008
V7	V13	0.8427	0.011
V7	V8,V15	0.9290	0.023
V7	V10	0.4454	0.147

Table 4.2 Summary of Regression Results

There are 15 regressions listed and their results will lead to our conclusions. Two statistics, R-square and P, indicate the significance of the regression result. The closer the value of R-square is to 1.0, the larger the proportion of the dependent variable values that can be predicted by the independent variables according to the regression formulas. For example, if R-square is 0.9 and we use the regression formula to predict 100 points, then 90% of the predicted points are reliable or trustworthy. P is the strength of the evidence that against the hypothesis : the coefficients of the independent variables in the regression formula are zeros. Therefore, the smaller the P is, the weaker the evidence is to against the hypothesis. Usually, if P is less than 0.05, the evidence will be considered too weak to against the hypothesis, i.e., the coefficients are trustworthy.

Therefore, the closer the value of R-square is to 1.0, the stronger the linear relationship between the dependent variable and the independent variables is. Also, the smaller the P is, the more confidence we should have to the relationship built by the regression formula between the two regressed variables. According to the results shown in Table 4.2, we can get our conclusions in the next section.

4.4. Statistical Conclusions

The conclusions will be stated one by one in the order of the regressions listed in Table 4.2. All the conclusions are presented in a literal way but not a numeric way because regressions were done based on both numeric values and non-numeric values (the normalized answers are still symbols but not numerical values). We will not use the regression formulas as deterministic relationships among criteria but only use them to conclude the patterns of how they affect each other. This is valid because the symbolic system we are using (the symbols representing the evaluative phrases) keep the correct order of the criteria even though they do not have any precise numeric meanings.

1) Ease of implementing a DFD is linearly affected by the complexity level of the DFD. This conclusion is from the first regression result with R-square=0.9889 and P=0.000.

2) Ease of implementing a DFD is linearly affected by the modularity of the DFD. The conclusion is from the second regression result with R-square of 0.7683 and P=0.023.

3) Ease of implementing a DFD is a linear function of the complexity measure developed in chapter three. This is concluded from the third regression in Table 4.2 with R-square=0.7205 and P=0.033.

4) The modularity measure developed in chapter three is not a valid measure because it doesn't have any relationship with the modularity evaluations from the survey. The regression between this measure and the survey result shows this with R-square of 0.0626 and P of 0.634.

5) The interconnection measure developed in chapter three shows good prediction ability. This can be conclude from the regression between this measure and the survey result

about the corresponding criterion. The R-square value is 0.8564 and P value is 0.009.

6) The interconnection measure is constructed from five components, average token burden ATB, average connection burden ACB, average path burden APB, data sharing degree DS, and loop density LD. Regression 6 through regression 10 demonstrated that interconnection criterion is linearly related with every one of the components except DS. Also, the result of regression 11 shows that the regression model consisted by those five components can perfectly predict the interconnection criterion (R-square is 1.0 !). This is a surprisingly good result which indicates that the design of interconnection measure is reasonable. By looking at the regression formula for regression 11 (see Appendix F), we notice that the coefficients of the components are different from the one we developed in chapter three. But the formula from chapter three is also significant enough to be used as a valid measure. This can be seen from the result of regression 5 (the regression between interconnection evaluation and the interconnection measures) with R-square value of 0.8737 and P value of 0.008.

   The fact that interconnection evaluation result is not significantly related to DS measure is actually the conclusion we are expecting. The regression between V8 - interconnection and V22 - DS measure shows no relationship. This can be seen from the low R-square value 0.039 and the very high P value 0.873 (see the result of regression between V8 and V22). Therefore, the conclusion that the subjects underestimated or ignored data sharing issue when they evaluated the interconnection has been proved by both correlation and regression analysis. This conclusion is the basis of the decision of dropping obj-5 (DFD5) when we try to built fuzzy membership functions of criteria in chapter five.

7) Complexity criterion is linearly affected by interconnection criterion. This conclusion is from regression 12. Regression 12 (V7 - complexity VS. V8 - interconnection) shows high R-square value 0.8882 and low P value 0.006.

8) Complexity criterion is also linearly affected by the conceptual size of DFDs. High R-square value 0.8737 and low P value 0.008 proved this.

9) Complexity criterion is affected by the linear combination of interconnection and the size of DFDs. The high R-square value 0.9298 and the low P value 0.022 verified that.

10) The complexity measure developed in chapter three is a valid measure. The result from regression 14 concluded this. The R-square value is 0.8448 and P value is 0.011. The interesting thing here is that linearly combining interconnection and the size of DFD to predict complexity degree seems better than multiplying them (this is the formula developed in chapter three) to predict the complexity degree. However, both are significant enough to predict complexity in this study.

11) There is no strong evidence about the relationship between complexity and logical cohesion criteria. The regression between V7 (complexity) and V10 (logical cohesion) shows this by the low R-square value 0.4454 and high P value 0.147.

The above conclusions make the formalizing DFD evaluation more possible. We can linearize a DFD by constructing its TR form, calculate the valid measures directly from the TR (such as ATB, APB, INTER, and COM), and report the degree of complexity, interconnection, or ease of the implementation of the DFD. The reported measures can be used as a guide in the decomposition of DFDs.

Chapter 5

Fuzzy Classifications of Evaluating Criteria

As we mentioned before, many software measures can not intuitively provide users a straightforward feeling about the quality of the measured object. For example, if we say McCabe's measure is 25, it usually doesn't say much about the relative complexity level of the measured software code, especially to those managers who are not familiar with software measurement mechanism. This is why after proving the validities of some designed measures in chapter four, this research tries to give users a more intuitive evaluation opinion like 'fairly complex', 'very easy', by applying fuzzy set theory to classify the different levels of the criteria.

This step of the research is to convert the DFDs' measures into the linguistic classification categories of these measures. This is a mapping from measurement ranges to fuzzy concepts in a natural language, for example, mapping McCabe's measure range to a classification concept 'very complex' with a certain grade of membership. We believe that the linguistic classification terms ('very complex' in this example) will usually give users a better feeling about the complexity level of the measured object.

The tool we used in this step of our research is fuzzy set theory. We choose it because 1) natural language concepts (terms) are inherently fuzzy [HERSH76], and 2) fuzzy set theory has been developed to offer a formal treatment of vagueness of natural language concepts.

5.1. Vagueness in a Natural Language and its Proper Representation

The vagueness in a natural language enters in the process of mapping a linguistic term unto a universe. This can be seen from two aspects, one is the vagueness of the boundary and the other one is the ambiguity. For example, when we say that a person is *tall* we really don't know what is the precise height (universe) boundaries for the concept 'tall' (actually, there is no precise boundary). Because of this, some concepts, such as 'tall' and 'fairly tall', may overlap over the universe. For example, for height 1.78 meters, one can describe it either as 'tall' or 'fairly tall'. This is exactly where the ambiguity comes from. Therefore, the boundary of a term is never a point but a region where the term gradually (but not sharply !) moves from being applicable to being nonapplicable.

Linguists have empirically assessed the hypothesis that natural language concepts (terms) can be described more completely and more precisely using the framework of fuzzy sets theory. The conclusions are positive [HERSH76]. We will introduce how to use fuzzy sets to represent the vagueness of natural language concepts in the later sections.

5.2. Basic Concepts of Fuzzy Set Theory

Human beings can understand and operate upon vague natural languages. Computers, however, are extremely rigid and precise information-processing systems. This inherent rigidity severely limits a computer's ability to abstract and generalize fundamental conceptual functions [HERSH76]. Since 1960's, Zadeh and other engineers have developed quantitative techniques for dealing with vagueness in complex systems. The techniques are based on fuzzy set theory, a generalization of the traditional theory of sets.

In the following paragraphs, basic concepts related to fuzzy sets theory are discussed before the formal definition of fuzzy sets is introduced.

Linguistic variables :

The unique feature of fuzzy logic is that it allows systems to contain both numeric and *linguistic variables.* Linguistic variables are the variables whose values can be words or sentences. They are defined as labels of fuzzy sets. For example, linguistic variable *complexity* may take on values of *very-complex, fairly-complex, ..., fairly-simple,* and *very-simple.*

Difference between fuzzy sets and non-fuzzy sets :

In traditional set theory (non-fuzzy set theory), a membership function specifies if an element x is a member of the set X (truth value for ($x \in X$) is 1, if x is) or not (truth value for ($x \in X$) is 0, if x is not). While in fuzzy set theory, the transition from membership to non-membership is seldom a step function. Rather, there is a gradual but specifiable change from membership to non-membership. That is, in fuzzy systems, the grade of membership or the corresponding truth value of the proposition $x \in X$ may take any value in the closed real interval [0.0, 1.0].

Difference between fuzzy membership and probability :

Fuzziness is distinctly different from the uncertainty measured by the probability of an event. The probability indicates how big the chance it will be for an event to occur. That is to say, the probability theory deals with the lack of the knowledge concerning an event occurring in the future. Once this knowledge becomes available, the state of affairs is completely determined. No vagueness is involved. One typical example is a coin toss. The uncertainty of a coin toss resulting in a head has a *certain* probability associated with it. Unlike coin toss, no matter how closely one measures or examines, a concept will apply more to some elements of the universe than others. A good example for this is *baldness.* No matter how carefully we count the number of hairs one man has, this information can not make the boundary between *bald* and *not bald* free of imprecision.

Definition of fuzzy sets :

Let X be a universe of elements (persons, DFDs, or heights) with a generic element of X denoted by x. A fuzzy subset of X, labeled A, is characterized by a membership function, f_A, that associates with each element x in X a real number, $f_A(x)$, in the closed interval from 0.0 to 1.0, which represents the grade of membership of x in A. An element of the fuzzy set A thus can be expressed by the ordered pair :

$$f_A(x) \ / \ x$$

where $f_A(x)$ is the grade of membership of x in A [ZADEH65]. The closer the value of $f_A(x)$ is to 1.0, the higher the grade of membership of x in A. If all the grades are either 0 or 1, then the set becomes non-fuzzy. Therefore, a traditional non-fuzzy set is just a special case of fuzzy sets.

One example of fuzzy sets is *complex* where the membership function specifies the grade of membership of complexity measures in the set labeled *complex*. Representative values might be: f(42)=0.0 f(194.84)=0.3 f(679.2)=0.7 and f(1947.8)=1.0. The fuzzy set *complex* looks like the following :

complex={0.0/42, 0.3/194.84, .., 0.7/679.2, 1.0/1947.8}

Thus, a DFD whose value of complexity measure is 42 clearly is not complex; A DFD whose value of complexity measure is 1947.8 is clearly complex; A DFD whose value of complexity measure is 679.2 is more complex than not complex.

Membership in a fuzzy set is specified by a mapping from the universe to the set in question. This mapping can be performed either by enumeration or by a function. Whatever the method, the result will be that every element in X will have associated with a number corresponding to its grade of membership in that fuzzy set. Once this mapping is specified, the set can be used as a linguistic variable in fuzzy inferences.

The discussion of fuzzy membership can be extended by having the grade of membership itself be a fuzzy set [HERSH76]. For example, if the universe X is a set of DFDs and A is the fuzzy set *complex,* then :

complex = {high/DFD1,medium/DFD2,...,low/DFDn}

here, 'high', 'medium', and 'low' are fuzzy subsets of the universe of possible grade of membership values. For example, *high* can be the following :

high={1.0/1.0,0.9/0.98,0.8/0.9,0.5/0.8,..,0.1/0.5}

A *normal* fuzzy set is a fuzzy set with at least one element x in X with a grade of membership 1.0.

Two fuzzy sets, A and B, are *equal* (A=B), iff $f_A(x) = f_B(x)$ for all x in X.

A fuzzy set A is contained in a fuzzy set B, or B *entails* A, or A is a *subset* of B, iff $f_A(x) \leq f_B(x)$ for all x in X.

The *union* of two fuzzy sets in fuzzy logic is defined by

$$C = A \bigcup B$$

the union fuzzy set C has membership function as following :

$$f_C(x) = Max\ [\ f_A(x),\ f_B(x)\ ],\ x \in X$$

The *intersection* of two fuzzy sets A and B is denoted by

$$C = A \bigcap B$$

the intersection C has the following membership function :

$$f_C(x) = Min\ [\ f_A(x),\ f_B(x)\ ],\ \ x \in X$$

The *algebraic product* of A and B is denoted by AB and is calculated by

$$f_{AB}(x) = f_A(x)\ *\ f_B(x)$$

The *algebraic sum* of A and B is denoted by A + B and is defined by

$$f_{A+B}(x) = f_A(x) + f_B(x)$$

if the sum is less than or equal to one

The *sum* of fuzzy sets is defined as :

$$f_{A \oplus B}(x) = f_A(x) + f_B(x) - f_{AB}(x)$$

[ZADEH65].

Next section, we will introduce how we apply fuzzy set theory in the research and build membership functions for the evaluative phrases used in our survey such as *very-complex, fairly-complex, or more-complex-than-simple.*

5.3. Fuzzy Sets Application in This Research

By looking at Appendix D and Appendix E, row 62 through row 67 are the frequencies of the answers cross all 6 DFDs (objects). In fact, each row represents an evaluative phrase. For example, row 62 on page D-2 represents phrase *very complex,* row 63 *fairly complex,* row 64 *more complex than simple,* and so on. For the normalized survey data, the corresponding frequencies are shown on the same rows, from row 62 through row 67 of Appendix E. There the phrases will correspond to a subrange but not a single value. Therefore, row 62 represents the phrase ' *very complex',* row 63 ' *fairly complex',* and so on. We will use the normalized data to apply fuzzy set theory.

There are basically two ways to build fuzzy membership functions. They are the statistical approach and the linguistic approach. Which one should be used depends on the nature of the data. Statistical approach asks for the satisfaction of certain assumptions and known distribution (or at least, to have the confidence to assume the distribution) of the data [CIVAN86]. Linguistic approach can be used when the data is from a poll or survey. Harry M. Hersh and Alfonso Caramazza have reported the results of building fuzzy membership functions from poll result. They conducted an experiment in Johns Hopkins University to assess the validity of fuzzy sets representation of natural language terms [HERSH76]. In our research, the linguistic approach has been adopted because of the nature of our data.

Since Zadeh (1968) had equated the probability of a fuzzy event with the expected grade of membership of the event, the proportion of *yes* responses for a particular DFD and a particular phrase can be interpreted as the grade of membership for that DFD in the fuzzy set labeled by the phrase. If we furthermore normalize the fuzzy set, we can build up the membership functions for all evaluative phrases in the following steps :

1) Choose the biggest number for each row (phrase) from row 62 through row 67, divide all the values in this row by the chosen number. The results are shown from row 78 through row 83 in Appendix E

2) Rearrange the columns for the results we get from step (1) in the order of the measures of the corresponding criterion (from the smallest to the largest). For example, for complexity criterion (see E-2), use the complexity measures COMs calculated based on six DFDs to rearrange (rank) the columns K through column P from the relatively simplest DFD to the relatively most complex DFD (the original order of the six DFDs is : DFD1, DFD2, DFD3, DFD4, DFD5, DFD6. After rearranging, the new order is : DFD3, DFD2, DFD1, DFD6, DFD5, DFD4). For interconnection criterion (see E-4), using the interconnection measures INTERs calculated from six DFDs to rearrange (rank) columns from the one that has the lowest interconnection measure to the one that has the highest interconnection measure (the new order is : DFD3, DFD1, DFD2, DFD6, DFD5, DFD4).

   It is valid to rearrange the DFDs according to the corresponding measures because we have proved that those measures are valid. That is, they keep the correct order of the DFDs. The result of this step is shown from row 87 through row 92.

3) Drop DFD5 from the rearranged data obtained from step (2). That is, remove the data on column O from the rearranged data. Then, carry out any necessary normalization if needed. The result of this step is displayed from row 96 through row 101. The reason for dropping DFD5 is stated in chapter four (see Conclusion section of chapter four).

4) Plot fuzzy membership functions based on row 96 through row 101. The X-axis is our universe (DFDs' measures) and the Y-axis is the grades of the membership for the elements in universe in certain fuzzy set.

   In the data from step (3), each row now is the fuzzy membership function of the corresponding evaluative phrase. In addition to the measures of the DFDs, rows will be the fuzzy sets of the evaluative phrases. For example, on page E-4, row 96 represents phrase *very-complex* and the values from column K to O of this row are the grades of membership for the DFD's complexity measures in the fuzzy set *very-complex*. The corresponding fuzzy set *very-complex* can be written as :

$$\text{very-complex} = \{\ 0.0/42,\ 0.0/146.76,\ 0.067/194.8,\ 0.067/443.9,\ 1.0/1947.8\ \}$$

   here, the universe is (42, 146.76, 194.8, 443.9, 1947.8), a set of complexity measures of DFDs.

Applying above four steps to the criteria that are in question in this research (they are *complexity, interconnection*, and *ease of implementation)*, We obtained figure 5.1, figure 5.2, and figure 5.3.

Figure 5.1 is the membership functions of *complexity*. Figure 5.1 (a) is the membership function of *very-complex* versus the membership function of *fairly-complex*. Figure 5.1 (b) is the membership function of *more-complex-than-simple* versus the membership function of *more-simple-than-complex*. Figure 5.1 (c) is the membership function of *very-simple* versus the membership function of *fairly simple.* The titles of the figures explained what they are.

From these figures, we can see that with the addition of an linguistic intensifier such as *very,* the graph of the base word moves away from the neutral point toward the extreme. This is exactly the conclusion Harry M. Hersh and Alfonso Caramazza successfully proved in their empirical research.

These figures were produced based on only five points. However, they have really presented reasonable, consistent, and fairly good results. The figures outline basic shapes of the membership functions. In order to get the exact functions, we may need not only get more points but also apply approximation theory to the data. So far, we have not covered these further study yet in this research. But again, the conclusions we had till now has provided a good basis for the further research of this direction.

Suppose we have obtained precise membership functions (either enumeration one or a function) for all the evaluative phrases of DFDs. Once a certain DFD's valid measure is available, say complexity measure COM, we can get its different grades of membership for different levels of complexity. For example, if we have COM=1012.45, we might get the grade of membership 0.75 for this DFD in fuzzy set *very-complex,* the grade of membership 0.98 for this DFD in fuzzy set *fairly-complex,* the grade of membership 0.2 for this DFD in fuzzy set *fairly-simple,* and so on. We can use different approaches to combine them. The union or the sum of fuzzy sets are the examples of possible choices.

Even though the number of objects used in this research is not enough to make the curves of membership functions smooth, the results have been very interesting, very useful and they have given us the courage to continue the further study in this area.

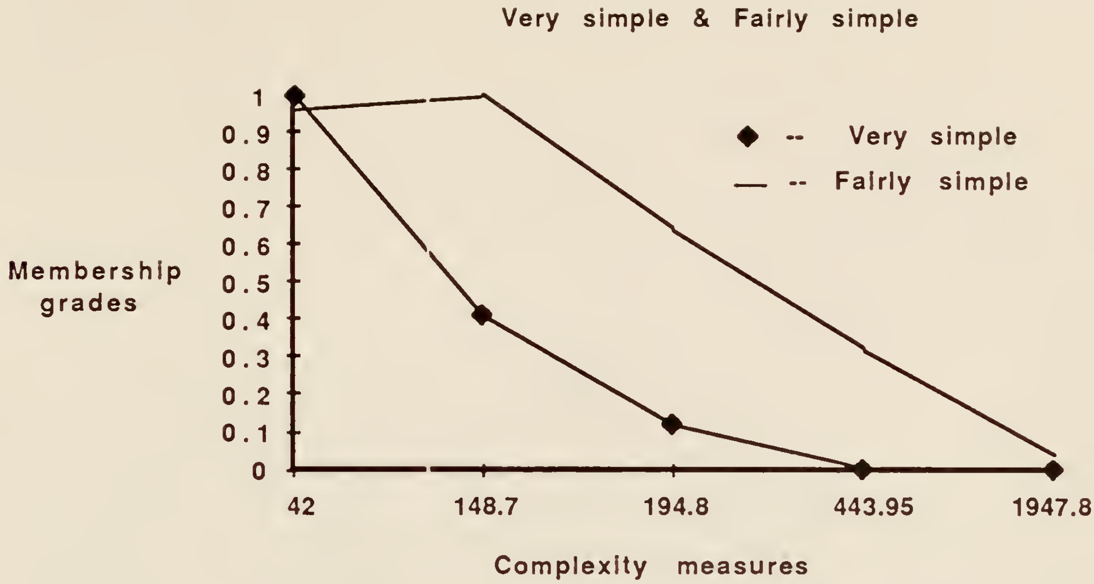


Figure 5.1 (a) F.M.F. of VS & FS for complexity

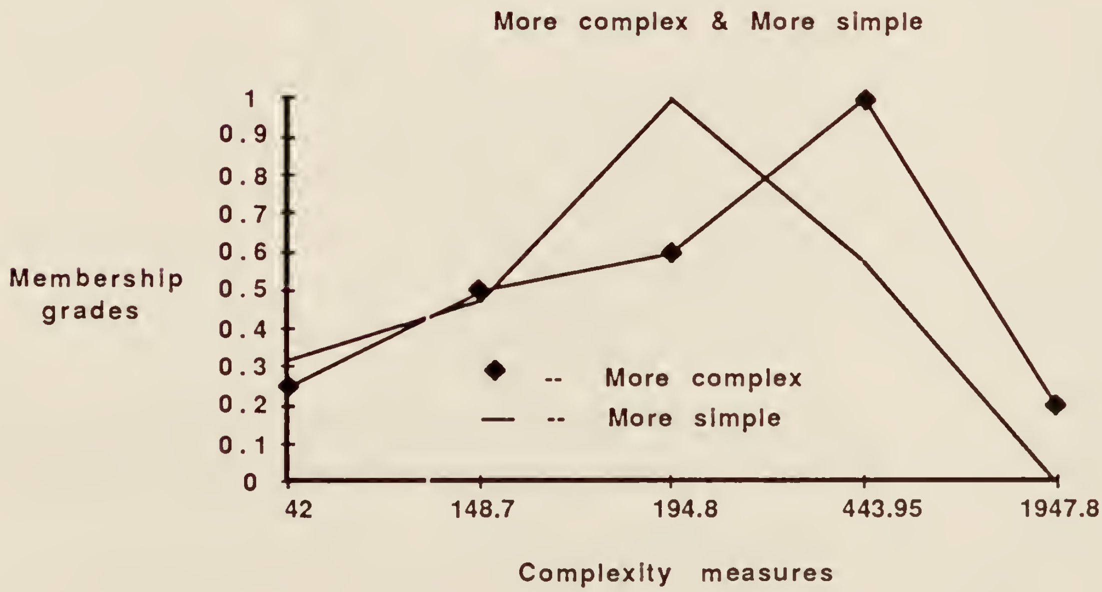


Figure 5.1 (b) F.M.F. of MC & MS for complexity

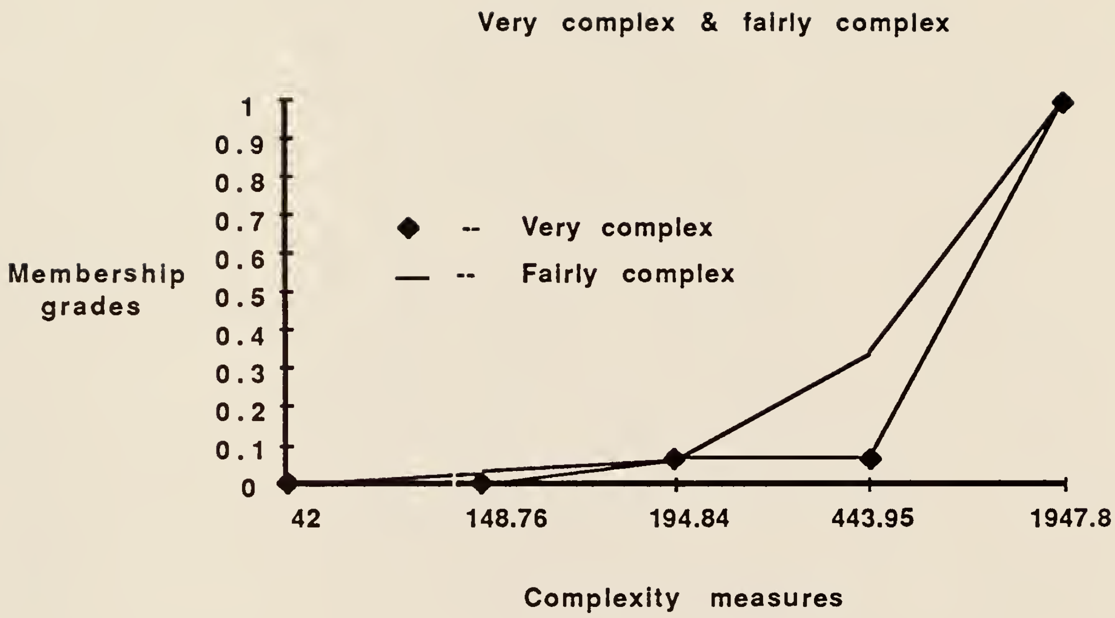


Figure 5.1 (c) F.M.F. of VC & FC for complexity

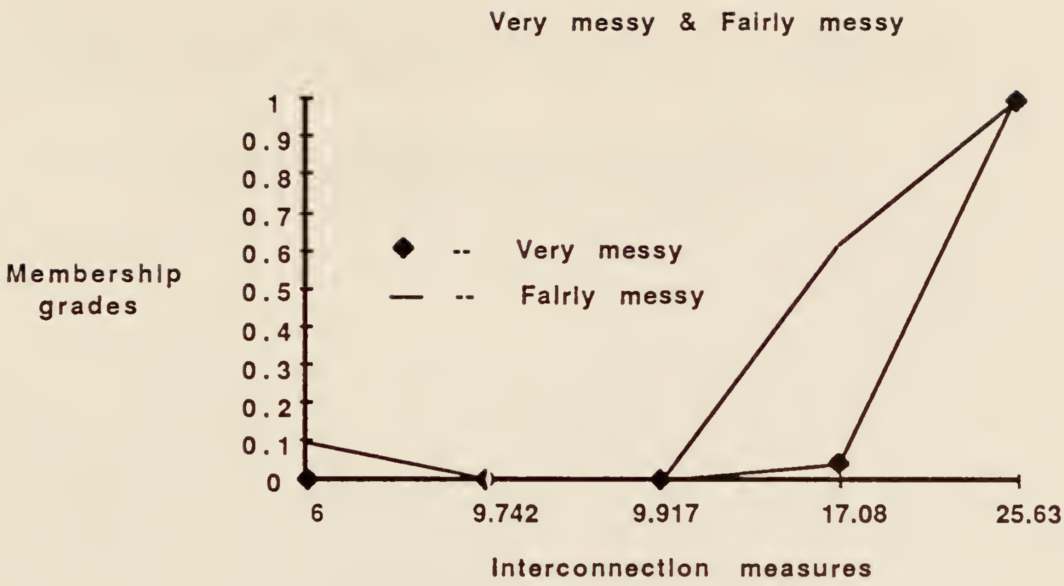


Figure 5.2 (a) F.M.F. of VM & FM for interconnection

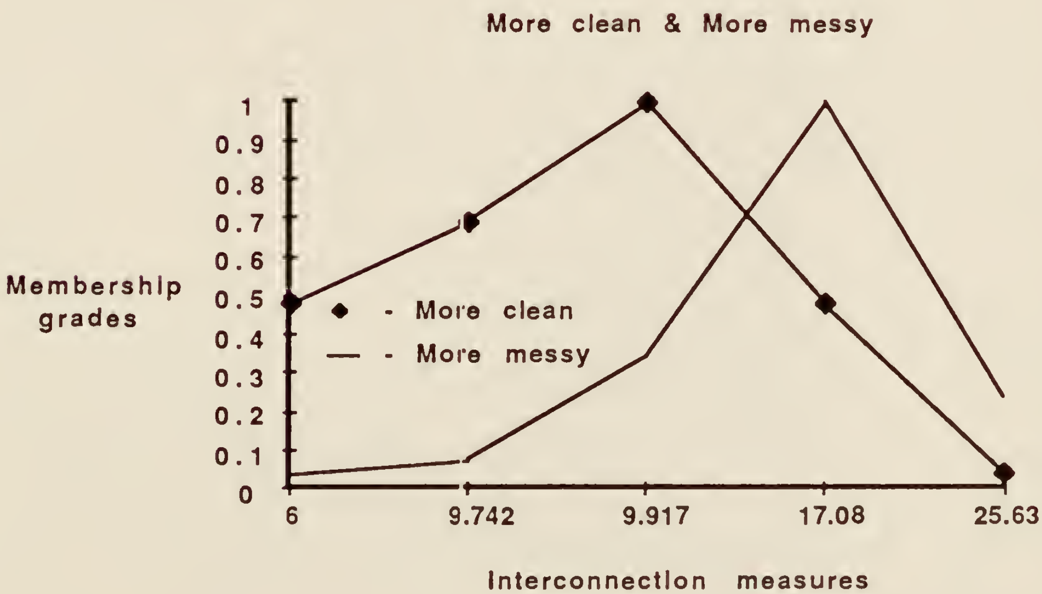


Figure 5.2 (b) F.M.F. of MC & MM for interconnection

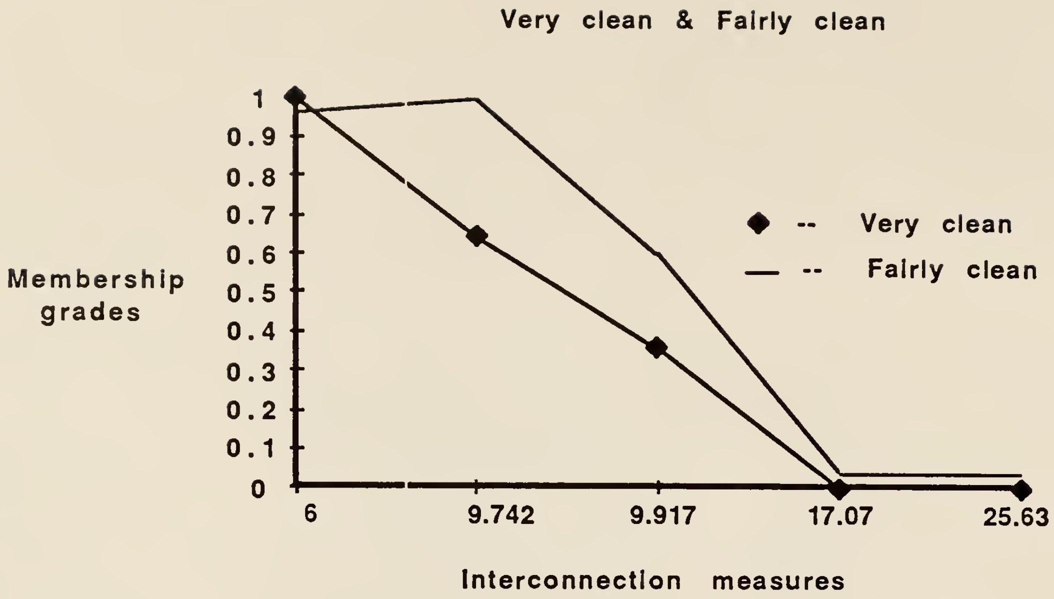


Figure 5.2 (c) F.M.F. of VC & FC for interconnection

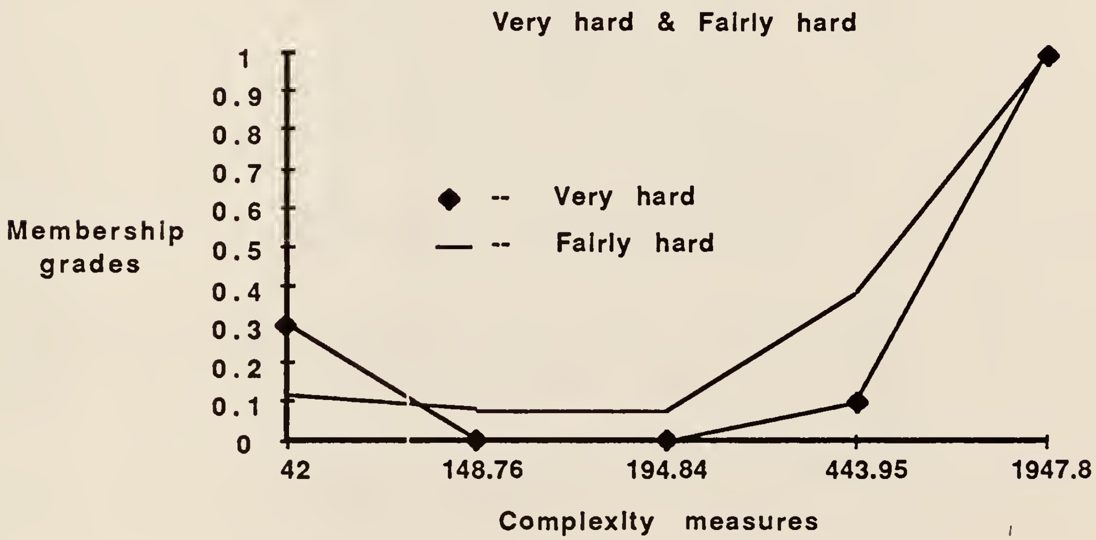


Figure 5.3 (a) F.M.F. of VH & FH for ease of imp.

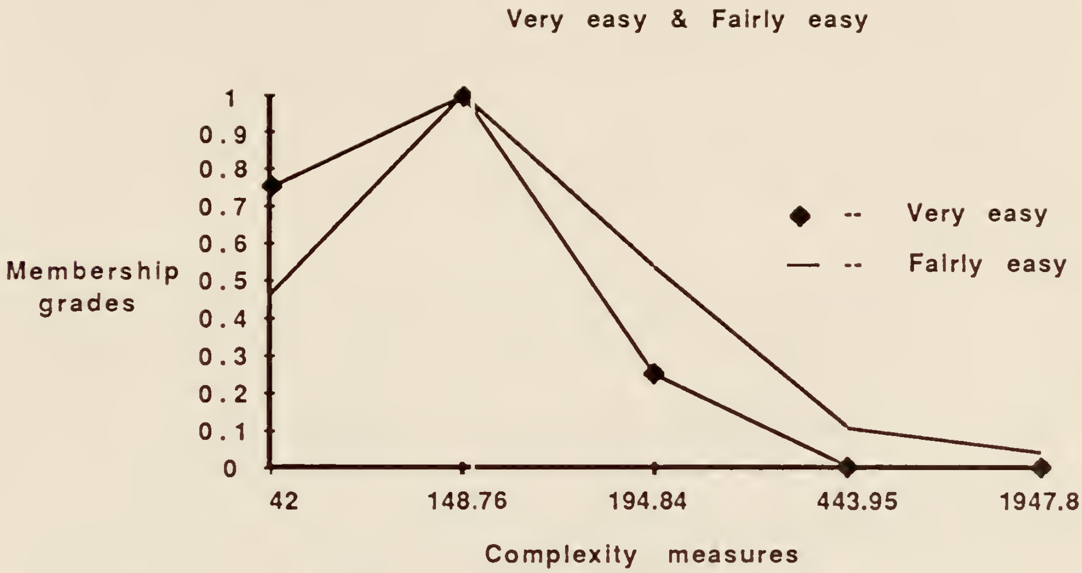


Figure 5.3 (b) F.M.F. of VE & FE for ease of imp.

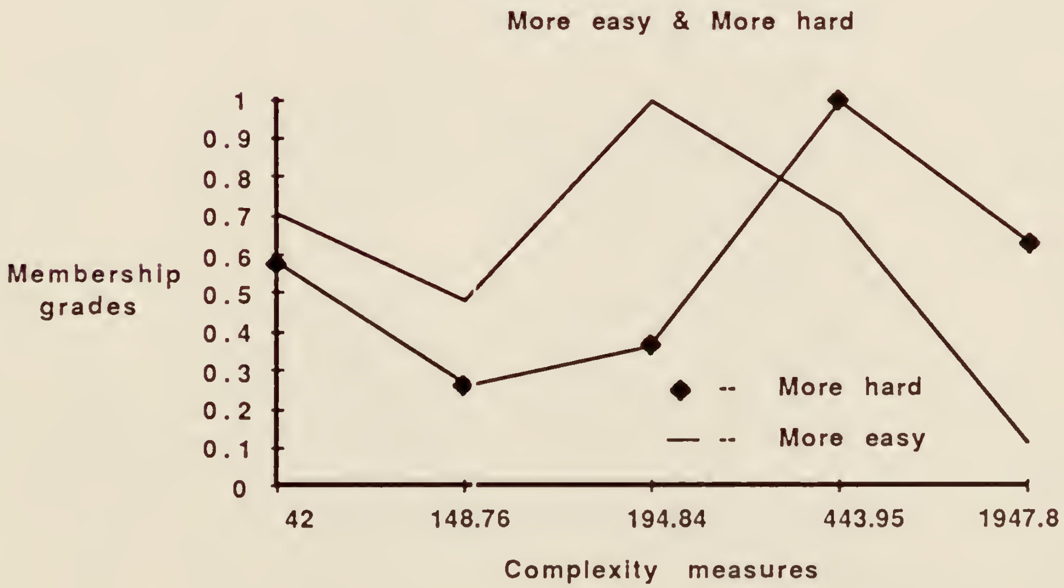


Figure 5.3 (c) F.M.F. of ME & MH for ease of imp.

Chapter 6

Conclusions

The following results have been shown in this research. They are summarized in figure 6.1. From a DFD, we can construct its linear representation TR by applying the converting rules stated in chapter two. After the TR is created, various kind of basic counts introduced in chapter three can be calculated from it. The advanced DFD measures such as ATB, ACB, APB, DS, LD, INTER, and COM can also be constructed. These measures will be further accepted by the fuzzy classification mechanism to get the linguistic evaluations of the DFD. The criteria that currently can be processed in the fuzzy classification mechanism are *interconnection, complexity,* and *ease of implementation* (as it shows in figure 6.1). These classified evaluations can be backfed as a guidance to DFD designers or even to the process of decomposition of DFDs. Once revised DFDs are created, they can be evaluated again. This feedback can be seen from the dotted part of figure 6.1.

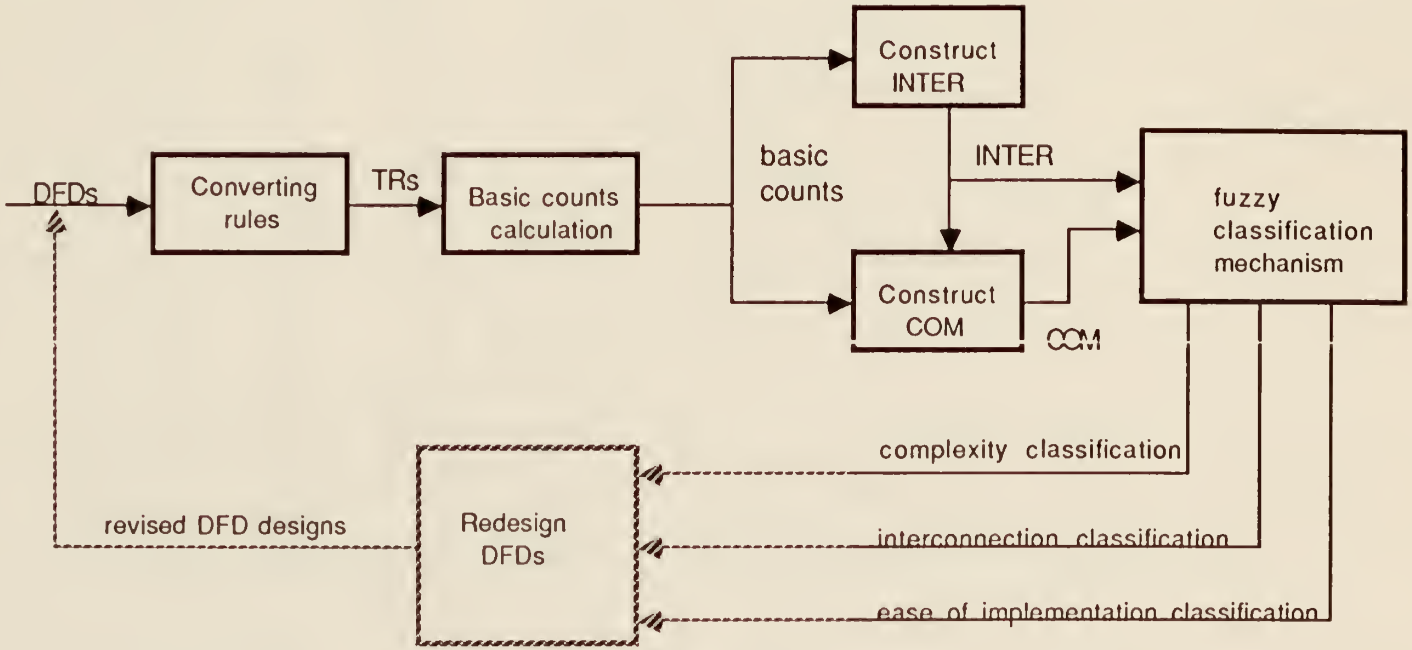


Fig. 6.1 The scheme of the research

Based on the previous chapters, a summary is given in the following sections.

6.1. Advantages

Several advantages can be seen from this research. They are claimed in the following.

1) Textual representation, a useful linearized form of a DFD has built a good foundation for formalizing the evaluation process of DFDs.

2) The structure of the TRs has made it possible to automate the calculation of basic counts needed in constructing advanced DFD measures.

3) The DFD measures developed in chapter three of this thesis have reasonable specificity to provide the information about what is contributing to the measure. Therefore, it will be able to guide DFD designers in how to improve the quality of the DFDs. For example, INTER is constructed by five factors that reflect the different aspects of interconnections such as average token burden or data sharing degree. When the interconnection measure of a DFD is too high, we can find the cause by checking the individual measures and then try to improve the quality.

4) DFD measures of certain criterion together with the fuzzy membership functions of these measures provide a normativeness for the evaluation of DFDs. As we mentioned before, if a measure doesn't provide a norm against which measures can be compared, it is meaningless to apply the measure to the object it measures in isolation. Based on the fuzzy membership functions, we can provide straightforward linguistic judgement such as 'very complex'. This kind of terms have a natural norm based on the human beings' common sense or intuition.

5) This research has explored another possible way to evaluate the software design tool DFD.

6) The scheme (figure 1.1) shows a closed environment of automatic DFD design tool. Even though this research has not touched the feedback part of the scheme, it has provided such a framework which indicates a possible direction of future research.

6.2. Problems

There are several problems revealed in this research.

1) The results might not be general enough to validate the DFD evaluation because 1) they were obtained from a specific survey, and 2) the sample size in this research is not big enough to support general conclusions. There are two possibilities related to this deficiency. One, if we change the group of subjects, we might get different result, and two, if we use bigger sample size, we might also get different result. However, we think that even if the change of the environment may lead to the change of the results, the basic natures of the conclusions we obtained from this research will be the same.

2) Some attempts in this research have failed such as modularity measure design, logical cohesion measure design. The statistical analysis results have showed the failures. The reason for this, we think, is that these criteria themselves have not been understood well enough yet by both survey designers and the subjects. We felt difficulty in designing the questions about them at the stage of preparing the data-collection form of the survey. It means that we ourselves may not have comprehended them clearly. Also, by checking the survey data, we noticed that the distribution of the responses for these criteria shows much more spread than some other one such as interconnection. The subjects seemed to answer questions about these criteria based on different understandings or, alternatively, the criteria might still be defined too vaguely to give people a clear feeling about them.

6.3. Possible Future Works

This research has presented a reasonable scheme to do some further study in this field. The possible trends can be :

1) In order to get more general conclusion, bigger sample sizes, including both the number of subjects and the number of objects, can be applied. It will help not only to increase the confidence about the DFD measures but also to approximate the fuzzy membership functions more reliably and more smoothly.

2) Building up fuzzy membership functions for average token burden (ATB), average connection burden (ACB), average path burden (APB), data sharing degree (DS), and the loop density (LD) will furthermore give DFD designers more intuitive guidance. These measures are the factors of interconnection criterion and interconnection influences complexity and the ease of implementation. The fuzzy classification of them will obviously provide the information about how to improve the compounded measures.

3) Better understanding of other measures related to the quality of DFDs will be necessary, for example, modularity or even some new aspects of DFDs. It will also depend on the development of the whole area of software engineering.

4) Based on the previously mentioned future works, it is possible to formalize the expertise of evaluating DFDs and complete the closed environment of automating the process of DFD designs as shown in figure 1.1.

Bibiography

[ADLER88] Adler, Mike., "An Algebra for Data Flow Diagram Process Decomposition," *IEEE Transactions On Software Engineering,* Vol. 14, No. 2, February 1988

[PRESS82] Pressman, Roger S., "Software Engineering : A Practitioners's Approach," *Computer Engineering, University of Bridgeport,* published by McGraw-Hill Book Company, 1982

[YOURD79] Yourdon, Edward., Constantine, Larry L., "Structured Design : Fundamentals of a Discipline of Computer Program and System Design," published by Yourdon Press, 1979

[DEMAR82] DeMarco, Tom., "Controlling Software Projects : Management, Measurement & Estimation," Forwarded by Barry W. Boehm, published by Yourdon Press, 1982

[HENRY81] Henry, S., Kafura, D., "Software Structure Metrics Based on Information Flow," *IEEE Transactions On Software Engineering,* Vol. SE-7, No. 5, (Sep. 1981), p510-518.

[CHAPI79] Chapin, Ned., "A Measure of Software Complexity," *AFIPS Conference Proceedings : National Computer Conference,* 1979

[TAI84] Tai, Kuo-Chung., "A Program Complexity Metric Based on Data Flow Information in Control Graphs," *IEEE Transactions on Software Engineering,* 1984, p239-249.

[TROY81] Troy, Douglas A., Zweben, Stuart H., "Measuring the Quality of Structured Designs," *The journal of Systems and Software,* No. 2, P113-120, 1981

[MELTO88] Melton, Austin C., Baker, Albert L., Bieman, James M., Gustafson, David A., "A Mathematical Perspective for Software Measures Research," *Report TR-CS-88-6* Department of Computing and Information Sciences, Kansas State University, April 1988

[CHEN88] Chen, Ying-Chi., "Tha Analysis of the Software Process Model," *Master's Thesis,* Department of Computing and Information Sciences, Kansas State University, 1988

[STAND87] "Software Engineering Standards," published by The Institute of Electrical and Electronics Engineers, Inc, May 20, 1987

[KEARN86] Kearney, Joseph K., Sedlmeyer, Robert L., Thompson, William B., Michael, Gray A., and Adler, Michael A., "Software Complexity Measurement," *Communication of the ACM* Vol. 29, No. 11 (November 1986), p1044-1050.

[HERSH76] Hersh, Harry M., Caramazza, Alfonso, "A Fuzzy Set Approach to Modifiers and Vagueness in Natural Language," *Journal of Experimental Psychology: General,* Vol. 105, No. 3, 1976, p254-276.

[ZADEH65] Zadeh, L.A., "Fuzzy Sets," *Information and Control,* Vol. 8, New York: Academic Press (1965), p338-353.

[GORZA87] Gorzalczany, Marian B., "A Method of Inference in Approximate Reasoning Based on Interval-valued Fuzzy Sets," *Fuzzy Sets and Systems,* Vol. 21, 1987, p1-17.

[CIVAN86] Civanlar, Reha M., and Trussell, Joel H., "Constructing Membership Functions Using Statistical Data," *Fuzzy Sets and Systems,* Vol. 18, 1986, p1-13.

[HENRY811] Henry, S., Kafura, D., and Harris, K., "On the Relationships Among Three Software Metrics," *Performance Evaluation Review,* Vol. 10, No. 1, 1981, p3-10.

[HENRY812] Henry, S., and Kafura, D., "Software Structure Metrics Based on Information Flow," *IEEE Trans. on Software Engineering,* Vol. SE-7, No. 5, Sept. 1981, p510-518.

[OVIED80] Oviedo, E. I., "Control Flow, Data Flow, and Program Complexity," *Proceedings of COMPSAC, 1980, p146-152.*

[IYENG82] Iyengar, S., Parameswaran, N., and Fuller, J., "A Measure of Logical Complexity of Programs," *Computer Languages,* Vol. 7, 1982, p147-160.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
1									Basic counts for DFD-1							
2																
3				UM	SM	LM	UI	CIi	UO	COi	CTM	UIC	CICi	CIC	UIV	CIV
4																
5	Measures			7	6	0	4		3			2		3	9	17
6																
7	Modules	1									2					
8		2									5					
9		3									3					
10		4									3					
11		5									2					
12		6									2					
13		7									2					
14																
15	Ex inputs															
16		a						1								
17		b						3								
18		c						1								
19		d						3								
20		Sum1						8								
21	Ex outputs	x								1						
22		y								1						
23		z								4						
24		Sum2								6						
25	Interconnects	M1											1			
26		M2											2			
27		Sum3											3			
28	Paths:															
29	a1x	1														
30	b2y	2														
31	b2z	3														
32	c2y	4														
33	c2z	5														
34	d2y	6														
35	d2z	7														
36	d3M1 5z	8														
37	b3M1 5z	9														
38	d4M2 6z	10														
39	d4M2 7z	11														
40	b4M2 6z	12														
41	b4M2 7z	13														

	Q	R	S	T	U	V	W	X	Y	Z
1				Basic counts for DFD-1						
2										
3	P	CW	NPMi	DPi	DP	CL	LLi	CAW	CAD	CSIZE
4										
5	13	4			5	0	N	1.89	3.9	20
6										
7			1							
8			6							
9			2							
10			4							
11			2							
12			2							
13			2							
14										
15										
16										
17										
18										
19										
20										
21										
22										
23										
24										
25										
26										
27										
28										
29				3						
30				3						
31				3						
32				3						
33				3						
34				3						
35				3						
36				5						
37				5						
38				5						
39				5						
40				5						
41				5						
42										

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
1							Basic counts for DFD-2									
2																
3				UM	SM	LM	UI	CIi	UO	COi	CTM	UIC	CICi	CIC	UIV	CIV
4																
5	Measures			4	3	0	2		1			3		3	6	10
6																
7	Modules	1									3					
8		2									3					
9		3									3					
10		4									4					
11	Ex inputs															
12		a						3								
13		b						3								
14		Sum1						6								
15	Ex outputs	x								1						
16		Sum2								1						
17	Interconnects	M1											1			
18		M2											1			
19		M3											1			
20		Sum3											3			
21	Paths:															
22	a1M1 4x	1														
23	b1M1 4x	2														
24	a2M2 4x	3														
25	b2M2 4x	4														
26	a3M3 4x	5														
27	b3M3 4x	6														

	Q	R	S	T	U	V	W	X	Y	Z
1			Basic counts for DFD-2							
2										
3	P	CW	NPMi	DPi	DP	CL	LLi	CAW	CAD	CSIZE
4										
5	6	3			5	0	N	1.67	5	15
6										
7			2							
8			2							
9			2							
10			6							
11										
12										
13										
14										
15										
16										
17										
18										
19										
20										
21										
22				5						
23				5						
24				5						
25				5						
26				5						
27				5						

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
2							Basic counts for DFD-3									
3																
4				UM	SM	LM	UI	CIi	UO	COi	CTM	UIC	CICi	CIC	UIV	CIV
5																
6	Measures			3	0	0	1		1			2		2	4	4
7																
8	Modules	1									2					
9		2									2					
10		3									2					
11	Ex inputs															
12		a						1								
13		Sum1						1								
14	Ex outputs	x								1						
15		Sum2								1						
16	Interconnects	M1											1			
17		M2											1			
18		Sum3											2			
19	Paths:															
20	a1M1 2M2 3x	1														

	Q	R	S	T	U	V	W	X	Y	Z
1				Basic counts for DFD-3						
2										
3	P	CW	NPMi	DPi	DP	CL	I_Li	CAW	CAD	CSIZE
4										
5	1	1			7	0	N	1.00	7.00	7.00
6										
7			1							
8			1							
9			1							
10										
11										
12										
13										
14										
15										
16										
17										
18										
19				7						

	A	B	C	D	E	F	G	H	I	J	K	L	M
1		Basic counts for DFD-4											
2													
3				UM	SM	LM	UI	CIi	UO	COi	CTM	UIC	CICi
4													
5	Measures			6	4	4	6		6			5	
6													
7	Modules	1									6		
8		2									6		
9		3									2		
10		4									5		
11		5									6		
12		6									3		
13	Ex inputs												
14		a						1					
15		b						1					
16		c						2					
17		d						1					
18		e						1					
19		f						2					
20		Sum1						8					
21	Ex outputs	v								1			
22		w								1			
23		x								1			
24		e								1			
25		f								1			
26		z								1			
27		Sum2								6			
28	Interconnects	M1											1
29		M2											1
30		M3											1
31		M4											1
32		e											4
33		Sum3											8
34	Paths:												
35	a1v	1											
36	b1v	2											
37	a1M2 2e3w	3											
38	b1M2 2e3w	4											
39	a1M2 2e4x	5											
40	b1M2 2e4x	6											
41	a1M2 2e4e	7											
42	b1M2 2e4e	8											
43	a1M2 2M4 6f	9											
44	b1M2 2M4 6f	10											
45	a1M2 2e4e5e6f	11											
46	b1M2 2e4e5e6f	12											
47	a1M2 2M3 1M2 2e3w	13											
48	b1M2 2M3 1M2 2e3w	14											
49	a1M2 2M3 1M2 2M4 6f	15											

	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
1						Basic counts for DFD-4							
2													
3	CIC	UIV	CIV	P	CW	NPMi	DPi	DP	CL	LLi	CAW	CAD	CSIZE
4													
5	8	17	22	60	4			19	2		1.29	9	76
6													
7						36							
8						46							
9						6							
10						30							
11						22							
12						20							
13													
14													
15													
16													
17													
18													
19													
20													
21													
22													
23													
24													
25													
26													
27													
28													
29													
30													
31													
32													
33													
34													
35							3						
36							3						
37							7						
38							7						
39							7						
40							7						
41							7						
42							7						
43							7						
44							7						
45							11						
46							11						
47							11						
48							11						
49							11						

	A	B	C	D	E	F	G	H	I	J	K	L	M
50	b1M2 2M3 1M2 2M46f	16											
51	a1M2 2M3 1M2 2e4x	17											
52	b1M2 2M3 1M2 2e4x	18											
53	a1M2 2M3 1M2 2e4e	19											
54	b1M2 2M3 1M2 2e4e	20											
55	a1M2 2M3 1M2 2e4e5e6f	21											
56	b1M2 2M3 1M2 2e4e5e6f	22											
57	a1M2 2M3 1M2 2e4e5z	23											
58	b1M2 2M3 1M2 2e4e5z	24											
59	a1M2 2e4e5M1 1M2 2e3w	25											
60	b1M2 2e4e5M1 1M2 2e3w	26											
61	a1M2 2e4e5M1 1M2 2M4 6f	27											
62	b1M2 2e4e5M1 1M2 2M4 6f	28											
63	a1M2 2e4e5M1 1M2 2e4x	29											
64	b1M2 2e4e5M1 1M2 2e4x	30											
65	a1M2 2e4e5M1 1M2 2e4e	31											
66	b1M2 2e4e5M1 1M2 2e4e	32											
67	a1M2 2e4e5M1 1M2 2e4e5e6f	33											
68	b1M2 2e4e5M1 1M2 2e4e5e6f	34											
69	a1M2 2e4e5M1 1M2 2e4e5z	35											
70	b1M2 2e4e5M1 1M2 2e4e5z	36											
71	c2e3w	37											
72	f2e3w	38											
73	c2e4x	39											
74	f2e4x	40											
75	c2e4e	41											
76	f2e4e	42											
77	c2M4 6f	43											
78	f2M4 6f	44											
79	c2e4e5e6f	45											
80	f2e4e5e6f	46											
81	c2e4e5z	47											
82	f2e4e5z	48											
83	d4x	49											
84	e4x	50											
85	d4e	51											
86	e4e	52											
87	d4e5e6f	53											
88	e4e5e6f	54											
89	d4e5z	55											
90	e4e5z	56											
91	c5e6f	57											
92	f5e6f	58											
93	c5z	59											
94	f5z	60											
95	Loops:												
96	M3 1 M2 2 M3	1											
97	M1 1 M2 2 e 4 e 5 M1	2											

	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
50							11						
51							11						
52							11						
53							11						
54							11						
55							15						
56							15						
57							13						
58							13						
59							15						
60							15						
61							15						
62							15						
63							15						
64							15						
65							15						
66							15						
67							19						
68							19						
69							17						
70							17						
71							5						
72							5						
73							5						
74							5						
75							5						
76							5						
77							5						
78							5						
79							9						
80							9						
81							7						
82							7						
83							3						
84							3						
85							3						
86							3						
87							7						
88							7						
89							5						
90							5						
91							5						
92							5						
93							3						
94							3						
95													
96										2			
97										4			

	A	B	C	D	E	F	G	H	I	J	K	L	M	N
1								Basic counts for DFD-5						
2														
3				UM	SM	LM	UI	CIi	UO	COi	CTM	UIC	CICi	CIC
4														
5	Measures			5	4	5	1		3			5		8
6														
7	Modules	1									8			
8		2									3			
9		3									3			
10		4									3			
11		5									3			
12	Ex inputs													
13		a						1						
14		Sum1						1						
15	Ex outputs	a								1				
16		x								1				
17		y								4				
18		Sum2								6				
19	Interconnects	M1											1	
20		M2											1	
21		M3											1	
22		M4											1	
23		x											4	
24		Sum3											8	
25	Paths:													
26	a1a	1												
27	a1x	2												
28	a1M1 2y	3												
29	a1M2 3y	4												
30	a1M3 4y	5												
31	a1M4 5y	6												
32	a1M1 2x1M1 2y	7												
33	a1M1 2x1M2 3y	8												
34	a1M1 2x1M3 4y	9												
35	a1M1 2x1M4 5y	10												
36	a1M2 3x1M1 2y	11												
37	a1M2 3x1M2 3y	12												
38	a1M2 3x1M3 4y	13												
39	a1M2 3x1M4 5y	14												
40	a1M3 4x1M1 2y	15												
41	a1M3 4x1M2 3y	16												
42	a1M3 4x1M3 4y	17												
43	a1M3 4x1M4 5y	18												
44	a1M4 5x1M1 2y	19												
45	a1M4 5x1M2 3y	20												
46	a1M4 5x1M3 4y	21												

	O	P	Q	R	S	T	U	V	W	X	Y	Z
1						Basic counts for DFD-5						
2												
3	UIV	CIV	P	CW	NPMi	DPi	DP	CL	LLi	CAW	CAD	CSIZE
4												
5	9	15	22	4			9	4		1.67	8.14	36.00
6												
7					22							
8					5							
9					5							
10					5							
11					5							
12												
13												
14												
15												
16												
17												
18												
19												
20												
21												
22												
23												
24												
25												
26						3						
27						3						
28						5						
29						5						
30						5						
31						5						
32						9						
33						9						
34						9						
35						9						
36						9						
37						9						
38						9						
39						9						
40						9						
41						9						
42						9						
43						9						
44						9						
45						9						
46						9						

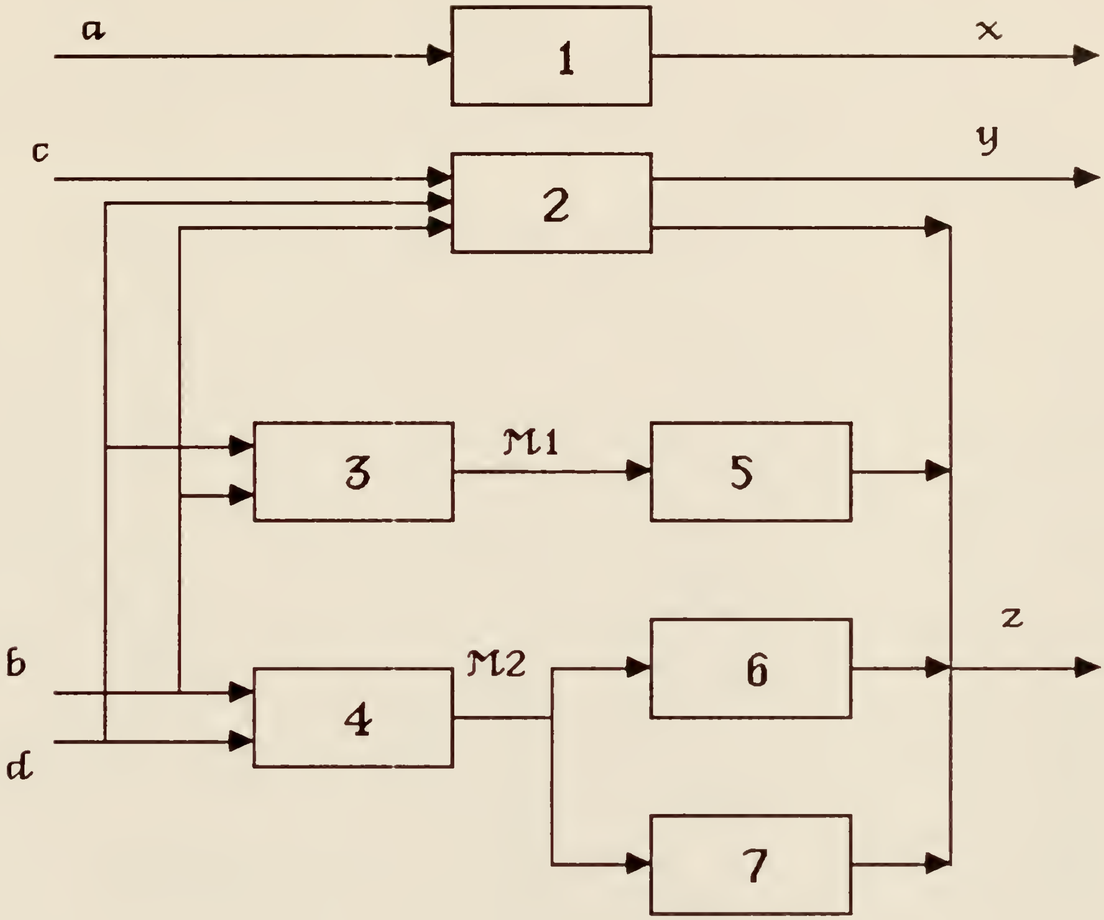
	A	B	C	D	E	F	G	H	I	J	K	L	M	N
47	a1M4 5x1M4 5y	22												
48	Loops:													
49	x 1 M1 2 x	1												
50	x 1 M2 3 x	2												
51	x 1 M3 4 x	3												
52	x 1 M4 5 x	4												

	O	P	Q	R	S	T	U	V	W	X	Y	Z
47						9						
48												
49									2			
50									2			
51									2			
52									2			

	A	B	C	D	E	F	G	H	I	J	K	L	M
1						Basic counts for DFD-6							
2													
3				UM	SM	LM	UI	CIi	UO	COi	CTM	UIC	CICi
4													
5	Measures			4	2	3	3		3			4	
6													
7	Modules	1									4		
8		2									6		
9		3									4		
10		4									2		
11	Ex inputs												
12		a						1					
13		b						1					
14		c						2					
15		Sum1						4					
16	Ex outputs	x								1			
17		y								1			
18		c								1			
19		Sum2								3			
20	Interconnects	M1											2
21		M2											1
22		a											1
23		a											1
24		Sum3											5
25	Paths:												
26	a 1 x	1											
27	a 1 a 2 M2 4 c	2											
28	a 1 a 2 a 3 y	3											
29	b 2 M2 4 c	4											
30	b 2 a 3 y	5											
31	c 2 M2 4 c	6											
32	c 2 a 3 y	7											
33	c 3 y	8											
34	a 1 a 2 M1 1 a 2 M2 4 c	9											
35	a 1 a 2 M1 1 a 2 a 3 y	10											
36	a 1 a 2 a 3 M1 1 a 2 M2 4 c	11											
37	a 1 a 2 a 3 M1 1 a 2 a 3 y	12											
38	Loops:												
39	M1 1 a 2 M1	1											
40	M1 1 a 2 a 3 M1	2											

	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
1						Basic counts for DFD-6							
2													
3	CIC	UIV	CIV	P	CW	NPMi	DPi	DP	CL	LLi	CAW	CAD	CSIZE
4													
5	5	10	12	12	2			13	2		1.20	7.33	26.00
6													
7						7							
8						10							
9						7							
10						5							
11													
12													
13													
14													
15													
16													
17													
18													
19													
20													
21													
22													
23													
24													
25													
26							3						
27							7						
28							7						
29							5						
30							5						
31							5						
32							5						
33							3						
34							11						
35							11						
36							13						
37							13						
38													
39										2			
40										3			

DFD - 1

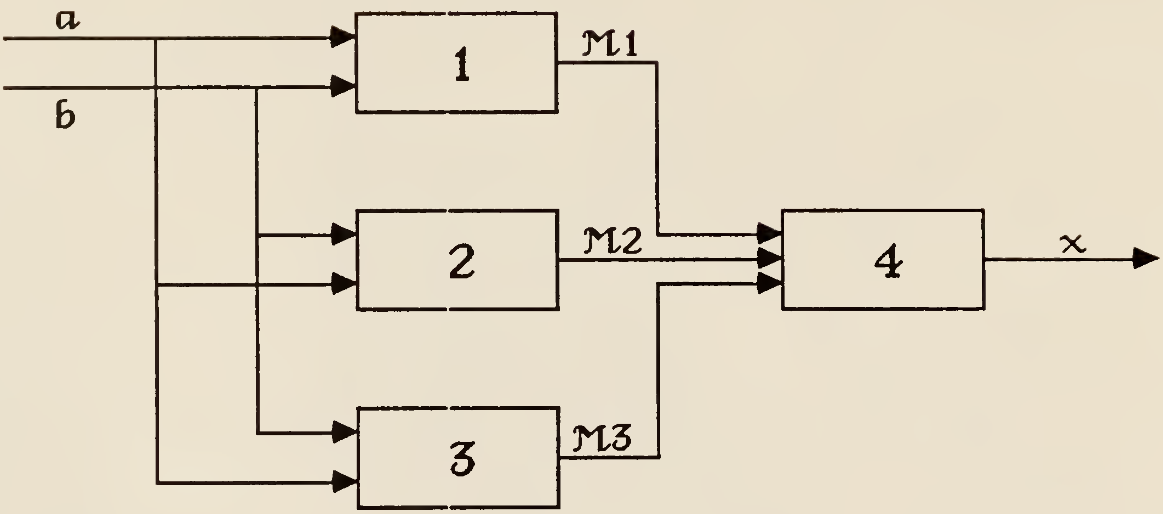


'a' $\xrightarrow{1}$ "x"

'b','d' { 'c' $\xrightarrow{2}$ "y" | : $\xrightarrow{3}$ (M1 $\xrightarrow{5}$) :

$\xrightarrow{4}$ (M2 { $\xrightarrow{6}$: $\xrightarrow{7}$ }) } "z"

DFD - 2

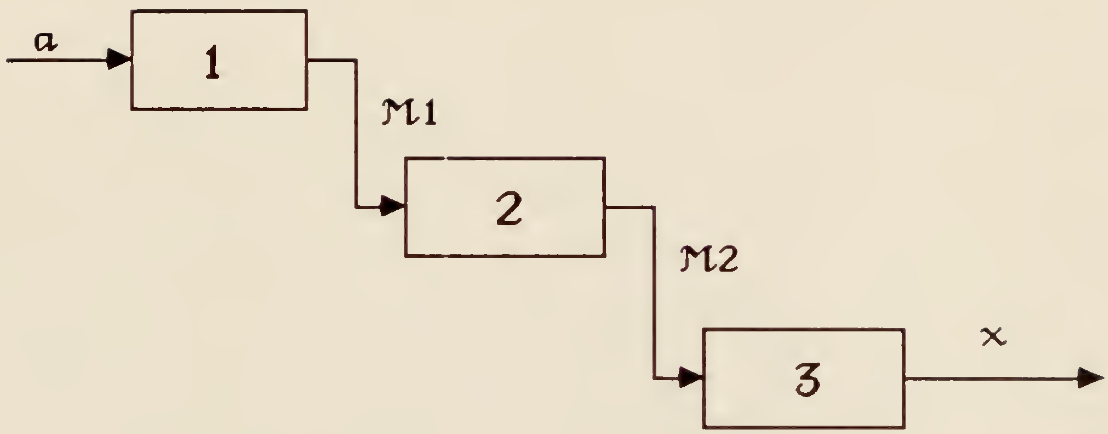


TR :

```
             1                    2
'a','b' { ---> [ M1 ] : ---> [ M2 ] :

      3          <--  <--  4
   ---> ( M3, M2, M1 ---> "x" ) }
```

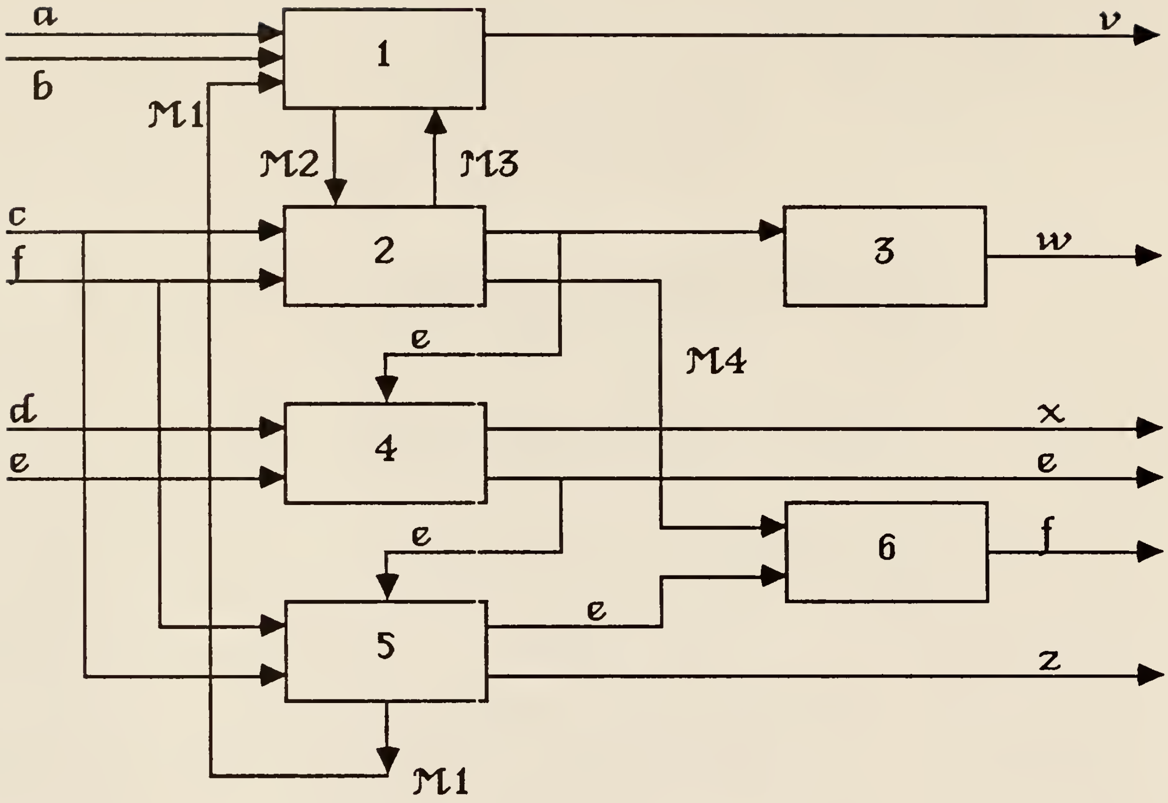
DFD - 3



TR :

```
      1              2               3
'a'  ---> ( M1  --->  ( M2  ---> "x" ))
```

DFD - 4



TR :

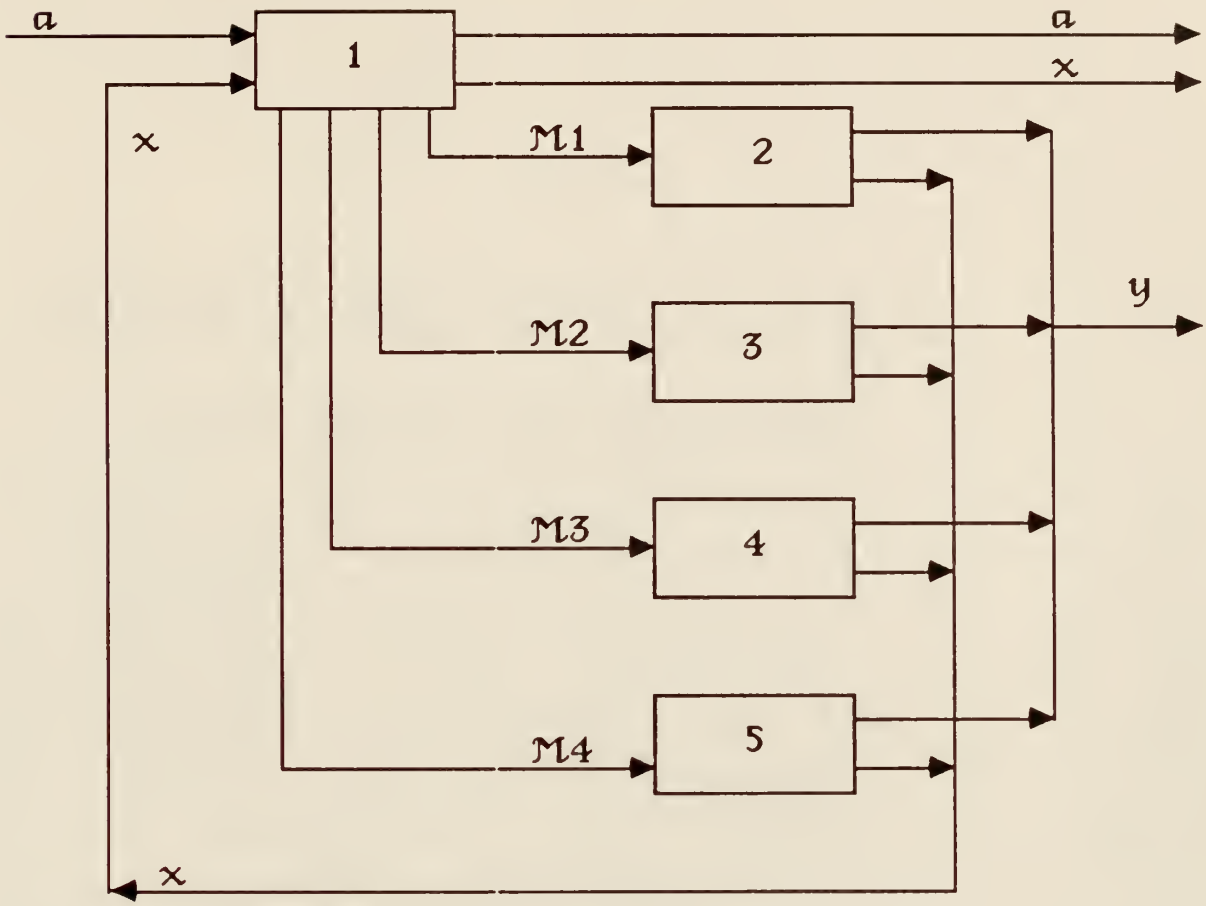
                        1   2
'a','b',M1,M3 ---> (M2,'c','f' ---> (M3)*

        3   <- 4
|| (e ---> "w") || ([M4],('d','e',e ---> "x"|

        <-   5   <--   6
"e" | (e,'c','f' ---> (M1)* || "z" || (e, M4 --->

"f"))))) || "v"

DFD - 5



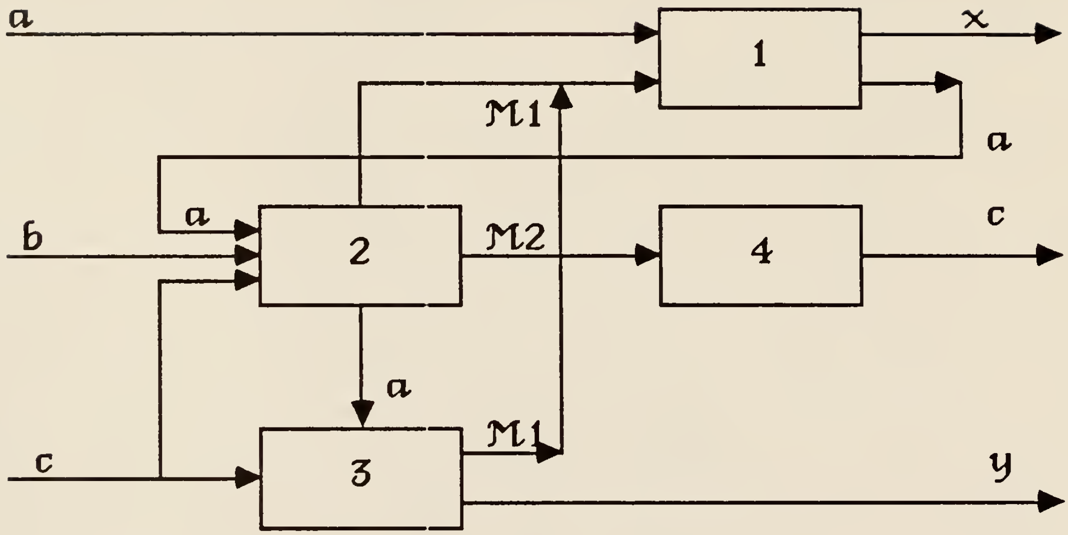
TR :

```
              1                  2            3
'a', x ---> ( { M1 ---> : M2 ---> :

        4              5
M3 ---> : M4 ---> } (x)* || "y" ) ||


"a" || "x"
```

DFD - 6



TR :

'a', M1 $\xrightarrow{1}$ (a,'b','c' $\xrightarrow{2}$ (M1)* ||

(a,'c' $\xrightarrow{3}$ (M1)* || "y") || (M2 $\xrightarrow{4}$

"c")) || "x"

Appendix C - Data-collection Form

The following is the criteria listed in the order the questions will be asked.

0. Complexity
1. Consistency
2. Interconnections
3. Modularity of the expansion
4. Cohesion
5. Clarity
6. Ease of implementation
7. Complexity

Before each question is asked, the definition of the related certerion will be given. Please read the definition carefully first so that you can answer the questions in a correct way.

COMPLEXITY

Complexity of the expansion is the STRUCTURAL complexity of the diagram (but not concerning with the psychological complexity of the diagram). High complexity implies that the processing implied by the diagram is not simple.

1. Do you think this expansion is

a) very complex
b) fairly complex
c) more complex than simple
d) more simple than complex
e) fairly simple
f) very simple

exp-1	exp-2	exp-3	exp-4	exp-5	exp-6
(a)	(a)	(a)	(a)	(a)	(a)
(b)	(b)	(b)	(b)	(b)	(b)
(c)	(c)	(c)	(c)	(c)	(c)
(d)	(d)	(d)	(d)	(d)	(d)
(e)	(e)	(e)	(e)	(e)	(e)
(f)	(f)	(f)	(f)	(f)	(f)

CONSISTENCY

A box and its expansion are consistent if they both imply the same process.

2. Consider the consistency between the process implied by the original box and the process implied by the expansion, these processes are

a) very consistent
b) fairly consistent
c) more consistent than inconsistent
d) more inconsistent than consistent
e) fairly inconsistent
f) very inconsistent

exp-1	exp-2	exp-3	exp-4	exp-5	exp-6
(a)	(a)	(a)	(a)	(a)	(a)
(b)	(b)	(b)	(b)	(b)	(b)
(c)	(c)	(c)	(c)	(c)	(c)
(d)	(d)	(d)	(d)	(d)	(d)
(e)	(e)	(e)	(e)	(e)	(e)
(f)	(f)	(f)	(f)	(f)	(f)

INTERCONNECTIVITY

Interconnection depends on the interface complexity among boxes of the expansion. Low interconnnection means that each box in expansion (module) should be easy to develop independently, i.e., interconnection is a measure of the relative independence among modules.

We can view interconnections in two aspects :

1) The number of interconnections in the graph
2) How are they related together (in a simple fashion or in a complicated way)

Usually, the larger number of interconnections will lead to a more messy-looking graph and fewer will lead to a clearer one. But attention must be paid to the following fact :

Small number of interconnections CAN be related in a complicated way (messy) and a lot of interconnections MIGHT be related in a logically simple (clean) fashion.

3. Observing the interconnections among the modules in the expansion, they are

a) very clean
b) fairly clean
c) more clean than messy
d) more messy than clean
e) fairly messy
f) very messy

exp-1	exp-2	exp-3	exp-4	exp-5	exp-6
(a)	(a)	(a)	(a)	(a)	(a)
(b)	(b)	(b)	(b)	(b)	(b)
(c)	(c)	(c)	(c)	(c)	(c)
(d)	(d)	(d)	(d)	(d)	(d)
(e)	(e)	(e)	(e)	(e)	(e)
(f)	(f)	(f)	(f)	(f)	(f)

MODULARITY

Modularity is the extent to which software is composed of discrete components such that a change to one component has minimal impact on other components.

4. Modularity of the expansion is

a) very good
b) fairly good
c) better than in between
d) worse than in between
d) fairly poor
e) very poor

exp-1	exp-2	exp-3	exp-4	exp-5	exp-6
(a)	(a)	(a)	(a)	(a)	(a)
(b)	(b)	(b)	(b)	(b)	(b)
(c)	(c)	(c)	(c)	(c)	(c)
(d)	(d)	(d)	(d)	(d)	(d)
(e)	(e)	(e)	(e)	(e)	(e)
(f)	(f)	(f)	(f)	(f)	(f)

COHESION

Cohesion is a measure of the relative functional strength of a module. We can view cohesion of the expansion from two stand points :

1) for each module in the expansion, consider the functional cohesion, i.e., a highly cohesive module should (ideally) do just one task. High functional cohesion modules have singularity of tasks.

2) for the whole expansion, consider the cohesion of the whole diagram,i.e.,see how strongly the modules in the expansion are related to each other and to the task of the original box. Low cohesion diagrams do many unrelated tasks.

Note: the questions concerning with different aspects of cohesion will be given seperately.

5. The average functional cohesion of modules in the expansion is

a) very good
b) fairly good
c) better than in between
d) worse than in between
e) fairly poor
f) very poor

exp-1	exp-2	exp-3	exp-4	exp-5	exp-6
(a)	(a)	(a)	(a)	(a)	(a)
(b)	(b)	(b)	(b)	(b)	(b)
(c)	(c)	(c)	(c)	(c)	(c)
(d)	(d)	(d)	(d)	(d)	(d)
(e)	(e)	(e)	(e)	(e)	(e)
(f)	(f)	(f)	(f)	(f)	(f)

6. The overall cohesion of the whole diagram is

a) very good
b) fairly good
c) better than in between
d) worse than in between
e) fairly poor
f) very poor

exp-1	exp-2	exp-3	exp-4	exp-5	exp-6
(a)	(a)	(a)	(a)	(a)	(a)
(b)	(b)	(b)	(b)	(b)	(b)
(c)	(c)	(c)	(c)	(c)	(c)
(d)	(d)	(d)	(d)	(d)	(d)
(e)	(e)	(e)	(e)	(e)	(e)
(f)	(f)	(f)	(f)	(f)	(f)

CLARITY

Clarity is related to how easily the expansion would be understood. It can be viewed from two aspects :

a) processes

By observing the expansion from the original box into the expandede diagram , does the process seem clear or not ?

b) details

By observing the expanded diagram , does it provide detail enough information to reflect the meaning of the original box ?

Note: The two questions concerning with the two aspects of clarity will also be given seperately.

7. Observing the way in which the original box has been expanded, it is ____________ to understand the PROCESS.

a) very easy
b) fairly easy
c) more easy than hard
d) more hard than easy
e) fairly hard
f) very hard

exp-1	exp-2	exp-3	exp-4	exp-5	exp-6
(a)	(a)	(a)	(a)	(a)	(a)
(b)	(b)	(b)	(b)	(b)	(b)
(c)	(c)	(c)	(c)	(c)	(c)
(d)	(d)	(d)	(d)	(d)	(d)
(e)	(e)	(e)	(e)	(e)	(e)
(f)	(f)	(f)	(f)	(f)	(f)

8. By observing the detailness of the expansion, it is ______ to understand the original box.

a) very easy
b) fairly easy
c) more easy than hard
d) more hard than easy
e) fairly hard
f) very hard

exp-1	exp-2	exp-3	exp-4	exp-5	exp-6
(a)	(a)	(a)	(a)	(a)	(a)
(b)	(b)	(b)	(b)	(b)	(b)
(c)	(c)	(c)	(c)	(c)	(c)
(d)	(d)	(d)	(d)	(d)	(d)
(e)	(e)	(e)	(e)	(e)	(e)
(f)	(f)	(f)	(f)	(f)	(f)

EASE OF IMPLEMENTATION

By observing the overall structure of the expanded diagram, how easy would it be to implement it ?

9. Based on the structure of the expanded diagram, the implementation of this diagram should be

a) very easy
b) fairly easy
c) more easy than hard
d) more hard than easy
e) fairly hard
f) very hard

exp-1	exp-2	exp-3	exp-4	exp-5	exp-6
(a)	(a)	(a)	(a)	(a)	(a)
(b)	(b)	(b)	(b)	(b)	(b)
(c)	(c)	(c)	(c)	(c)	(c)
(d)	(d)	(d)	(d)	(d)	(d)
(e)	(e)	(e)	(e)	(e)	(e)
(f)	(f)	(f)	(f)	(f)	(f)

10. Based on the thoughts developed in answering above questions, you think that the expanded diagram is

a) very complex
b) fairly complex
c) more complex than simple
d) more simple than complex
d) fairly simple
e) very simple

exp-1	exp-2	exp-3	exp-4	exp-5	exp-6
(a)	(a)	(a)	(a)	(a)	(a)
(b)	(b)	(b)	(b)	(b)	(b)
(c)	(c)	(c)	(c)	(c)	(c)
(d)	(d)	(d)	(d)	(d)	(d)
(e)	(e)	(e)	(e)	(e)	(e)
(f)	(f)	(f)	(f)	(f)	(f)

	A	B	C	D	E	F	G	H	I
1									
2			Collected data for evaluating DFDs complexity						
3									
4	Complexity		expn-1	expn-2	expn-3	expn-4	expn-5	expn-6	Avg
5									
6	p1g		4	6	6	1	3	2	3.67
7	p2g		4	4	2	2	4	2	3.00
8	p3g		3	5	5	2	3	4	3.67
9	p4g		4	3	4	3	5	2	3.50
10	p5g		3	5	6	1	4	3	3.67
11	p7g		3	2	4	1	3	2	2.50
12	p8g		4	4	4	1	3	3	3.17
13	p9g		2	6	5	3	6	2	4.00
14	p10g		5	3	4	1	4	3	3.33
15	p11g		3	2	5	1	4	2	2.83
16	p12g		5	3	5	2	5	3	3.83
17	p13g		3	5	5	3	5	5	4.33
18	p14g		3	4	2	1	5	3	3.00
19	p15g		3	4	5	1	2	3	3.00
20	p16g		4	3	5	2	4	3	3.50
21	p17g		3	1	5	1	3	2	2.50
22	p18g		3	4	5	1	3	2	3.00
23	p19u		2	4	5	1	3	3	3.00
24	p20u		3	5	2	1	3	4	3.00
25	p21u		2	2	4	1	2	2	2.17
26	p22u		2	3	5	1	3	4	3.00
27	p23u		2	3	3	1	2	3	2.33
28	p24u		5	3	4	2	3	3	3.33
29	p25u		3	1	4	2	4	3	2.83
30	p26u		4	5	3	1	4	1	3.00
31	p27u		4	4	6	1	4	5	4.00
32	p28u		5	5	6	2	4	4	4.33
33	p29u		3	3	4	1	2	2	2.50
34	p30u		3	2	4	1	2	2	2.33
35	p31u		3	4	5	1	5	3	3.50
36	p32u		5	6	6	2	5	4	4.67
37	p33u		5	5	6	2	3	3	4.00
38	p34u		3	3	2	1	2	3	2.33
39	p35u		3	4	5	2	3	4	3.50
40	p36u		3	5	5	1	5	2	3.50
41	p37u		3	5	5	2	3	3	3.50
42	p38u		4	3	4	1	2	4	3.00
43	p39u		5	3	6	1	5	4	4.00
44	p40u		4	5	5	1	4	2	3.50
45	p41u		2	4	3	1	2	3	2.50
46	p42u		3	4	3	1	3	?????	2.80

	A	B	C	D	E	F	G	H	I
47	p43u		3	5	5	2	4	2	3.50
48	p44u		4	3	5	1	3	2	3.00
49	p45u		3	3	2	1	2	2	2.17
50	p46u		5	5	6	3	3	5	4.50
51	p47u		4	5	5	1	2	2	3.17
52	p48u		3	5	5	5	3	3	4.00
53	p49u		5	6	5	2	4	3	4.17
54	p50u		4	5	5	1	4	2	3.50
55	p51u		1	5	6	1	4	5	3.67
56	p52u		5	5	4	2	5	2	3.83
57	p53u		5	6	3	2	4	1	3.50
58									
59									
60									
61									
62	Freq. 1		1	2	0	32	0	2	
63	Freq. 2		6	4	5	15	10	19	
64	Freq. 3		22	13	5	4	17	18	
65	Freq. 4		12	11	11	0	15	8	
66	Freq. 5		11	16	23	1	9	4	
67	Freq. 6		0	6	8	0	1	0	
68									
69	total		52	52	52	52	52	51	
70									
71	AVG		3.50	4.00	4.48	1.52	3.50	2.86	
72	S.D.		1.02	1.28	1.18	0.80	1.06	1.00	
73									
74			Curve	Data :					
75									
76	Calculate fuzzy membership grades from the poll results								
77									
78			0.031	0.063	0	1	0	0.063	
79			0.316	0.211	0.263	0.789	0.526	1	
80			1	0.591	0.227	0.182	0.773	0.818	
81			0.8	0.733	0.733	0	1	0.533	
82			0.478	0.696	1	0.043	0.391	0.174	
83			0	0.75	1	0	0.125	0	
84									
85	Rearrange the columns in the order of complexity measures								
86									
87			0	0.063	0.031	0.063	0	1	
88			0.263	0.211	0.316	1	0.526	0.789	
89			0.227	0.591	1	0.818	0.773	0.182	
90			0.733	0.733	0.8	0.533	1	0	
91			1	0.696	0.478	0.174	0.391	0.043	
92			1	0.75	0	0	0.125	0	

	A	B	C	D	E	F	G	H	I
93									
94	Remove the outlier DFD5 and normalize when needed								
95									
96			0	0.063	0.031	0.063	1		
97			0.263	0.211	0.316	1	0.789		
98			0.227	0.591	1	0.818	0.182		
99			0.917	0.917	1	0.667	0	.....	
100			1	0.696	0.478	0.174	0.043		
101			1	0.75	0	0	0		
102									

	A	B	C	D	E	F	G	H	I
1									
2			Collected data for evaluating DFDs interconnection						
3									
4	Interconnction		expn-1	expn-2	expn-3	expn-4	expn-5	expn-6	Avg
5									
6	p1g		2	1	1	6	4	5	3.17
7	p2g		1	3	3	5	2	4	3.00
8	p3g		2	3	1	4	3	3	2.67
9	p4g		2	3	1	6	2	5	3.17
10	p5g		2	1	1	4	2	4	2.33
11	p7g		2	5	6	3	2	5	3.83
12	p8g		2	2	3	5	2	3	2.83
13	p9g		3	2	2	4	2	5	3.00
14	p10g		2	3	4	5	2	5	3.50
15	p11g		2	3	2	6	3	3	3.17
16	p12g		2	3	1	5	3	4	3.00
17	p13g		2	1	1	4	2	2	2.00
18	p14g		2	4	2	5	4	4	3.50
19	p15g		3	2	4	5	6	3	3.83
20	p16g		2	3	5	5	2	3	3.33
21	p17g		1	3	3	6	4	4	3.50
22	p18g		3	1	1	6	3	3	2.83
23	p19u		2	2	1	6	2	3	2.67
24	p20u		2	3	4	6	4	5	4.00
25	p21u		1	3	1	6	3	4	3.00
26	p22u		1	1	1	6	4	5	3.00
27	p23u		1	2	1	6	3	4	2.83
28	p24u		2	3	3	5	4	5	3.67
29	p25u		2	2	1	5	3	4	2.83
30	p26u		2	3	1	3	2	4	2.50
31	p27u		2	3	1	6	3	3	3.00
32	p28u		1	3	1	4	2	2	2.17
33	p29u		2	2	1	6	4	5	3.33
34	p30u		2	3	1	5	3	4	3.00
35	p31u		2	3	2	6	3	4	3.33
36	p32u		2	3	1	4	2	3	2.50
37	p33u		2	2	1	5	2	3	2.50
38	p34u		2	2	3	4	3	4	3.00
39	p35u		2	2	2	3	2	3	2.33
40	p36u		2	2	1	5	2	5	2.83
41	p37u		2	2	1	5	4	4	3.00
42	p38u		1	2	3	6	3	3	3.00
43	p39u		1	3	1	5	2	3	2.50
44	p40u		4	2	2	6	3	4	3.50
45	p41u		3	3	2	6	3	3	3.33
46	p42u		4	3	2	5	2	3	3.17

	A	B	C	D	E	F	G	H	I
47	p43u		2	1	2	6	3	6	3.33
48	p44u		2	3	1	4	2	2	2.33
49	p45u		2	3	4	6	3	4	3.67
50	p46u		2	2	1	5	2	4	2.67
51	p47u		2	3	1	6	5	6	3.83
52	p48u		2	3	2	3	5	4	3.17
53	p49u		1	2	1	5	3	3	2.50
54	p50u		2	3	1	4	2	3	2.50
55	p51u		2	2	1	6	2	3	2.67
56	p52u		2	1	2	6	2	3	2.67
57	p53u		1	2	1	3	3	4	2.33
58									
59									
60									
61									
62	Freq. 1		10	7	29	0	0	0	
63	Freq. 2		36	18	11	0	23	3	
64	Freq. 3		4	25	6	5	18	19	
65	Freq. 4		2	1	4	9	8	18	
66	Freq. 5		0	1	1	17	2	10	
67	Freq. 6		0	0	1	21	1	2	
68									
69	total		52	52	52	52	52	52	
70									
71	AVG		1.96	2.44	1.85	5.04	2.85	3.79	
72	S.D.		0.66	0.83	1.21	0.99	0.96	0.96	
73									
74									
75									
76	Calculate fuzzy membership grades from the poll results								
77									
78			0.345	0.241	1	0	0	0	
79			1	0.5	0.306	0	0.639	0.083	
80			0.16	1	0.24	0.2	0.72	0.76	
81			0.111	0.056	0.222	0.5	0.444	1	
82			0	0.059	0.059	1	0.118	0.588	
83			0	0	0.048	1	0.048	0.095	
84									
85	Rearrange the columns in the order of interconnection measure								
86									
87			1	0.345	0.241	0	0	0	
88			0.306	1	0.5	0.083	0.639	0	
89			0.24	0.16	1	0.76	0.72	0.2	
90			0.222	0.111	0.056	1	0.444	0.5	
91			0.059	0	0.059	0.588	0.118	1	
92			0.048	0	0	0.095	0.048	1	

	J	K	L	M	N	O	P
47	p43u	-0.50	1.50	1.50	-1.50	0.50	-1.50
48	p44u	1.00	0.00	2.00	-2.00	0.00	-1.00
49	p45u	0.83	0.83	-0.17	-1.17	-0.17	-0.17
50	p46u	0.50	0.50	1.50	-1.50	-1.50	0.50
51	p47u	0.83	1.83	1.83	-2.17	-1.17	-1.17
52	p48u	-1.00	1.00	1.00	1.00	-1.00	-1.00
53	p49u	0.83	1.83	0.83	-2.17	-0.17	-1.17
54	p50u	0.50	1.50	1.50	-2.50	0.50	-1.50
55	p51u	-2.67	1.33	2.33	-2.67	0.33	1.33
56	p52u	1.17	1.17	0.17	-1.83	1.17	-1.83
57	p53u	1.50	2.50	-0.50	-1.50	0.50	-2.50
58							
59	Range	Max	2.5	Min:	- 3	Total:	5.5
60	Subrange						0.917
61							
62	-3.0 -- -2.084	1	0	0	15	0	1
63	-2.084 -- -1.168	2	1	0	32	2	11
64	-1.168 -- -0.252	12	10	5	4	15	20
65	-0.252 -- 0.664	19	9	6	0	21	11
66	0.664 -- 1.58	16	25	24	1	12	8
67	1.58 -- 2.5	2	7	17	0	2	0
68							
69	total	52	52	52	52	52	51
70							
71	Avg	0.19	0.69	1.17	-1.79	0.19	-0.46
72	S.D.	0.83	0.98	0.90	0.68	0.82	0.91
73	Max	1.67	2.50	2.50	1.00	2.00	1.33
74	Min	-2.67	-1.83	-1.00	-3.00	-1.50	-2.50
75							
76	Calculate fuzzy membership grades from the poll results						
77							
78		0.067	0	0	1	0	0.067
79		0.063	0.031	0	1	0.063	0.344
80		0.6	0.5	0.25	0.2	0.75	1
81		0.905	0.429	0.286	0	1	0.524
82		0.64	1	0.96	0.04	0.48	0.32
83		0.118	0.412	1	0	0.118	0
84							
85	Rearrange the columns in the order of complexity measures						
86							
87		0	0	0.067	0.067	0	1
88		0	0.031	0.063	0.344	0.063	1
89		0.25	0.5	0.6	1	0.75	0.2
90		0.286	0.429	0.905	0.524	1	0
91		0.96	1	0.64	0.32	0.48	0.04
92		1	0.412	0.118	0	0.118	0

	J	K	L	M	N	O	P
93							
94	Remove the outlier DFD5 and normalize when needed						
95							
96		0	0	0.067	0.067	1	
97		0	0.031	0.063	0.344	1	
98		0.25	0.5	0.6	1	0.2	
99		0.316	0.474	1	0.579	0	.....
100		0.96	1	0.64	0.32	0.04	
101		1	0.412	0.118	0	0	

	J	K	L	M	N	O	P
1							
2	Normalized data for evaluating DFDs' interconnection						
3							
4	Interconnection	expn-1	expn-2	expn-3	expn-4	expn-5	expn-6
5							
6	p1g	-1.17	-2.17	-2.17	2.83	0.83	1.83
7	p2g	-2.00	0.00	0.00	2.00	-1.00	1.00
8	p3g	-0.67	0.33	-1.67	1.33	0.33	0.33
9	p4g	-1.17	-0.17	-2.17	2.83	-1.17	1.83
10	p5g	-0.33	-1.33	-1.33	1.67	-0.33	1.67
11	p7g	-1.83	1.17	2.17	-0.83	-1.83	1.17
12	p8g	-0.83	-0.83	0.17	2.17	-0.83	0.17
13	p9g	0.00	-1.00	-1.00	1.00	-1.00	2.00
14	p10g	-1.50	-0.50	0.50	1.50	-1.50	1.50
15	p11g	-1.17	-0.17	-1.17	2.83	-0.17	-0.17
16	p12g	-1.00	0.00	-2.00	2.00	0.00	1.00
17	p13g	0.00	-1.00	-1.00	2.00	0.00	0.00
18	p14g	-1.50	0.50	-1.50	1.50	0.50	0.50
19	p15g	-0.83	-1.83	0.17	1.17	2.17	-0.83
20	p16g	-1.33	-0.33	1.67	1.67	-1.33	-0.33
21	p17g	-2.50	-0.50	-0.50	2.50	0.50	0.50
22	p18g	0.17	-1.83	-1.83	3.17	0.17	0.17
23	p19u	-0.67	-0.67	-1.67	3.33	-0.67	0.33
24	p20u	-2.00	-1.00	0.00	2.00	0.00	1.00
25	p21u	-2.00	0.00	-2.00	3.00	0.00	1.00
26	p22u	-2.00	-2.00	-2.00	3.00	1.00	2.00
27	p23u	-1.83	-0.83	-1.83	3.17	0.17	1.17
28	p24u	-1.67	-0.67	-0.67	1.33	0.33	1.33
29	p25u	-0.83	-0.83	-1.83	2.17	0.17	1.17
30	p26u	-0.50	0.50	-1.50	0.50	-0.50	1.50
31	p27u	-1.00	0.00	-2.00	3.00	0.00	0.00
32	p28u	-1.17	0.83	-1.17	1.83	-0.17	-0.17
33	p29u	-1.33	-1.33	-2.33	2.67	0.67	1.67
34	p30u	-1.00	0.00	-2.00	2.00	0.00	1.00
35	p31u	-1.33	-0.33	-1.33	2.67	-0.33	0.67
36	p32u	-0.50	0.50	-1.50	1.50	-0.50	0.50
37	p33u	-0.50	-0.50	-1.50	2.50	-0.50	0.50
38	p34u	-1.00	-1.00	0.00	1.00	0.00	1.00
39	p35u	-0.33	-0.33	-0.33	0.67	-0.33	0.67
40	p36u	-0.83	-0.83	-1.83	2.17	-0.83	2.17
41	p37u	-1.00	-1.00	-2.00	2.00	1.00	1.00
42	p38u	-2.00	-1.00	0.00	3.00	0.00	0.00
43	p39u	-1.50	0.50	-1.50	2.50	-0.50	0.50
44	p40u	0.50	-1.50	-1.50	2.50	-0.50	0.50
45	p41u	-0.33	-0.33	-1.33	2.67	-0.33	-0.33
46	p42u	0.83	-0.17	-1.17	1.83	-1.17	-0.17

	J	K	L	M	N	O	P
47	p43u	-1.33	-2.33	-1.33	2.67	-0.33	2.67
48	p44u	-0.33	0.67	-1.33	1.67	-0.33	-0.33
49	p45u	-1.67	-0.67	0.33	2.33	-0.67	0.33
50	p46u	-0.67	-0.67	-1.67	2.33	-0.67	1.33
51	p47u	-1.83	-0.83	-2.83	2.17	1.17	2.17
52	p48u	-1.17	-0.17	-1.17	-0.17	1.83	0.83
53	p49u	-1.50	-0.50	-1.50	2.50	0.50	0.50
54	p50u	-0.50	0.50	-1.50	1.50	-0.50	0.50
55	p51u	-0.67	-0.67	-1.67	3.33	-0.67	0.33
56	p52u	-0.67	-1.67	-0.67	3.33	-0.67	0.33
57	p53u	-1.33	-0.33	-1.33	0.67	0.67	1.67
58							
59	Range	Max:	3.33	Min:	-2.83	Total:	6.167
60	Subrange						1.028
61							
62	-2.83 -- -1.802	9	5	14	0	1	0
63	-1.802 -- -0.774	25	15	24	1	8	1
64	-0.774 -- 0.254	16	23	11	1	30	11
65	0.254 -- 1.282	2	9	1	6	11	26
66	1.282 -- 2.31	0	0	2	21	2	13
67	2.31 -- 3.33	0	0	0	23	0	1
68							
69	total	52	52	52	52	52	52
70							
71	Avg	-1.03	-0.54	-1.14	2.05	-0.14	0.80
72	S.D.	0.69	0.78	0.98	0.90	0.78	0.77
73	Max	0.83	1.17	2.17	3.33	2.17	2.67
74	Min	-2.50	-2.33	-2.83	-0.83	-1.83	-0.83
75							
76	Calculate fuzzy membership grades from the poll results						
77							
78		0.643	0.357	1	0	0.071	0
79		1	0.6	0.96	0.04	0.32	0.04
80		0.533	0.767	0.367	0.033	1	0.367
81		0.077	0.346	0.038	0.231	0.423	1
82		0	0	0.095	1	0.095	0.619
83		0	0	0	1	0	0.043
84							
85	Rearrange the columns in the order of interconnection measure						
86							
87		1	0.643	0.357	0	0.071	0
88		0.96	1	0.6	0.04	0.32	0.04
89		0.367	0.533	0.767	0.367	1	0.033
90		0.038	0.077	0.346	1	0.423	0.231
91		0.095	0	0	0.619	0.095	1
92		0	0	0	0.043	0	1

	J	K	L	M	N	O	P
9 3							
9 4	Remove the outlier DFD5 and normalize when needed						
9 5							
9 6		1	0.643	0.357	0	0	
9 7		0.96	1	0.6	0.04	0.04	
9 8		0.478	0.695	1	0.478	0.043	• • • • •
9 9		0.038	0.077	0.346	1	0.231	
1 0 0		0.095	0	0	0.619	1	
1 0 1		0	0	0	0.043	1	

	J	K	L	M	N	O	P
1							
2		Normalized data for evaluating DFDs modularity					
3							
4	Modularity	expn-1	expn-2	expn-3	expn-4	expn-5	expn-6
5							
6	p1g	-2.67	-1.67	1.33	1.33	0.33	1.33
7	p2g	0.00	-1.00	1.00	0.00	1.00	-1.00
8	p3g	0.50	-1.50	-0.50	1.50	-0.50	0.50
9	p4g	-1.33	1.67	1.67	0.67	-1.33	-1.33
10	p5g	-0.67	-0.67	-0.67	1.33	-0.67	1.33
11	p7g	-1.33	0.67	1.67	0.67	-2.33	0.67
12	p8g	-1.00	0.00	0.00	1.00	-1.00	1.00
13	p9g	-0.33	-0.33	-0.33	-0.33	-0.33	1.67
14	p10g	-1.33	-2.33	1.67	1.67	-1.33	1.67
15	p11g	0.00	-1.00	1.00	0.00	0.00	0.00
16	p12g	-0.33	-1.33	0.67	0.67	-0.33	0.67
17	p13g	0.50	-0.50	-0.50	1.50	-0.50	-0.50
18	p14g	-1.83	0.17	0.17	1.17	0.17	0.17
19	p15g	-0.50	-2.50	0.50	1.50	1.50	-0.50
20	p16g	0.00	-2.00	1.00	1.00	0.00	0.00
21	p17g	-1.33	-1.33	-0.33	2.67	-0.33	0.67
22	p18g	-0.50	0.50	1.50	-0.50	-0.50	-0.50
23	p19u	-0.33	-0.33	-1.33	2.67	-0.33	-0.33
24	p20u	-0.50	-1.50	1.50	-0.50	0.50	0.50
25	p21u	-2.83	0.17	1.17	2.17	1.17	-1.83
26	p22u	0.00	0.00	-1.00	1.00	0.00	0.00
27	p23u	-1.67	-0.67	0.33	1.33	0.33	0.33
28	p24u	-0.50	-1.50	0.50	1.50	0.50	-0.50
29	p25u	-1.00	2.00	-1.00	0.00	-1.00	1.00
30	p26u	-0.50	-1.50	1.50	0.50	-1.50	1.50
31	p27u	-0.67	-0.67	-0.67	1.33	0.33	0.33
32	p28u	0.50	-0.50	-0.50	0.50	-0.50	0.50
33	p29u	-1.50	0.50	-0.50	1.50	-0.50	0.50
34	p30u	-1.50	-1.50	0.50	1.50	-0.50	1.50
35	p31u	-0.83	0.17	0.17	1.17	-0.83	0.17
36	p32u	-0.17	-1.17	-0.17	1.83	-1.17	0.83
37	p33u	-1.33	-0.33	0.67	-0.33	-0.33	1.67
38	p34u	0.33	-0.67	0.33	0.33	-0.67	0.33
39	p35u	-0.33	-0.33	-0.33	-0.33	0.67	0.67
40	p36u	-0.50	0.50	1.50	1.50	-1.50	-1.50
41	p37u	-0.50	1.50	1.50	-1.50	-0.50	-0.50
42	p38u	-0.67	-0.67	0.33	0.33	0.33	0.33
43	p39u	0.33	0.33	-0.67	0.33	-0.67	0.33
44	p40u	-0.50	-0.50	0.50	1.50	-1.50	0.50
45	p41u	-1.17	-1.17	1.83	0.83	-0.17	-0.17
46	p42u	-0.50	0.50	-0.50	0.50	-0.50	0.50

	J	K	L	M	N	O	P
47	p43u	-1.33	-2.33	0.67	1.67	-1.33	2.67
48	p44u	1.00	0.00	2.00	-2.00	0.00	-1.00
49	p45u	-1.17	-1.17	-0.17	0.83	0.83	0.83
50	p46u	-0.33	-0.33	-0.33	1.67	-0.33	-0.33
51	p47u	-1.00	-1.00	0.00	1.00	0.00	1.00
52	p48u	-0.33	-0.33	-0.33	0.67	0.67	-0.33
53	p49u	-0.17	-1.17	-1.17	1.83	-0.17	0.83
54	p50u	-0.83	-0.83	-0.83	1.17	0.17	1.17
55	p51u	-0.33	-0.33	1.67	-0.33	-0.33	-0.33
56	p52u	-0.50	-1.50	2.50	0.50	-1.50	0.50
57	p53u	-2.00	-2.00	1.00	2.00	0.00	1.00
58							
59	Range	Max:	2.667	Min:	-2.83	Total:	5.5
60	Subrange						0.917
61							
62	-2.83 -- -1.914	3	5	0	1	1	0
63	-1.914 -- -0.998	15	16	4	.1	10	5
64	-0.998 -- 0.082	24	16	17	6	22	10
65	0.082 -- 0.834	9	12	13	16	16	24
66	0.834 -- 1.75	1	2	15	22	3	12
67	1.75 -- 2.67	0	1	3	6	0	1
68							
69	total	52	52	52	52	52	52
70							
71	Avg	-0.68	-0.61	0.39	0.86	-0.32	0.36
72	S.D.	0.77	0.98	0.96	0.94	0.77	0.88
73	Max	1.00	2.00	2.50	2.67	1.50	2.67
74	Min	-2.83	-2.50	-1.33	-2.00	-2.33	-1.83

	J	K	L	M	N	O	P
1							
2	Normalized data for evaluating DFDs' logical cohesion						
3							
4	Logical cohesion	expn-1	expn-2	expn-3	expn-4	expn-5	expn-6
5							
6	p1g	-0.33	-1.33	0.67	1.67	-0.33	-0.33
7	p2g	-1.00	0.00	1.00	0.00	0.00	0.00
8	p3g	-0.33	-1.33	-0.33	0.67	1.67	-0.33
9	p4g	-0.67	0.33	-1.67	1.33	-0.67	1.33
10	p5g	-1.00	-1.00	-1.00	2.00	0.00	1.00
11	p7g	-0.50	1.50	1.50	-0.50	-1.50	-0.50
12	p8g	0.50	-0.50	-0.50	0.50	0.50	-0.50
13	p9g	-0.17	-1.17	0.83	-0.17	-1.17	1.83
14	p10g	-0.33	-0.33	1.67	-0.33	-1.33	0.67
15	p11g	0.67	-1.33	0.67	0.67	-1.33	0.67
16	p12g	-0.17	-1.17	2.83	-1.17	-0.17	-0.17
17	p13g	0.67	-0.33	-0.33	0.67	-0.33	-0.33
18	p14g	-0.67	0.33	-0.67	1.33	-0.67	0.33
19	p15g	-2.00	-2.00	0.00	2.00	2.00	0.00
20	p16g	-1.17	-1.17	1.83	0.83	-0.17	-0.17
21	p17g	-0.33	-2.33	0.67	1.67	-1.33	1.67
22	p18g	1.17	-1.83	1.17	0.17	-1.83	1.17
23	p19u	-0.17	-0.17	-1.17	1.83	-0.17	-0.17
24	p20u	1.67	-1.33	0.67	-0.33	-1.33	0.67
25	p21u	-0.17	-0.17	-1.17	-0.17	0.83	0.83
26	p22u	0.17	-0.83	-0.83	0.17	0.17	1.17
27	p23u	-0.50	-0.50	0.50	0.50	-0.50	0.50
28	p24u	-1.17	0.83	-0.17	-0.17	0.83	-0.17
29	p25u	0.33	0.33	-0.67	-0.67	-0.67	1.33
30	p26u	-0.67	-0.67	1.33	0.33	-0.67	0.33
31	p27u	0.00	0.00	0.00	0.00	-1.00	1.00
32	p28u	0.33	-0.67	-0.67	0.33	0.33	0.33
33	p29u	-0.67	1.33	-0.67	0.33	-0.67	0.33
34	p30u	-0.33	-0.33	-0.33	0.67	-0.33	0.67
35	p31u	-0.33	-0.33	-0.33	1.67	-1.33	0.67
36	p32u	0.17	-0.83	-0.83	1.17	0.17	0.17
37	p33u	-0.67	-0.67	0.33	1.33	-0.67	0.33
38	p34u	0.33	0.33	0.33	-0.33	-0.67	-0.67
39	p35u	-0.17	-0.17	-0.17	-0.17	-0.17	0.83
40	p36u	-1.00	-1.00	1.00	1.00	-1.00	1.00
41	p37u	-0.67	-0.67	1.33	-0.67	0.33	0.33
42	p38u	-0.83	-0.83	0.17	1.17	0.17	0.17
43	p39u	-0.67	0.33	-0.67	0.33	0.33	0.33
44	p40u	0.17	-0.83	0.17	0.17	0.17	0.17
45	p41u	-0.83	-0.83	0.17	1.17	0.17	0.17
46	p42u	1.33	0.33	-0.67	-0.67	-0.67	0.33

	J	K	L	M	N	O	P
1							
2	Normalized data for evaluating DFDs' ease of implementation						
3							
4	Ease of imp.	expn-1	expn-2	expn-3	expn-4	expn-5	expn-6
5							
6	p1g	-1.00	-1.00	-1.00	2.00	0.00	1.00
7	p2g	-1.67	0.33	0.33	1.33	0.33	-0.67
8	p3g	0.00	1.00	0.00	1.00	-1.00	-1.00
9	p4g	-0.33	-0.33	-0.33	0.67	-1.33	1.67
10	p5g	0.00	-1.00	-2.00	3.00	-1.00	1.00
11	p7g	0.17	0.17	1.17	0.17	-1.83	0.17
12	p8g	-1.67	-1.67	-0.67	2.33	0.33	1.33
13	p9g	-1.50	-1.50	0.50	1.50	0.50	0.50
14	p10g	-0.17	-1.17	0.83	0.83	-1.17	0.83
15	p11g	-0.33	-1.33	0.67	1.67	-1.33	0.67
16	p12g	-1.17	0.83	0.83	0.83	-2.17	0.83
17	p13g	0.00	-1.00	-1.00	2.00	0.00	0.00
18	p14g	0.50	0.50	0.50	-1.50	-1.50	1.50
19	p15g	-0.83	-0.83	0.17	1.17	1.17	-0.83
20	p16g	-1.17	-2.17	2.83	1.83	-1.17	-0.17
21	p17g	-0.67	-2.67	0.33	2.33	-0.67	1.33
22	p18g	-0.33	-1.33	1.67	0.67	-1.33	0.67
23	p19u	0.17	-0.83	-0.83	2.17	-0.83	0.17
24	p20u	-0.50	-1.50	1.50	1.50	-0.50	-0.50
25	p21u	-1.00	0.00	-1.00	2.00	0.00	0.00
26	p22u	0.83	-1.17	-2.17	1.83	0.83	-0.17
27	p23u	-0.50	-0.50	-0.50	1.50	-0.50	0.50
28	p24u	-1.67	1.33	-0.67	0.33	0.33	0.33
29	p25u	-0.33	1.67	-1.33	0.67	-1.33	0.67
30	p26u	0.67	-1.33	0.67	-0.33	-1.33	1.67
31	p27u	-0.33	-0.33	-1.33	2.67	-0.33	-0.33
32	p28u	0.50	-0.50	-0.50	1.50	-0.50	-0.50
33	p29u	0.33	-0.67	-0.67	1.33	-0.67	0.33
34	p30u	0.00	-1.00	-1.00	2.00	0.00	0.00
35	p31u	-0.50	-0.50	-0.50	1.50	-0.50	0.50
36	p32u	-0.17	-1.17	-1.17	1.83	-0.17	0.83
37	p33u	0.17	-0.83	-1.83	2.17	-0.83	1.17
38	p34u	0.00	-1.00	1.00	1.00	?????	-1.00
39	p35u	-0.83	-0.83	-0.83	1.17	1.17	0.17
40	p36u	0.83	-1.17	0.83	1.83	-2.17	-0.17
41	p37u	-1.33	-1.33	-1.33	1.67	1.67	0.67
42	p38u	-0.67	-0.67	-0.67	1.33	0.33	0.33
43	p39u	-0.17	0.83	-0.17	-0.17	-0.17	-0.17
44	p40u	0.67	-1.33	-1.33	1.67	-0.33	0.67
45	p41u	-0.50	-1.50	-0.50	1.50	0.50	0.50
46	p42u	1.67	-0.33	-0.33	0.67	-1.33	-0.33
47	p43u	-0.50	-1.50	-0.50	1.50	-0.50	1.50
48	p44u	1.33	-0.67	-0.67	0.33	-0.67	0.33
49	p45u	-0.33	-1.33	0.67	1.67	-0.33	-0.33
50	p46u	-0.50	-0.50	-0.50	2.50	-0.50	-0.50

	J	K	L	M	N	O	P
51	p47u	-0.33	-1.33	-0.33	2.67	-0.33	-0.33
52	p48u	-0.33	-1.33	-0.33	0.67	0.67	0.67
53	p49u	-0.83	-0.83	-0.83	2.17	-0.83	1.17
54	p50u	-1.17	-1.17	-1.17	2.83	-1.17	1.83
55	p51u	-1.17	-0.17	-0.17	0.83	0.83	-0.17
56	p52u	-0.83	-1.83	2.17	1.17	-1.83	1.17
57	p53u	-2.67	-2.67	2.33	1.33	-0.67	2.33
58							
59	Range	Max:	3	Min:	-2.67	Total:	5.667
60	Subrange						0.944
61							
62	-2.67 -- -1.725	1	4	3	0	4	0
63	-1.725 -- -0.78	15	28	13	1	15	3
64	-0.78 -- 0.165	27	13	19	3	20	19
65	0.165 -- 1.11	7	5	11	12	9	19
66	1.11 -- 2.055	2	2	3	26	3	10
67	2.055 -- 3.00	0	0	3	10	0	1
68							
69	total	52	52	52	52	51	52
70							
71	Avg	-0.39	-0.79	-0.18	1.40	-0.47	0.42
72	S.D.	0.80	0.90	1.09	0.85	0.87	0.77
73	Max	1.67	1.67	2.83	3.00	1.67	2.33
74	Min	-2.67	-2.67	-2.17	-1.50	-2.17	-1.00
75							
76	Calculate the fuzzy membership grades from the poll results						
77							
78		0.25	1	0.75	0	1	0
79		0.536	1	0.464	0.036	0.536	0.107
80		1	0.481	0.704	0.111	0.741	0.704
81		0.368	0.263	0.579	0.632	0.474	1
82		0.077	0.077	0.115	1	0.115	0.385
83		0	0	0.3	1	0	0.1
84							
85	Rearrange the columns in the order of complexity measures						
86							
87		0.75	1	0.25	0	1	0
88		0.464	1	0.536	0.107	0.536	0.036
89		0.704	0.481	1	0.704	0.741	0.111
90		0.579	0.263	0.368	1	0.474	0.632
91		0.115	0.077	0.077	0.385	0.115	1
92		0.3	0	0	0.1	0	1
93							
94	Remove the outlier DFD5 and normalize when needed						
95							
96		0.75	1	0.25	0	0	
97		0.464	1	0.536	0.107	0.036	
98		0.704	0.481	1	0.704	0.111	
99		0.579	0.263	0.368	1	0.632	
100		0.115	0.077	0.077	0.385	1	
101		0.3	0	0	0.1	1	

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
1						Data file used by SAS									
2															
3															
4		V1	V2	V3	V4	V5	V6	V7	V8	V9	V10	V11	V12	V13	V14
5		comp1	inter	modul	lcohe	easeofi	comp2	ncomp1	ninter	nmodul	nlcohe	neaseofi	ncomp2	COM	INTER
6															
7	DFD-1	3.5	1.96	2.12	2.31	2.62	3.73	0.19	-1.03	-0.68	-0.16	-0.39	0.33	194.84	9.742
8	DFD-2	4	2.44	2.19	1.9	2.21	4.19	0.69	-0.54	-0.61	-0.57	-0.79	0.79	148.755	9.917
9	DFD-3	4.48	1.85	3.19	2.63	2.83	3.58	1.17	-1.14	0.39	0.16	-0.18	0.18	42	6
10	DFD-4	1.52	5.04	3.65	3.06	4.4	2.04	-1.79	2.05	0.86	0.59	1.4	-1.36	1947.804	25.63
11	DFD-5	3.5	2.85	2.48	2.12	2.53	3.73	0.19	-0.14	-0.32	-0.36	-0.47	0.33	679.212	18.87
12	DFD-6	2.86	3.79	3.15	2.81	3.42	3.12	-0.46	0.8	0.36	0.34	0.42	-0.28	443.95	17.08

	P	Q	R	S	T	U	V	W	X	Y	Z
1		Data file used by SAS									
2											
3											
4	V15	V16	V17	V18	V19	V20	V21	V22	V23	V24	V25
5	CSIZE	MOD	ATB	ACB	APB	MSDR	DSR	DS	ALEN	LF	LD
6											
7	20	2.429	2.71	0.429	1.857	1.857	1.889	3.746	1	1	1
8	15	2.5	3.25	0.75	1.5	1.75	1.667	3.417	1	1	1
9	7	1.33	2	0.667	0.333	1	1	2	1	1	1
10	76	3.667	4.667	1.333	10	1.667	1.294	2.961	4	1.667	6.668
11	36	3	4	1	4.4	1.8	1.667	3.467	3	2	6
12	26	3	4	1.25	3	1.5	1.2	2.7	3.5	1.75	6.125

çorrelation Among Variables

File: try.2 size: 6 * 25 MISS= -9999.
Include all cases

VARIABLE	MEAN	STD. DEV.
V7	-.002	1.033
V8	0.000	1.226
V9	.000	.626
V10	0.000	.440
V11	-.002	.796
V12	-.002	.749
V13	576.093	710.340
V14	14.538	7.286
V15	30.000	24.585
V16	2.654	.787
V17	3.438	.977
V18	.905	.352
V19	3.515	3.465
V20	1.596	.317
V21	1.453	.339
V22	3.049	.637
V23	2.250	1.405
V24	1.403	.455
V25	3.632	2.892

CORRELATION MATRIX

	V7	V8	V9	V10	V11	V12	V13	V14	V15	V16	V17	V18	V19	V20
V7	1.00	-.94	-.58	-.67	-.88	.89	-.92	-.91	-.93	-.89	-.85	-.76	-.93	-.36
V8	-.94	1.00	.70	.68	.88	-.88	.90	.93	.90	.86	.90	.92	.90	.19
V9	-.58	.70	1.00	.92	.88	-.86	.63	.52	.56	.25	.37	.72	.57	-.55
V10	-.67	.68	.92	1.00	.93	-.90	.61	.49	.56	.29	.33	.60	.56	-.41
V11	-.88	.88	.88	.93	1.00	-.99	.85	.75	.82	.59	.61	.75	.82	-.12
V12	.89	-.88	-.86	-.90	-.99	1.00	-.89	-.77	-.86	-.61	-.63	-.74	-.86	.08
V13	-.92	.90	.63	.61	.85	-.89	1.00	.92	.99	.80	.81	.74	1.00	.25
V14	-.91	.93	.52	.49	.75	-.77	.92	1.00	.94	.93	.96	.86	.94	.39
V15	-.93	.90	.56	.56	.82	-.86	.99	.94	1.00	.85	.84	.73	1.00	.35
V16	-.89	.86	.25	.29	.59	-.61	.80	.93	.85	1.00	.97	.75	.85	.65
V17	-.85	.90	.37	.33	.61	-.63	.81	.96	.84	.97	1.00	.87	.85	.51
V18	-.76	.92	.72	.60	.75	-.74	.74	.86	.73	.75	.87	1.00	.75	.02
V19	-.93	.90	.57	.56	.82	-.86	1.00	.94	1.00	.85	.85	.75	1.00	.34
V20	-.36	.19	-.55	-.41	-.12	.08	.25	.39	.35	.65	.51	.02	.34	1.00
V21	.08	-.29	-.85	-.70	-.51	.46	-.15	-.06	-.05	.22	.05	-.45	-.07	.88
V22	-.13	-.06	-.72	-.58	-.33	.29	.05	.17	.15	.44	.28	-.23	.13	.97
V23	-.84	.91	.65	.61	.77	-.78	.80	.94	.81	.83	.91	.95	.82	.17
V24	-.57	.66	.37	.29	.43	-.46	.57	.82	.60	.72	.81	.80	.61	.25
V25	-.77	.85	.57	.52	.67	-.69	.74	.93	.76	.81	.90	.92	.77	.20

	V21	V22	V23	V24	V25
V7	.08	-.13	-.84	-.57	-.77
V8	-.29	-.06	.91	.66	.85
V9	-.85	-.72	.65	.37	.57

V10	-.70	-.58	.61	.29	.52
V11	-.51	-.33	.77	.43	.67
V12	.46	.29	-.78	-.46	-.69
V13	-.15	.05	.80	.57	.74
V14	-.06	.17	.94	.82	.93
V15	-.05	.15	.81	.60	.76
V16	.22	.44	.83	.72	.81
V17	.05	.28	.91	.81	.90
V18	-.45	-.23	.95	.80	.92
V19	-.07	.13	.82	.61	.77
V20	.88	.97	.17	.25	.20
V21	1.00	.97	-.29	-.11	-.23
V22	.97	1.00	-.07	.07	-.02
V23	-.29	-.07	1.00	.89	.99
V24	-.11	.07	.89	1.00	.95
V25	-.23	-.02	.99	.95	1.00

Number of cases: 6
Number of missing cases: 0

MULTIPLE REGRESSION RESULTS

DEPENDENT VARIABLE: V11

INDEPENDENT VARIABLES: V12

MULTIPLE CORRELATION: .9944 F(1, 4) = 355.354 p = 0.000
R-square: .9889

BETA for V12 = -.9944189 B = -1.0569654 t(4) = -18.851 p =0.000

INTERCEPT = -.0034283

Analysis of variance

	SS	MS	df	F	p
REGRESSION	3.131	3.13	1	355.35	0.000
RESIDUAL	.035	.01	4		
TOTAL	3.166				

MULTIPLE REGRESSION RESULTS

DEPENDENT VARIABLE: V11

INDEPENDENT VARIABLES: V9

MULTIPLE CORRELATION: .8766 F(1, 4) = 13.267 p = .023
R-square: .7683

BETA for V9 = .8765532 B = 1.1145439 t(4) = 3.642 p = .023

INTERCEPT = -.0016667

Analysis of variance

	SS	MS	df	F	p
REGRESSION	2.432	2.43	1	13.27	.023
RESIDUAL	.733	.18	4		
TOTAL	3.166				

MULTIPLE REGRESSION RESULTS

DEPENDENT VARIABLE: V11

INDEPENDENT VARIABLES: V13

MULTIPLE CORRELATION: .8488 F(1, 4) = 10.309 p = .033
R-square: .7205

BETA for V13 = .8487989 B = .0009508 t(4) = 3.211 p = .033

INTERCEPT = -.5494314

Analysis of variance

	SS	MS	df	F	p
REGRESSION	2.281	2.28	1	10.31	.033
RESIDUAL	.885	.22	4		
TOTAL	3.166				

MULTIPLE REGRESSION RESULTS

DEPENDENT VARIABLE: V9

INDEPENDENT VARIABLES: V16

MULTIPLE CORRELATION: .2501 F(1, 4) = .267 p = .634
R-square: .0626

BETA for V16 = .2501416 B = .1990354 t(4) = .517 p = .634

INTERCEPT = -.5283063

Analysis of variance

	SS	MS	df	F	p
REGRESSION	.123	.12	1	.27	.634
RESIDUAL	1.836	.46	4		
TOTAL	1.958				

MULTIPLE REGRESSION RESULTS

DEPENDENT VARIABLE: V8

INDEPENDENT VARIABLES: V14

MULTIPLE CORRELATION: .9254 F(1, 4) = 23.846 p = .009
R-square: .8564

BETA for V14 = .9253942 B = .1556988 t(4) = 4.883 p = .009

INTERCEPT = -2.2636012

Analysis of variance

	SS	MS	df	F	p
REGRESSION	6.435	6.43	1	23.85	.009
RESIDUAL	1.079	.27	4		
TOTAL	7.514				

MULTIPLE REGRESSION RESULTS

DEPENDENT VARIABLE: V8

INDEPENDENT VARIABLES: V17

MULTIPLE CORRELATION: .8983 F(1, 4) = 16.721 p = .016
R-square: .8070

BETA for V17 = .8983080 B = 1.1268505 t(4) = 4.089 p = .016

INTERCEPT = -3.8739241

Analysis of variance

	SS	MS	df	F	p
REGRESSION	6.064	6.06	1	16.72	.016
RESIDUAL	1.451	.36	4		
TOTAL	7.514				

MULTIPLE REGRESSION RESULTS

DEPENDENT VARIABLE: V8

INDEPENDENT VARIABLES: V18

MULTIPLE CORRELATION: .9210 F(1, 4) = 22.347 p = .010
R-square: .8482

BETA for V18 = .9209673 B = 3.2101341 t(4) = 4.727 p = .010

INTERCEPT = -2.9046363

Analysis of variance

	SS	MS	df	F	p
REGRESSION	6.373	6.37	1	22.35	.010
RESIDUAL	1.141	.29	4		

TOTAL 7.514

MULTIPLE REGRESSION RESULTS

DEPENDENT VARIABLE: V8

INDEPENDENT VARIABLES: V19

MULTIPLE CORRELATION: .9031 F(1, 4) = 17.689 p = .015

R-square: .8156

BETA for V19 = .9030920 B = .3194922 t(4) = 4.206 p = .015

INTERCEPT = -1.1230151

Analysis of variance

	SS	MS	df	F	p
REGRESSION	6.128	6.13	1	17.69	.015
RESIDUAL	1.386	.35	4		
TOTAL	7.514				

MULTIPLE REGRESSION RESULTS

DEPENDENT VARIABLE: V8

INDEPENDENT VARIABLES: V22

MULTIPLE CORRELATION: .0612 F(1, 4) = .015 p = .874

R-square: .0037

BETA for V22 = -.0612277 B = -.1179179 t(4) = -.123 p = .874

INTERCEPT = .3594728

Analysis of variance

	SS	MS	df	F	p

REGRESSION	.028	.03	1	.02	.874
RESIDUAL	7.486	1.87	4		
TOTAL	7.514				

MULTIPLE REGRESSION RESULTS

DEPENDENT VARIABLE: V8

INDEPENDENT VARIABLES: V25

MULTIPLE CORRELATION: .8472 F(1, 4) = 10.175 p =
.034
R-square: .7178

BETA for V25 = .8472442 B = .3591264 t(4) = 3.190 p =
.034

INTERCEPT = -1.3044070

Analysis of variance

	SS	MS	df	F	p
REGRESSION	5.394	5.39	1	10.18	.034
RESIDUAL	2.120	.53	4		
TOTAL	7.514				

MULTIPLE REGRESSION RESULTS

DEPENDENT VARIABLE: V8

INDEPENDENT VARIABLES: V17 V18 V19 V22 V25

MULTIPLE CORRELATION: 1.0000 F(5, 0) = .000 p
=******
R-square: 1.0000

BETA for V17 =-45.4355772 B = -56.9950411 t(0) = .000 p
=*****
BETA for V18 = 40.6040409 B = 141.5299048 t(0) = .000 p
=*****
BETA for V19 = 6.9138418 B = 2.4459509 t(0) = .000 p
=*****
BETA for V22 = 20.9756645 B = 40.3968890 t(0) = .000 p
=*****

BETA for V25 = -.6366466 B = -.2698592 t(0) = .000 p =*****

INTERCEPT = -62.8887842

* Warning: Multiple R is equal to 1.0.
* Significance of R and Beta cannot be
* calculated

MULTIPLE REGRESSION RESULTS

DEPENDENT VARIABLE: V7

INDEPENDENT VARIABLES: V8

MULTIPLE CORRELATION: .9425 F(1, 4) = 31.804 p = .006
R-square: .8883

BETA for V8 = -.9424870 B = -.7939900 t(4) = -5.640 p = .006

INTERCEPT = -.0016667

Analysis of variance

	SS	MS	df	F	p
REGRESSION	4.737	4.74	1	31.80	.006
RESIDUAL	.596	.15	4		
TOTAL	5.333				

MULTIPLE REGRESSION RESULTS

DEPENDENT VARIABLE: V7

INDEPENDENT VARIABLES: V15

MULTIPLE CORRELATION: .9336 F(1, 4) = 27.157 p = .008
R-square: .8716

BETA for V15 = -.9336054 B = -.0392191 t(4) = -5.211 p = .008

INTERCEPT = 1.1749051

Analysis of variance

	SS	MS	df	F	p
REGRESSION	4.648	4.65	1	27.16	.008
RESIDUAL	.685	.17	4		
TOTAL	5.333				

MULTIPLE REGRESSION RESULTS

DEPENDENT VARIABLE: V7

INDEPENDENT VARIABLES: V13

MULTIPLE CORRELATION: .9180 F(1, 4) = 21.427 p = .011
R-square: .8427

BETA for V13 = -.9179804 B = -.0013346 t(4) = -4.629 p = .011

INTERCEPT = .7672092

Analysis of variance

	SS	MS	df	F	p
REGRESSION	4.494	4.49	1	21.43	.011
RESIDUAL	.839	.21	4		
TOTAL	5.333				

MULTIPLE REGRESSION RESULTS

DEPENDENT VARIABLE: V7

INDEPENDENT VARIABLES: V8 V15

MULTIPLE CORRELATION: .9639 F(2, 3) = 19.635 p = .023
R-square: .9290

BETA for V8 = -.5373151 B = -.4526564 t(3) = -1.558 p = .217
BETA for V15 = -.4526687 B = -.0190158 t(3) = -1.312 p = .281

INTERCEPT = .5688068

Analysis of variance

	SS	MS	df	F	p
REGRESSION	4.954	2.48	2	19.63	.023
RESIDUAL	.378	.13	3		
TOTAL	5.333				

MULTIPLE REGRESSION RESULTS

DEPENDENT VARIABLE: V7

INDEPENDENT VARIABLES: V10

MULTIPLE CORRELATION: .6674 F(1, 4) = 3.212 p = .147
R-square: .4454

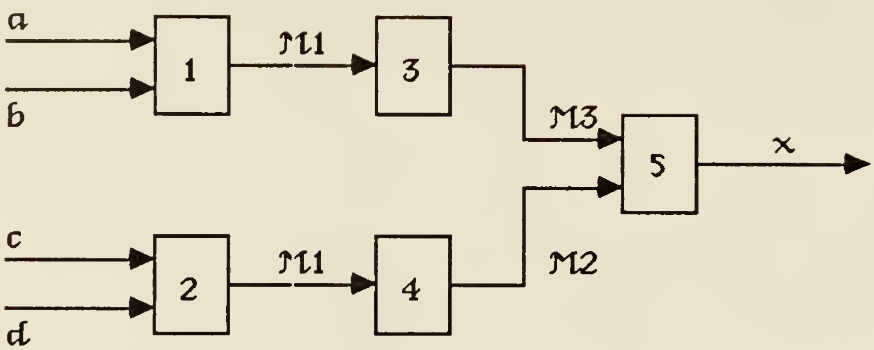
BETA for V10 = -.6673713 B = -1.5652981 t(4) = -1.792 p = .147

INTERCEPT = -.0016667

Analysis of variance

	SS	MS	df	F	p
REGRESSION	2.375	2.38	1	3.21	.147
RESIDUAL	2.958	.74	4		
TOTAL	5.333				

Example one :



Two possible TR representations :

1) starts with module '1'

```
                1           3                          2
'a','b'  ---> (M1  ---> ([M3], ('c','d'  --->
         4            <--    5
(M1  ---> (M2, M3  ---> "x" )))))
```

Atomic I/O relationships :

```
'a','b' --> M1(1)        (M1(1) -- the M1 from module '1')
M1(1) --> M3
'c','d' --> M1(2)        (M1(2) -- the M1 from module '2')
M1(2) --> M2
M3 --> x
M2 --> x
```

According to the above atomic I/O relationships, an I/O matrix can be built as shown on next page

	M1 (1)	M1 (2)	M2	M3	x
a	x				
b	x				
c		x			
d		x			
M1 (1)				x	
M1 (2)			x		
M2					x
M3					x

2) starts with module '2'

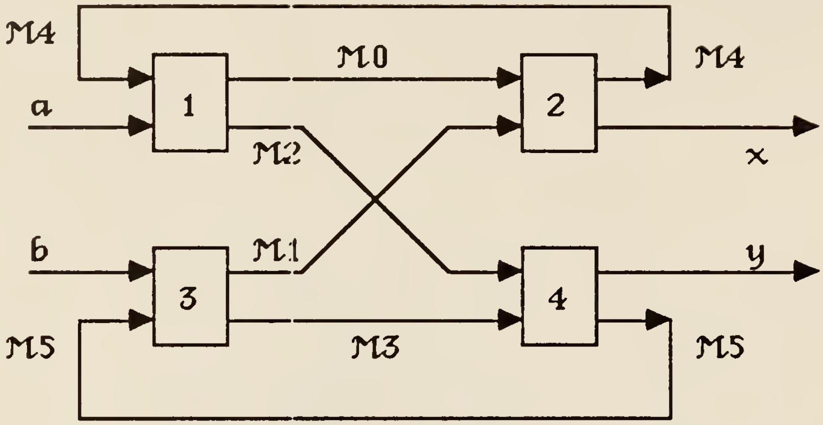
```
            2           4                          1
'c','d' ---> (M1 ---> ([M2],('a','b' --->
      <--    5
(M3, M2 ---> "x" )))))
```

Atomic I/O relationships :

'c','d' --> M1(2) ,   M1(2) --> M2
'a','b' --> M1(1) ,   M1(1) --> M3
M2, M3 --> x.

According to the above atomis I/O relationships, we will get exactly the same I/O matrix as the one shown above.

Example two :



Two possible TR representations :

1) starts with module '1'

$$\text{'a'},M4 \xrightarrow{1} ([M0],(\text{'b'},M5 \xrightarrow{3} [M3] \mid (M1,\overleftarrow{M0} \xrightarrow{2}$$

$$M4^* \parallel \text{"x"})) \mid (M2,\overleftarrow{M3} \xrightarrow{4} M5^* \parallel \text{"y"})$$

Atomic I/O relationships :

'a',M4 --> M0 | M2
'b',M5 --> M3 | M1
M1,M0 --> M4 | "x"
M2,M3 --> M5 | "y"

Two loops are involved :

M4 - 1 - M0 - 2 - M4
M5 - 3 - M3 - 4 - M5

2) starts with module '2' :

$$\text{'b'}, M5 \xrightarrow{3} ([M3], (\text{'a'}, M4 \xrightarrow{1} [M0] \mid (M2, \overleftarrow{M3} \xrightarrow{4}$$

$$M5^* \parallel \text{"y"})) \mid (M1, \overleftarrow{M0} \xrightarrow{2} M4^* \parallel \text{"x"})$$

Atomic I/O relationships :

'b',M5 --> M3 | M1
'a',M4 --> M0 | M2
M1,M0 --> M4 | "x"
M2,M3 --> M5 | "y"

Two loops are involved :

M4 - 1 - M0 - 2 - M4
M5 - 3 - M3 - 4 - M5

All the atomic I/O relationships and the loop paths are exactly the same as the one we developed from the first TR representation. Therefore, they correspond to the same DFD.

Evaluating Data Flow Diagrams

by

Qian Huang

B.S. Shanghai University of Technology, 1982

AN ABSTRACT OF A THESIS

submitted in partial fulfillment of the

requirements for the degree

MASTER OF SCIENCE

Department of Computing and Information Sciences

KANSAS STATE UNIVERSITY

Manhattan, Kansas

1989

Abstract

Data Flow Diagrams are important tools in software design. They can not only express the data flow within the software system but also partially show the control structure which is a critical factor. Therefore, evaluating data flow diagram is as important as other evaluations of software design. This is why some of the measures of software complexity were developed based on the data flow diagram, e.g., Henry- Kafura's. But one problem in calculating the measures is that one needs to calculate measures based on graphs which is not direct and which can lead to mistakes. This has motivated some researchers to try to develop better representations for data flow diagrams and put effort into making them easier to be formalized. One example is Adler's representation. There still exist problems in these representations. For example, it is not general enough to express different kinds of graphs. Thus, it is hard to use it as the base of calculating measures. Furthermore, a representation should reflect the characteristics of the whole graph so that useful measures can be derived directly from it and the evaluation of the data flow diagrams can be automated by storing knowledge of the representation and evaluation into an expert system.

The motivation of this thesis is to develop such an representation by extending Adler's. This representation should be useful as the basis of software measures for whole data flow diagrams. Another aim of this research is to acquire the knowledge of evaluating DFD by comparing the resultant measures and the evaluation of the DFDs by experts and to build classification categories with the help of membership functions of fuzzy set theory. This will provide a good basis for automatic evaluation of DFDs.

The thesis will cover the following :

1) Introduce the representation of DFD and discuss how it can reflect the characteristics of DFDs.

2) Develop and describe how to calculate measures directly from the representation and demonstrate it based on sample DFDs.

3) Compare the calculated measures with the expert's opinion and build up the membership functions for classification categories of the evalution of DFDs.