



A Rule-Based System with Numerical
Degree for Computer Configuration
Consultation

by

Mr. Masayuki Ando

Submitted in Partial Fulfillment of the
Requirements for the Degree of
Master of Science
In Computer Science
Assumption University

October, 2002

A Rule-Based System with Numerical Degree for Computer Configuration Consultation



By
Mr. Masayuki Ando

**Submitted in Partial Fulfillment of the
Requirement for the Degree of
Master of Science in
Computer Science
Assumption University**

October , 2002

The Faculty of Science and Technology

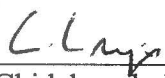
Master Thesis Approval

Thesis Title A Rule-Based System with Numerical Degree for Computer Configuration Consultation


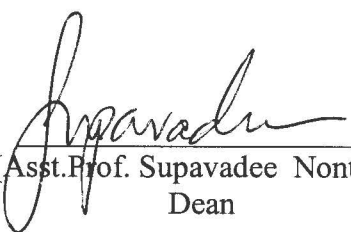
By Mr. Masayuki Ando
Thesis Advisor Professor Dr. Nguyen Hoang Phuong
Academic Year 1/2002

The Department of Computer Science, Faculty of Science and Technology of Assumption University has approved this final report of the **twelve** credits course. **SC7000 Master Thesis**, submitted in partial fulfillment of the requirements for the degree of Master of Science in Computer Science.

Approval Committee:


(Professor Dr. Nguyen Hoang Phuong)
Advisor
(Asst. Prof. Dr. Pratit Santiprabhob)
Committee Member
(Asst. Prof. Dr. Thitipong Tanprasert)
Committee Member
(Professor Dr. Chidchanok Lursinsap)
Representative of Ministry of
University Affairs

Faculty Approval:


(Asst. Prof. Dr. Tang Van To)
Program Director
(Asst. Prof. Supavadee Nontakao)
Dean

December/ 2002

Table of Contents

ACKNOWLEDGEMENT	i
LIST OF FIGURES	ii
ABSTRACT	iii
Chapter One: INTRODUCTION	
1.1 Motivation for Applying Computer to Computer Business	1
1.2 Motivation for Applying Rule-based System Technology for Consultation of Computer Configuration	2
1.3 Source of Information	2
1.4 Statement of the Problem	3
1.5 Objectives of the Study	3
Chapter Two: LITERATURE REVIEW	
2.1 Survey of Related Works	4
2.2 Some Expert Systems in Business	4
2.3 Comparison and Discussions	7
Chapter Three: BACKGROUND OF EXPERT SYSTEMS AND RULE-BASED SYSTEMS	
3.1 Expert System	8
3.1.1 Knowledge Base	9
3.1.2 Knowledge Representation Techniques	9
3.1.3 Working Memory	11
3.1.4 Inference Engine	12
3.1.5 Knowledge Acquisition	16
3.2 Set Operations	19

Chapter Four: COMPUTER RETAIL BUSINESS

4.1 Business World	24
4.2 What retailers are doing	24

Chapter Five: DEVELOPING THE RULE-BASED CONSULTATION SYSTEM FOR COMPUTER CONFIGURATION.

5.1 Introduction	28
5.2 Structure of the System	28
5.3 Steps for Establishing the system.	29
5.4 Testing	41

Chapter Six: IMPLEMENTATION AND PERFORMANCE OF THE SYSTEM

6.1 Programming Language	44
6.2 Performance of the System	44
a) Starting the System	44
b) Knowledge Acquisition	45
c) Inferring Advice	51

Chapter Seven: CONCLUSION

7.1 Summary	57
7.2 Future Works	57

References	58
------------	----

APPENDIX A: RULES	60
-------------------	----

APPENDIX B: SOURCE CORD (INFERENCE ENGINE)	71
--	----

APPENDIX C: SOURCE CORD (KNOWLEDGE BASE)	106
--	-----

ACKNOWLEDGEMENT

First of all I would like to thank my advisor, Professor Dr. Nguyen Hoang Phuong, for his kindness and invaluable advice.

Sincere thanks are also for all faculty members and my friends for their help and support, and for the committee members who advised to enhance my knowledge and this master's thesis.

I would also like to extend my heartfelt thanks to my uncle and parents who were unfailingly supportive while I studied this program. Their love, concern, and encouragement have strengthened me through my studies.



LIST OF FIGURES

Figure 3-1: Structure of Expert System	8
Figure 3-2: Relationship of Object, Attribute, and Value	10
Figure 3-3: Certainty Factor Used in MYCIN	11
Figure 3-4: Diagram of Forward Chaining Inference Process Cycle	13
Figure 3-5: Knowledge Acquisition Cycle	17
Figure 5-1: Structure of a Consultation System for Computer Configurations	28
Figure 5-2: Comparing Consumed Time for Advising by the Expert and the System	43
Figure 6-1: Top of the System Interface	44
Figure 6-2: Edit Computer Unit Using Microsoft Access	45
Figure 6-3: Top View of Knowledge Acquisition	46
Figure 6-4: List of Rules in the Knowledge Base	47
Figure 6-5: Dialog Box for Rule Edit	48
Figure 6-6: Confirmation for Deleting Rule	49
Figure 6-7: Configuration View	50
Figure 6-8: Confirmation for Deleting Selected Configuration Part	51
Figure 6-9: Inference Interface	52
Figure 6-10: Questions and Requests for Each Question	53
Figure 6-11: Requirements are Selected	53
Figure 6-12: Result for Requested Configuration	54
Figure 6-13: Did Not Satisfy with Reached Result	65
Figure 6-14: New Result with Lower Requests Configuration	66

ABSTRACT

An expert system is a computer program, which mimics a reasoning of domain expert(s) in decision-making. A fuzzy rule-based system is a reasoning system which is based on known fuzzy facts given by users and matches these facts with the fuzzy rules in rule base of the system in order to deduce any recommendation. My research is focused on applying fuzzy expert system technology in the problem of consultation for computer configurations where customer's requirements are mainly fuzzy information, for example, high CPU level, large monitor size, etc. These requirements can be represented as fuzzy variables and the knowledge of an expert to solve the problem can be represented as fuzzy rules.

The main aim of the thesis is to study and apply the fuzzy rule-based system into the problem of computer consultation. I describe the structure of the fuzzy rules based consultation system for computer configuration with the three main components as knowledge base, inference engine and knowledge acquisition.

The system will generate the best configuration for customers based on their needs. Then a data set consisting of many computer parts combination, which can be used for making one computer, is generated. Different combinations of computer part based on customers needs and based with related fuzzy degree will also be calculated.

The prototype system is implemented in the computer in V.B. 6.0 programming language with the rule base of about 300 fuzzy rules.

This fuzzy modeling can be applied for other similar problems where facts are fuzzy in nature, for example, in business and medicine, especially in Oriental medicine.

Chapter One: INTRODUCTION

1.1 Motivation for Applying Computers to Computer Business.

Since the Internet has been widely spread, most people hence started to use computer. This technical innovation has shown new usage of computers to people. Then people, who did not use a computer before, are interested in computers. By natural way of business, if consumers increase, producers will consider to develop new or better products to keep or gain their share of market. Also, development will lead to reduced costs. In computer business, products are developed every three months and their costs are reduced dramatically. Nowadays, you can buy a computer that cost 30,000 baht one year before for only 15,000 baht.

Now customers are different from those 10 years before. Computers are spread to the world very fast. New consumers are not be well educated about computer. The most recognized reason to start using a computer is motivation from others. Their friends or colleagues are using it, so they want to use it too. Starting something new is a very nice thing, but this situation brings difficulty to computer sellers, because they don't have clear reasons to use a computer. Most computer makers establish three or four configurations to consumers as their choice, even if consumers needs take up various topics.

In my actual situation, for example, customers do not know what they should buy and they do not have knowledge about computer parts, but when I ask them a few questions, they start to show their needs. Every time I ask for configuration I try to give my best suggestions at that time. My suggestions are based on my knowledge about computer parts. This would be better by using rule-based system to assist to solve this problem.

By using a computer, I thought that I could provide faster suggestions, reduce

teaching time to others and increase the quality of service.

1.2 Motivation for Applying Rule Based System Technology for Consultation of Computer Configurations.

One of the attributes of humans in problem solving often deals with ambiguity. Humans can handle ambiguous expression. An example of ambiguity in daily conversations is when people say, “This is too expensive.” From this sentence, most people understand that talking about the price of goods and this price is above normal price (maybe the product is high quality). In my business case, customers use ambiguous expressions such as “the price is high” or “I need middle grade computer”. To handle this ambiguous information in decision-making process with computers, degree of belief is helpful. Actual information that sellers gather through conversations is ambiguous information. From this information, we recommend to make one configuration that matches their needs. Our purpose is to apply rule-based expert system technology to model the reasoning of computer sellers in computer configuration consultations and develop a prototype system to help computer sellers in their business.

1.3 Source of Information.

Information about computer parts and questions were collected from my actual business situation. A list of computer parts was collected from one cooperating shop in Panthip Plaza. Questions were asked during conversations in actual work. I rearranged the list of computer parts more systematically according to the usage of the system. It was collected from the perception and experience of one person who is familiar with computer configurations and computer parts.

1.4 Statement of the Problem

An Expert system is a computer program that acts as a human expert. Nowadays, There are many expert systems in business because managers are considering an expert system will be a nice new employee.

In this study, I try to develop a consultation system for computer configurations. It can give one configuration which matches with a user's need. When a Human's feeling or conversation is very ambiguous, this system should be able to handle the ambiguity. During this study I tried to model and develop a rule-based system with a degree of belief on how computer parts are combined and computer configurations suggested by a system belongs to which level.

1.5 Objectives of the Study

The objectives of this study are to study and develop a rule-based consultation system for computer configurations. This study will help people who belong to the computer selling business to suggest computer configurations based on their customers' needs. Also if this system were spread through the Internet, this system would help potential customers to find the best configurations for them. The details of the objectives are as follows.

- 1) People in the computer business will reduce time for negotiation and cost. Also, companies will be more productive. New employees will give advice for their customers as experts. Also, companies will be able to reduce their training time for new employees.
- 2) Potential customers will reduce their research time to find the best configuration based on their needs. In the old way to find a computer configuration, people should visit computer shops and ask them about configurations. With this system, advice is produced automatically and they don't have to visit shops.

Chapter Two: LITERATURE REVIEW

2.1 Survey of Related Works. [5], [9], [11], [15], [20], [22], [23], [25]

In the field of artificial intelligence, the expert systems technology has been applied successfully in many application areas such as in medicine, geology, manufacturing, and especially in business and management. Today, there are probably about 2000 expert systems being used for business and nonbusiness applications. Expert systems for business and management such as in taxation, auditing, financial analysis, insurance and marketing have been constructed and used. But many of these systems were developed based on the traditional Artificial Intelligence methodologies, i.e. applying the mathematical tools for managing symbolic information to develop intelligent systems which can mimic the reasoning and/or behavior of human experts in problem solving. In recent years, computational intelligence techniques such as fuzzy logic, neural networks, genetic algorithms, probabilistic reasoning, etc, and the combination of these techniques were applied in developing intelligent systems.

In this review, we give some survey of some typical expert systems for consultations of business activities. We also compare our approach in applying the rule-based system with some degree of belief in the rule to develop a consultation system for computer configurations with the existing approaches and discuss the advantages of our approach.

2.2 Some Expert Systems in Business

The most considered point of expert systems for business is its productivity. If a company can reduce the consumed time for doing it or can reduce its cost, the expert system is considered as efficient. These are the related works of expert systems for business.

2.2.1 XCON/XSEL System [23]

XCON/XSEL System is an expert system developed jointly by Digital Equipment Corporation (DEC) and Carnegie Mellon University in the late 1970s. This system is developed for configuring a computer system in the DEC VAX 11/780 series to meet a specific customer order. Large computer systems, such as the VAX, have a great number of different component parts designed to perform different tasks. There are thousands of combinations of these parts, each combination represents a unique computer system tailored to a specific customer. XCON/XSEL takes the given customer system specifications and produces a plan for assembling the final system.

2.2.2 APEX : Sales and Marketing Support Systems [20]

APEX (APplied Expert Systems) builds expert systems to help financial services firms to more effectively manage their sales and marketing activities. These systems link the salesforce to the marketing and product development activities of the firm are able to add significant value to the sales process in the retail financial services. The products of expert analyses are a client report that contains comprehensive, objective financial advice to the client, a sales report that provides timely product information and sales points to the salesforce, and a data extract that contains the client information and associated recommendations. The data extract is used by the marketing function to better target market the firm's products, and by sales management and product development to understand and manage the sales process.

2.2.3 ANSWERS : An Expert System for Financial Analysis [23]

This system is designed for a very specific type of financial analysis. The system consists of two main modules: ratio analysis and projection analysis. The system is also used in a training application. The trend analysis module is designed to

focus on the three most critical audit areas – accounts receivable/sales, inventory/cost of sales, and payroll, and a general overview of the financial condition for the entity, which is simply called financial analysis. The project module is to provide the ability to perform statistically accurate forecasting applications. The ANSWERS system would be coded in FOXBASE for screen handling and TURBO-PASCAL for calculations. The system would be able to run on the IBM-DOS or any compatible type of computer.

2.2.4 RESERVE : Expert System for Estimating Casualty Insurance Loss Reserves

RESERVE is a rule-based expert system [2] that estimates casualty insurance loss reserves. Loss reserves are liabilities incurred by insurers for unpaid claims arising from insured events. The knowledge in RESERVE is represented as if-then rules. RESERVE is modeled on EXSYS, an expert system development tool that contains all of the elements of an expert system except the knowledge base. RESERVE uses a forward-chaining inference structure. This structure is used for two reasons. First, the subproblems must be sequentially solved. Second, one of the planned evaluations of RESERVE was for another actuary to review RESERVE's reasoning process. Forward chaining facilitates the reasoning evaluation by requiring the researcher to sequence the rules in a logical order. A limitation of RESERVE at its current stage of development is that it represents the estimation process of one domain expert and is inherently limited to that domain expert's experience and knowledge. The estimation process of RESERVE may not be generalized to other human experts.

2.3 Comparison and Discussions

In this thesis, I develop a prototype rule-based consultation system for computer

configurations to support sellers to choose an appropriate computer configuration according to a customer's requirements. The new feature of this approach in comparison with the existing methods in developing the expert systems in business mentioned above is to apply the numerical degree of belief to represent a customer's requirements as the level of computer parts and an inference engine to operate the ambiguous information to deduce the consultation about computer configurations. This approach is more natural than the existing methods above because a great deal of information in business has an ambiguous nature, and I try to mimic the reasoning of a human expert in decision making in consultations of computer configurations.



**Chapter Three: SOME BACKGROUND OF EXPERT SYSTEMS AND
RULE-BASED SYSTEMS**

3.1 Expert System [3], [10], [12], [26]

An expert system is one part of AI. It is a computer program that acts as a human expert. A human expert has domain knowledge in a particular field. This is a human effort to analyze the way of thinking and decision-making methods of a human expert to establish an expert system. An expert system consists of three main parts. A knowledge base, a working memory and an inference engine are its main parts as shown in figure 3.1. A knowledge base contains an expert domain knowledge that is used to solve a particular problem. A working memory is used during problem solving. It contains information that a user gives to the expert system. An inference engine uses a domain knowledge and information that are given by a user to solve a problem as expert.

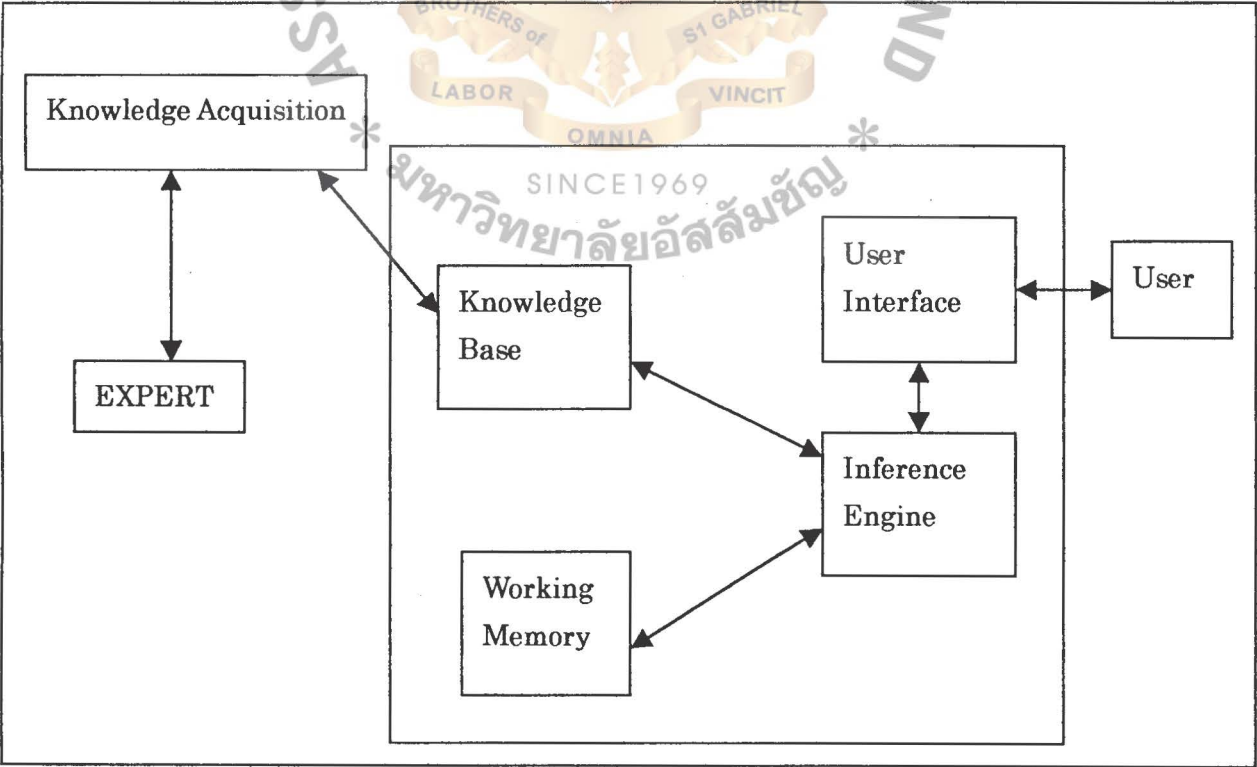


Figure 3-1: Structure of Expert System.

3.1.1 Knowledge Base [10], [12]

A knowledge base contains highly specialized knowledge on the problem area. A human expert provides this knowledge. It includes problem facts, rules, concepts, and relationships. The definition for knowledge base is:

Part of an expert system that contains the domain knowledge.

Now, I would like to propose the details.

Knowledge is understanding of a subject area and

Domain is a well-focused subject area.

There are few types of knowledge.

Procedural knowledge is how a problem is solved. This type of knowledge gives direction on how to do something.

Declarative knowledge is what is known about a problem.

Meta-knowledge is knowledge about knowledge. This type of knowledge is used to pick other knowledge that is best suited for solving a problem.

Heuristic knowledge describes a rule-of-thumb that guides the reasoning process. This is experimental knowledge that is gathered from experts who solved past problems.

Structural knowledge describes knowledge structure. The expert's mental model of concepts, subconcepts, and objects is typical of this type of knowledge.

3.1.2 Knowledge Representation Techniques [10], [27]

A number of effective ways of representing knowledge in a computer were developed. I would like to introduce some techniques.

(1) Object-Attribute-Value Triplets (O-A-V Triplets)

All human knowledge uses facts as basic building blocks. A fact is a form of

declarative knowledge. In expert systems, facts are used to help describe parts of frame, semantic networks or rules. Facts are also used to represent relationships between more complex knowledge structures. A fact is often called as a proposition.

A Proposition is a statement that is either true or false.

A proposition can be a simple statement as “this is expensive”. The Answer for this question will be stored in the working memory of an expert system.

O-A-V is a more complex type of proposition. Suppose there is a sentence “monitor’s quality is high”. “Monitor quality” is object and “is” as attribute and “high” for value.

O-A-V is only applied to physical items.

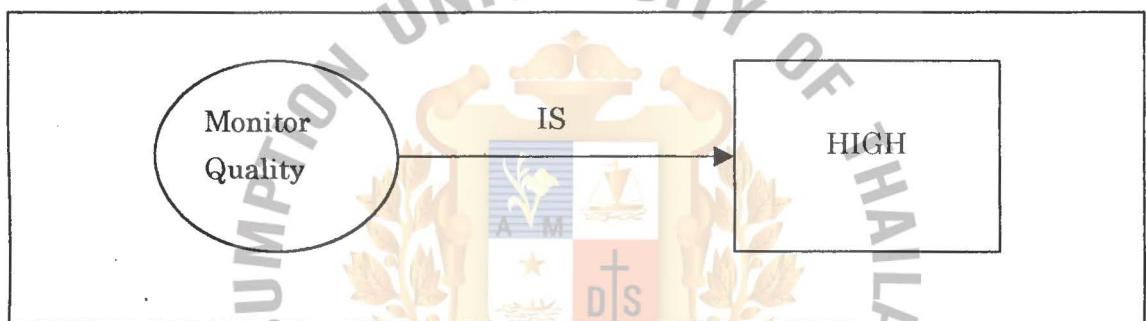


Figure 3-2: Relationship of Object, Attribute, and Value

(2) Rules

the definition for rules is:

A knowledge structure that relates some known information to other information that can be concluded or inferred to be known.

A rule is a form of procedural knowledge. It associates given information to some action. In a sense, rules are describing how to solve problem.

A structure of rule is usually connecting one or more antecedents (or premises) to one or more consequences (or conclusion).

To represent a expert’s knowledge in computer, usually rules are applied. Basically, modus ponens is used for rules. Modus ponens is IF – THEN style as below.

IF premise THEN conclusion

For example,

IF today is Sunday THEN you don't have to go company

In IF – THEN style, when the premise of rule is true, the conclusion is fired.

Now I would like to show an example of rule with weights. Assume that there is a sentence that describes:

IF Today is Monday and it is 9:00 am THEN I will be late for class ($w = 0.8$)

w stand for weight. ($w = 0.8$) shows the possibility of the conclusion in this case.

Rules' weights are not only for possibilities but also show importance of rules.

Numbers between 0 and 1 are usually chosen to represent weights.

In a conventional expert system, especially in MYCIN [10],[12], a certainty factor is applied to express ambiguity. Ambiguity is expressed numbers between -1 and 1 . I thought that certainty factor is more difficult to understand by system users.

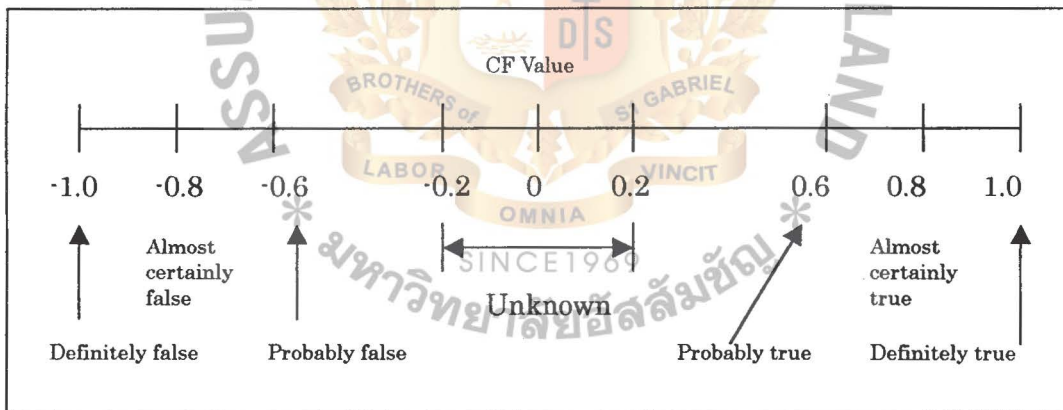


Figure 3-3: Certainty factor used in MYCIN

3.1.3 Working Memory

A working memory contains information from a user and reached conclusion during problem solving. At first, there is nothing in a working memory. When a user answers questions, that answer is stored in a working memory. Based on this answer matched rule is inferred and reached conclusion is stored in a working memory. The

definition for working memory is: **Part of an expert system that contains the problem facts that are discovered during the session.**

A working memory is like a short-term memory of humans. Some truth will get in here and from the truth some conclusions will be generated. Then, the conclusions will be stored in the working memory.

3.1.4 Inference Engine

An inference engine searches a knowledge base and a working memory to solve a problem. The definition for inference engine is:

Processor in an expert system that matches the facts contained in the working memory with the domain knowledge contained in the knowledge base, to draw conclusions about the problem.

The inference engine works with the facts and knowledge to derive new information. It searches the rules for a match between their premise and fact or information that is contained in a working memory. When the inference engine finds a match, it adds a conclusion of rule to the working memory, and continues to scan other rules to find other matches.

Forward Chaining

The solution process for some problems naturally begins by collecting information. This information is then reasoned with to infer logical conclusions. This style of reasoning is modeled in an expert system using data-driven search. It is also called forward chaining. The definition of forward chaining is:

Forward chaining is an inference strategy that begins with a set of known facts, derives new facts using rules whose premises match the known facts, and continues this process until a goal state is reached, or until no further rules have premises that match the known or derived facts.

An example for the process of forward chaining in a rule-based system proceeds as follows.

1. Information from user is stored into the working memory.
2. The inference engine then scans the rules whose premise match the contents in the working memory.
3. If it finds a rule, the rule's conclusion is added to the working memory. Then it goes back to the first rule and scans rules again. In this cycle, fired rules are ignored.
4. If it cannot find rules whose premises match the contents of the working memory, and no more rules remain, then it stops the inference.

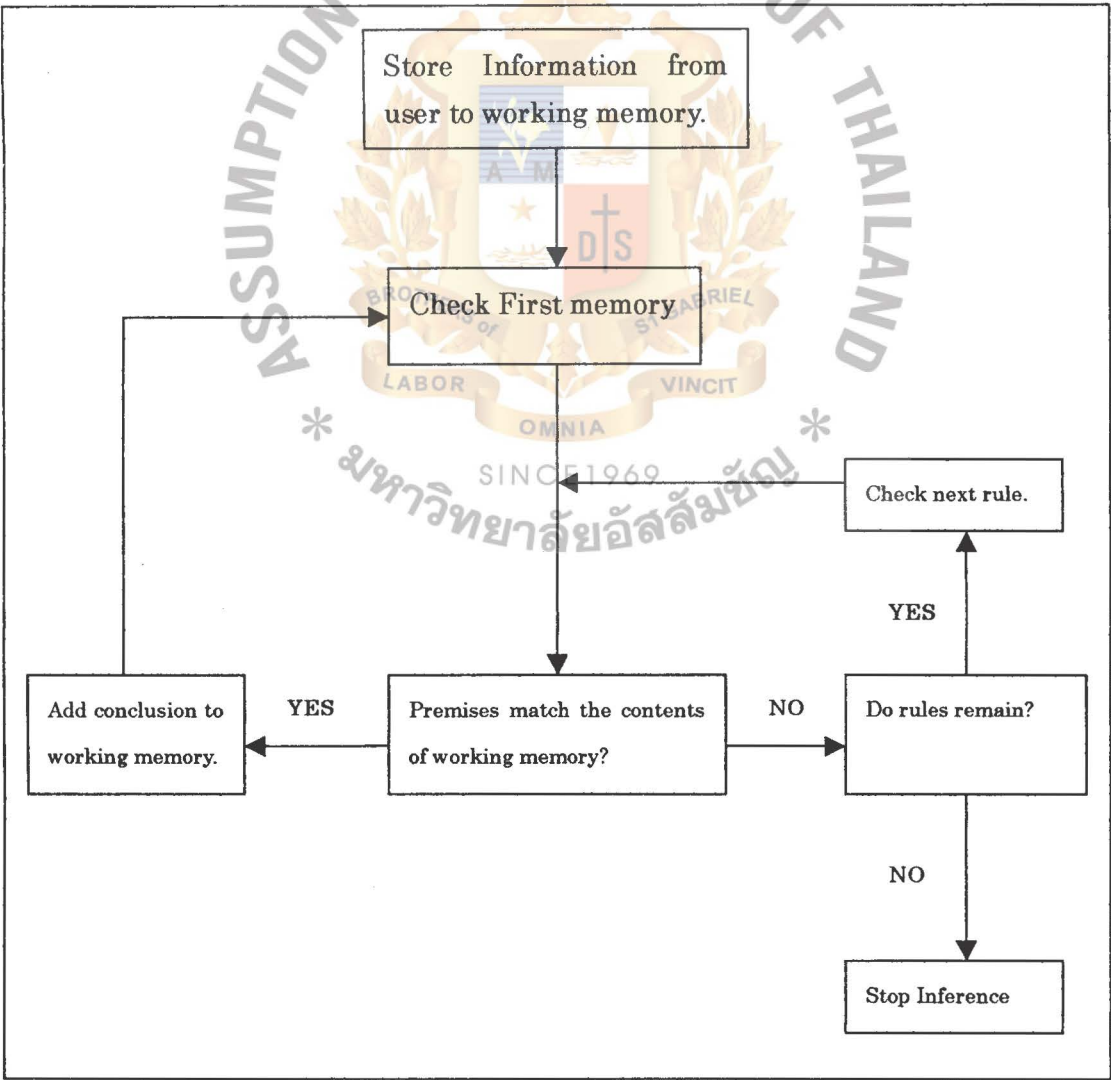


Figure 3-4: Diagram of forward chaining inference process cycle

Some advantages of forward chaining

1. A major advantage of forward chaining is that it works well when the problem naturally begins by gathering information and then seeing what can be inferred from it.
2. Forward chaining can provide a considerable amount of information from only a small amount of data.
3. Forward chaining is an excellent approach for certain types of problem solving task, such as planning, monitoring, control, and interpretation.

Some disadvantages of forward chaining

1. One of the major disadvantages of a forward-chaining system is that it may have no means of recognizing that some evidence might be more important than others. The system will ask all possible questions, even though it may only need to ask few questions to arrive at a conclusion.
2. The system also asks unrelated questions though the answers to these questions may be important. It is disconcerting to users to answer questions on unrelated subject.

Backward Chaining

In backward chaining, I begin with a hypothesis and then attempt to prove it by gathering information. This style of reasoning is modeled in an expert system using goal-driven search. The definition of backward chaining is:

Backward Chaining is an inference strategy that attempts to prove a hypothesis by gathering supporting information.

Backward chaining system begins with a goal to prove.

1. It first checks the working memory to see if the goal has been previously added.

2. If the goal has not been previously proven, the system searches its rules looking for one (or more) that contains the goal in its THEN part.
3. The system checks to see if the goal rule's premises are listed in the working memory.
4. If the premises are not listed in the working memory then a rule's conclusion will be a new goal to prove (sub-goal). A Sub-goal may be supported in other rules.
5. It continues the searching process until the system finds a premise that is not supported by any rule. This type of rule is called primitive.
6. The system asks question about primitive.
7. It uses the answer from a user about primitive to prove the sub-goal and the original goal.

For example, assume that there are rules as:

Rule 1: IF virus affects your computer

THEN Computer doesn't work

Rule 2: IF you recently received a strange mail

THEN virus affected a computer.

Current problem is computer doesn't work.

In this example, the expert system starts to search rules that contain a goal (computer doesn't work) in its conclusion part. If the expert system finds rule 1 then it considers the rule's premise (virus affect your computer). Now the computer searches for a new goal and finds Rule 2. Then the expert system searches rules that contains the new goal (you recently received a strange mail) in its conclusion part. There are no more rules. So the expert system asks the user "Did you recently receive a strange mail?". If the answer is yes, this is the reason for why the computer doesn't work.

Some advantages of backward chaining

1. One of the major advantages of a backward chaining system is that it works well

when the problem naturally begins by forming a hypothesis and then seeing if it can be proven.

2. Backward chaining remains focused in a given goal. This produces a series of questions on related topics, a situation that is comfortable for the user.
3. Whereas a forward chaining system attempts to infer everything possible from available information, a backward chaining system searches only that part of the knowledge base that is relevant to the current problem.
4. Backward chaining is an excellent approach for certain types of problem solving tasks, such as diagnostics, prescription, and debugging.

Disadvantage of backward chaining

1. The principal disadvantage of a backward chaining system is that it will continue to follow a given line of reasoning even if it should drop it and switch to a different one.

Summary for forward chaining and backward chaining

Forward chaining is data-driven search technique. It searches for a conclusion that matches information to premise. On the other hand, backward chaining is a goal-driven search technique. It searches for reason for goals.

In the nature of my thesis, first I gather information from the users. Then the system searches for conclusion (computer configuration) based on user's needs. Then I should apply forward chaining to a consultation system for computer configurations.

3.1.5 Knowledge Acquisition.

To make a knowledge base, an expert's knowledge about a problem must be gathered. Gathered knowledge is used to provide both insight into the problem and the

material for the design of the expert system.

Definition for knowledge acquisition is

The process of acquiring, organizing, and studying knowledge. [10]

There are many ways to gain knowledge for a problem.

Expert – the primary source of knowledge for most expert system projects.

End user – a valuable additional source of information. They give information about general understanding of the problem.

Multiple experts – another source of knowledge. Gaining information from multiple experts will make confusion.

Literature – additional sources of information may come in the form of documents, such as reports, regulations, guidelines and books.

During knowledge acquisition, these tasks are involved.

Collect – collecting knowledge from an expert.

Interpret – following the collected information, I should interpret to find key knowledge for a problem.

Analyze – from interpretation, I should know how to use key knowledge.

Design – from gathered information and key knowledge found try to form some new understanding or new concepts.

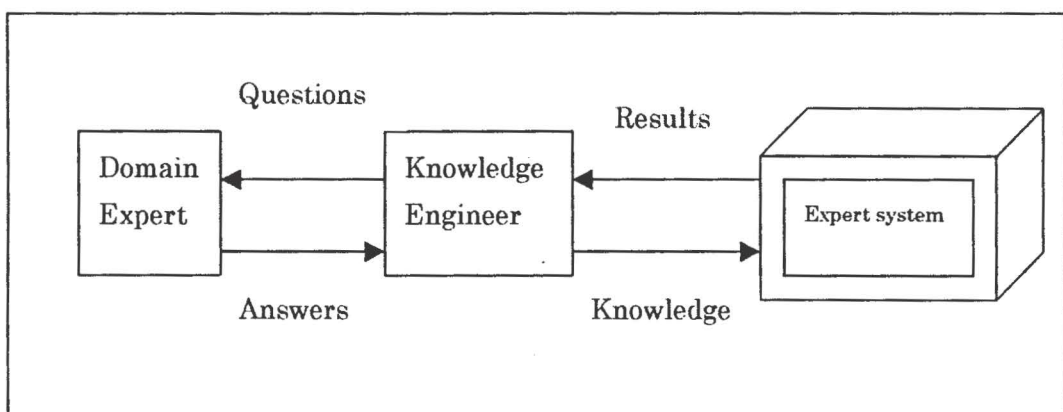


Figure 3-5: Knowledge acquisition cycle.

Difficulties with knowledge acquisition

During elicitation of the knowledge I will meet with these difficulty.

1. Experts may be unaware of the knowledge used.

Experts usually acquire their expertise through the experience of working on similar problems. After a period, experts may become unaware of the deep knowledge they use to solve the problem.

2. Experts may be unable to verbalize the knowledge.

There are difficulties to express actions in words.

3. Experts may provide irrelevant knowledge.

Experts may provide invaluable knowledge during knowledge acquisition. Knowledge engineers should get some sense from gathered information.

4. Experts may provide incomplete knowledge.

Experts may often provide an incomplete description of their mental processes. When an expert introspects about problem-solving knowledge, he or she may often leap over some of the intuitive issues.

5. Experts may provide incorrect knowledge.

Experts may make simple mistakes during introspect.

6. Experts may provide inconsistent knowledge.

Inconsistent knowledge is often found in an expert's explanation of a problem-solving strategy.

Knowledge acquiring method

Knowledge acquiring methods can be categorized into two general areas: direct and indirect techniques. Direct methods involve the articulation by the domain expert of the problem-solving knowledge and primarily include interviews and case studies. Indirect methods do not rely on experts being able to articulate the knowledge.

Common indirect methods involve questionnaires, repertory grid, and card sorting.

The most common knowledge acquisition technique used in the design of an expert system is the interview method. This technique provides quick understanding about a problem and a solution for the problem.

Case studies contain both the steps taken to solve a problem and the final solution. A case is an actual problem that has been solved in the past. There are two primary ways that a case is used during knowledge acquisition session. One is retrospective: the expert is asked to review the case and explain in retrospect how the problem was solved. The other way is observational technique that involves asking the expert to solve the problem discussed in the case while I observe.

3.2 Set operations. [16], [17]

In order to connect numerical sets, I use some connectives “AND”, “OR”, “NOT”..., Now I review some conventional set operations.

Intersection (t-Norm)

In the classical set theory, the intersection of two sets contains some elements that belong to both sets. The intersection of two sets contains a part of the elements because the elements have some degree of membership between 0 and 1. I can't say one element that belongs to both sets is more likely to the other set.

The operation for intersection of two sets is given below.

$$\begin{aligned}\text{degree } A \wedge B(X) &= \min(\text{degree } A(x), \text{degree } B(x)) && \text{for all } x \in X \\ &= \text{degree } A(x) \wedge \text{degree } B(x) \\ &= \text{degree } A(x) \cap \text{degree } B(x)\end{aligned}$$

For example, there are two degrees of belief, “low price” and “high price”. In this case

I would like to show two levels of degree of belief as below:

$$\text{Low price} = (1/0, 0.8/5, 0.5/10, 0.2/15, 0/20)$$

$$\text{High price} = (0/0, 0.2/5, 0.5/10, 0.8/15, 1/20)$$

For the interpretation of the notation, 1/0 means when price is 0 baht then membership function of low price is 1. 0.8 at 5,000 baht, 0.5 at 10,000 baht and so on.

According to the formula above, I get the intersection of low and high price, as

$$\text{degree}_{\text{low price} \wedge \text{high price}}(x) = (0/0, 0.2/5, 0.5/10, 0.2/15, 0/20)$$

For the intersection of the two sets, I take their minimum.

Union (t-Conorm)

Union is also defined as “OR”. Union infers to elements that belong to one of two sets or both, easily, elements that belong to set A or set B. In this case, the membership degree can't be less than the membership degree of the original set. Then I take the max value during an operation. The operation for the union of two sets is

$$\begin{aligned} \text{degree}_{A \vee B}(x) &= (\text{degree}_A(x), \text{degree}_B(x)) \quad \text{for all } x \in X \\ &= \text{degree}_A(x) \vee \text{degree}_B(x) \\ &= \text{degree}_A(x) \cup \text{degree}_B(x) \end{aligned}$$

For example, recall two level of degrees of belief, low price and high price. The union of these two sets will be

$$\text{Low price} = (1/0, 0.8/5, 0.5/10, 0.2/15, 0/20)$$

$$\text{High price} = (0/0, 0.2/5, 0.5/10, 0.8/15, 1/20)$$

And according to the formula,

$$\text{degree}_{\text{low price} \vee \text{high price}}(x) = (1/0, 0.8/5, 0.5/10, 0.8/15, 1/20)$$

You can see that I take the maximum value of each element for the union operation.

Complementation (Not)

I can find complementation by

$$\text{degree}_{\text{not } A}(x) = 1 - \text{degree}_A(x)$$

For example, the degree of belief not high price is

$$\text{NOT high price} = (1/0, 0.8/5, 0.5/10, 0.2/15, 0/20)$$

The three sets of operations mentioned above are basic. In a case of difficulties, different classes of functions are proposed.

Now we would like to look at more details of t-Norms and t-Conorms.

More about t-Norms (AND operation)

Any intersection function i must have appropriate properties, which ensure that numerical sets produced by i are intuitively acceptable as meaningful numerical intersections of any given pair of numerical sets. The axiom for intersection: t-norm is here:

Axiom 1. $i(a, 1) = a$ (boundary condition)

Axiom 2. if $b \leq d$ then $i(a, b) \leq i(a, d)$ (monotonicity)

Axiom 3. $i(a, b) = i(b, a)$ (commutativity)

Axiom 4. $i(a, i(b, d)) = i(i(a, b), d)$ (associativity)

The intersection function i must satisfy these axioms.

There are some more restrictions for the intersection function to reduce the class of intersection.

Axiom 5. i is a continuous function (continuity).

Axiom 6. $i(a, a) < a$ (subidempotency).

Axiom 7. $a_1 < a_2$ and $b_1 < b_2$ implies $i(a_1, b_1) < i(a_2, b_2)$ (strict monotonicity)

Axiom 5 prevents a situation in which a very small change in the membership grade

of either set A or B leads a to large change in the membership grade of intersection of A and B. Axiom 6 is used in a special case of intersection where the membership grade of A and B is at the same value a, then the intersection of A and B should not exceed a. Axiom 7 expresses a stronger form of monotonicity.

The following are examples of frequently used numerical intersections.

Standard intersection: $i(a, b) = \min(a, b)$

Algebraic product: $i(a, b) = ab$

Bounded difference: $i(a, b) = \max(0, a+b-1)$

Drastic intersection: $i(a, b) = a$ when $b=1$
 $= b$ when $a=1$
 $= 0$ otherwise.

More about t-Conorms (OR operation)

Union function u must satisfy at least these axioms.

Axiom 1. $u(a, 0) = a$ (boundary condition).

Axiom 2. $b \leq d$ implies $u(a, b) \leq u(a, d)$ (monotonicity).

Axiom 3. $u(a, b) = u(b, a)$ (commutativity)

Axiom 4. $u(a, u(b, d)) = u(u(a, b), d)$ (associativity)

These axioms are essential for numerical unions.

The most important additional requirements for numerical unions are expressed as follows:

Axiom 5. u is continuous function (continuity).

Axiom 6. $u(a, a) > a$ (superidempotency).

Axiom 7. $a_1 < a_2$ and $b_1 < b_2$ implies $u(a_1, b_1) < u(a_2, b_2)$ (strict monotonicity)

Here are some examples for union function that are frequently used.

Standard union: $u(a, b) = \max(a, b)$

Algebraic sum: $u(a, b) = a + b - ab$

Bounded sum : $u(a, b) = \min(1, a+b)$

Drastic union: $u(a, b) = a$ where $b=0$, b where $a=0$, 1 otherwise.



Chapter Four: COMPUTER RETAIL BUSINESS

4.1 Business world. [6], [7], [19], [27]

Once some businesses become as cash cow, many companies will get into that business to make profits. Since the Internet was introduced to consumers, the consumption of computer was dramatically increased. Currently, there are many companies in computer retail business. As the nature of business theory, if the numbers of competitors increase, companies should increase competitive edge to survive.

The competitive edge is the quality of goods and/or quality of service and so on that helps company to compete with others. For small retailers, to increase the quality of service is the most important. The reasons why companies reach this conclusion are as below. Computer retailers sell computers to their customers. There are many companies that produce each part of a computer. The retailer's decide which brand's goods to handle in their shop based on their beliefs. So if the choice is wrong, maybe they can change their choice of goods. Price is the most attractive feature to the customers. If customers can buy the same product at different prices, the customers will choose the lower priced product. This method is very useful when you have enough number of customers. Reducing price has direct impact to profit. To cover lost profit, small retailers should look for new customers. This is difficult to small retailers. Suppose that there are two retailers that sell the same products at the same price but of different quality of service. In this situation, customers will choose the retailer that serves better quality of service.

4.2 What retailers are doing.

Nowadays, you can find catalogs of computers that are issued by retailers. You can choose one configuration from the catalogs to make an order. Is this sales method convenient enough for customers? The answer is NO. I found that customers would choose the nearest configuration to their needs. I thought that they could buy products that match their needs. I believe that this service become competitive edge.

Now I would like to show what I did 1) define their needs through asking questions; 2) suggest configurations based on their needs; 3) negotiate to approach the best configuration; and 4) suggest after negotiation.

Now I would like to show each step briefly.

1) Define their needs through asking questions.

Usually these questions are asked to gather information about computer configurations which users wish to buy. The following are examples of questions. Normally, the users whom we met did not have much knowledge about computers. Questions should be friendly for the users. (I thought that daily conversation would be friendly for them) These questions were asked to know what they really want to buy. Examples of questions:

How much will you spend on computer?

What PC level do you want?

Will you use this PC as a server?

and so on.

2) Suggest configuration based on their needs.

During this step, I try to suggest computer configurations from their answers.

At first I try to show the computer configuration that they really wanted to buy regardless of their budget. Before I show why we do like this, I would like to tell about the users I met.

From my experience, users can be categorized into three types. (1) People who start researching about computers; (2) people who exactly don't have any knowledge about computers; and (3) people who have knowledge about each part of a computer. People who belong to (1) and (2) don't have enough knowledge about each part of a computer, so they search for computers that will meet their budget. On the other hand, people belong to group (3) tend to search for computers based on the configurations they want.

Now I show why I do like this. Most people I met belong to group (1) or (2). They don't know so many things about computer parts. So I should show exactly what they had better buy to satisfy their needs but not their budget.

3) **Negotiate to approach the best configuration.**

In step (2), we told that I show configurations that are based on their needs but not their budget, because based on the natural reaction of people, when they find better things they tend to buy a better one if its price is in an acceptable range for them. This is exactly beyond their thinking. Some people accept the configurations that are shown in the second step, while others do not. Why don't they accept the configurations? The answer is because price is not acceptable for them. So we start the negotiation. To reach their needs, I must know what is the most important thing for them. Based on the important things for them, I try to change the configurations to meet their budget. If the most important thing were budget, I would suggest a configuration whose price is closest to their budget. If there were some configurations on the same price, I would show the one with the

highest CPU speed one.

Suggestion after negotiation.

Show the configurations after negotiation. This will be the final answer from me. This answer is closest to their budget and keeps things they need.



**Chapter Five: DEVELOPING THE RULE-BASED CONSULTATION SYSTEM
FOR COMPUTER CONFIGURATION.**

5.1 Introduction

Based on the problems buying computer devices, customers don't have much knowledge about computer parts. They couldn't decide which configurations should be bought. It is easier for the customers to buy when advisers ask them some questions such as "Which CPU do you want?, Do you want to play DVD on this PC?, Which monitor size do you want?". Depending on the responses from the customers such as "I like very high quality CPU, play DVD on the PC and large monitor size", the advisers can recommend configurations that best fit the customer's requirements.

In fact, it is also important to gather knowledge and experiences from advisers, technicians and engineers in the computer field to help customers.

5.2 Structure of the system

Here is a figure of the structure of the system.

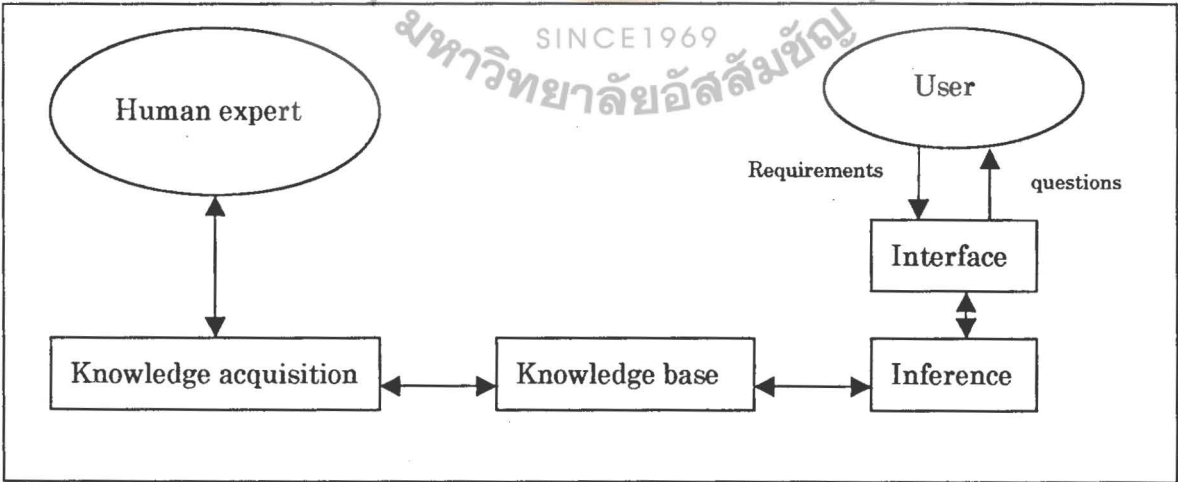


Figure 5-1: Structure of a consultation system for computer configurations

- Knowledge acquisition: collects a customer's requirements with their types and

degrees, delete data, edit or modify rules in the knowledge base.

- Knowledge base: consists of all of the requirements, types and degrees, inferential rules as well as standard configurations and relative price.
- Inference engine: from selected requirements of customers, the inference engine will check all rules on knowledge base in turn and give suitable configurations and prices fitting to the requirements.
- Interface: for interaction between customers and the system in menu form.
- Explanation: helps customers to understand how the recommended configurations and process are reached by tracing all the rules fired during a consultation.

5.3 Steps for establishing the system.

The structure of an advising system for buying computers can be implemented as follows:

- 1) Define the problem
- 2) Define the linguistic variables
- 3) Define the parts of computer sets
- 4) Define the rules with degree of belief
- 5) Build the system
- 6) Test the system
- 7) Tune the system

1) Define the problem for the system.

Nowadays, customers can find a catalog of computers that is issued from retailers. They can choose one configuration from the catalog to make an order. Is this sales method convenient enough for customers? The answer is NO. I found that the customers would choose the configurations nearest to their needs. I thought that they

could buy products that match their needs. I believe that this service becomes the competitive edge.

In fact, to advise customers, computer sellers will do the following:

1. Define customers' needs through asking questions.
2. Suggest configuration that based on their need.
3. Negotiate to approach the best configuration
4. Suggest after negotiation.

Details for these steps are in chapter 3.2.

2) Define computer parts.

In this section I would like to show the computer parts that are used in the system. These computer parts are recognized in an actual seller's work. When they ask questions about computer configurations, they ask about the size or quality of the part. Sellers ask questions, such as "Which CPU level do you want "or" Do you need high quality VGA card." From their questions, I define linguistic variables as follows:

CPU level

Memory capacity

Monitor size

VGA card quality

Hard disk capacity

CD-ROM quality

The reason why I use these parts is because sellers ask questions about them frequently.

3) Define numerical level for each set.

For each computer part, we assign high quality, medium quality and low

quality or large, medium and small. For example, the monitor will be categorized as large size monitor, medium size monitor and small size monitor. The CPU level will be categorized as high quality, medium quality, and low quality. “High, Medium, Low” or “Large, Medium, Small” are primary terms of each computer part.

Each primary term takes five linguistic levels as “sure, almost, middle, possible, sure not”. In the case of a large size monitor, they will appear as “sure large, almost large, middle large, possible large, and sure not large.” Each linguistic level represents the numerical interval of the degree of belief. This relationship between linguistic level and numerical interval is below.

Considered levels for each computer part	Numerical interval
Sure	1
Almost	[0.6-0.99] ~ 0.75
Middle	[0.4-0.59] ~ 0.5
Possible	[0.1-0.39] ~ 0.25
Sure Not	0

We take 0.75, 0.5, and 0.25 as the average values of intervals [0.6-0.99], [0.4-0.59], [0.1-0.39].

In similar way, we establish the relation between the level of primary and numerical value for “sure medium, almost medium, middle medium, possible medium, sure not medium” and “sure small, almost small, middle small, possible small, sure not small”. Also for primary terms “high, medium, low”, we apply the levels for computer parts as sure, almost, middle, possible, and sure not in similar way.

4) Define rules with weight (degree of belief for rules)

In this system rules appear as

IF Requirement 1 is Lev₁ **AND**
Requirement 2 is Lev₂ **AND** **AND**
Requirement _n is Lev _n
THEN CompConfigTyp _j
degree (CompConfigTyp _j)

Where Requirement _i (_i = 1, _n) is a computer part and Lev _i is a primary term. CompConfigTyp _j (_j = 1, _m) is a computer configuration level and degree (CompConfigTyp _j) is a degree of sureness that the rule indicates the degree of belief about a confirmation of the conclusion. Degree (CompConfigTyp _j) takes values in [0, 1].

for example of a rule is as follows:

IF CPU level is High **AND**
Monitor size is Large **AND**
VGA card quality is High **AND**
Hard disk capacity is Large **AND**
Memory capacity is High **AND**
CD-ROM quality is High
THEN Computer configuration is High
degree (Computer configuration is High) = 1

The computer configurations are divided into five types: High, Almost high, Middle, Almost low, and Low.

5) Build the system

The main parts of the system are the knowledge base and the inference

engine. I have already talked about knowledge base in previous sections. Now I would like to focus on the inference process of the system.

The process of the inference system can be divided into the following steps:

Step 1: Input numerical value as degree of level of primary term

At first, from the user input, we set the numerical value that is related to linguistic level as level of primary term. For example, we got

“CPU Level is sure high” = “CPU level (high) = 1”

I do this procedure to all inputs.

Step 2: Search for rules

After step 1, the system tries to search for rules that match to current inputs. If there are rules that match to current inputs, the system fires these rules and calculates the degree of belief for computer configurations by using a suitable t-norm operation (using min operation) to form a firing strength that indicates the degree to which the premise part of the rule is satisfied.

$$\text{degree of belief for configuration level } j = \text{Lev } 1 (\text{requirement } 1) \wedge \dots \wedge \text{Lev } n (\text{requirement } n) \wedge \text{degree of belief for rule}$$

For example, assume that there is a rule “IF CPU Level is high THEN computer configuration is Medium (0.21)”. From example of step 1, we know that the user input is CPU Level (high) = 1. Then the configuration level of the computer will be Medium = 0.21.

It is calculated from $\text{Min}(1, 0.21) = 0.21$.

After calculation, the inference system starts searching for the next match.

Step 3: Induce conclusions for each level

If we have several rules, r_1, \dots, r_m leading to the same conclusion “CompConfigTyp_j”, then the degree (CompConfigTyp_j), to which one of them is applicable for a given effect, CompConfigTyp_j, is:

$$\mu_{\text{Total}}(\text{CompConfigTyp}_i) = \mu_{\text{rule1}}(\text{CompConfigTyp}_i) \vee \dots \vee \mu_{\text{rule}_m}(\text{CompConfigTyp}_i)$$

where \vee is a suitable t-conorm (we take \vee as a max operation, i.e. $a \vee b = \max(a, b)$).

Step 4: Overall output as final decision.

The system takes the computer configuration level with the highest degree as the final decision.

Assume that if there are some distinct sets of rules leading to the different conclusions in the knowledge base, then the degree of the final conclusion is a maximum of all overall output membership functions computed in step 4.

If reached computer configuration levels with degree are

$$\text{CompConfig}(\text{high}) = 0.9$$

$$\text{CompConfig}(\text{almost high}) = 0.7$$

the system chooses $\text{CompConfig}(\text{high})$ as the answer.

If there are different computer configurations with the same degree, the system takes the higher level. From the above example if $\text{CompConfig}(\text{almost high}) = 0.9$, then the system takes $\text{CompConfig}(\text{high}) = 0.9$ as the answer.

After the system chooses the computer configuration level, the system searches for the computer configuration that has the nearest degree of belief to the degree of the computer configuration level. Computer configurations are listed in database that is named by level, such as “high”.

For Example:

Given a part of rule base:

R1: IF CPU level is High **AND**

Monitor size is Large AND

VGA card quality is High

THEN Configuration is Almost High

degree (Configuration is Almost High) = 0.9

R2: IF CPU level is High AND

Hard disk capacity is Small

THEN Configuration is Almost High

degree (Configuration is Almost High) = 0.2

Now I apply an inference process described above into this rule base to deduce a configuration.

Assume that the inputs are given as follows:

The term “CPU level is High” takes the level of primary term as “Sure High”.

According to the relation between the level of primary terms and numerical values of requirements in Table 1,

degree_{High} (CPU level) = 1.

degree_{Medium} (CPU level) = 0.

degree_{Low} (CPU level) = 0.

In similar way, we get:

The degree of terms “Monitor size is Large” takes the linguistic value “Sure Large” to the linguistic variable “Monitor size”, i.e. degree_{Large} (Monitor size) = 1, degree_{Medium} (Monitor size) = 0, and degree_{Small} (Monitor size) = 0

The degree of term “VGA card quality is High” takes the linguistic level “Sure High” to the computer part “VGA card quality”, i.e. degree_{High} (VGA card quality) = 1, degree_{Medium} (VGA card quality) = 0, and degree_{Low} (VGA card quality) = 0.

The degree of term “Hard disk capacity is small” takes the linguistic level “Almost Small” to the computer part “Hard disk capacity”, i.e. $\text{degree}_{\text{Small}}(\text{Hard disk capacity}) = 0.75$, $\text{degree}_{\text{Large}}(\text{Hard disk capacity}) = 0.25$, and $\text{degree}_{\text{Medium}}(\text{Hard disk capacity}) = 0.25$

Step 1: Assume that there are only two rules R1 and R2. Compare the known requirements above with the premises of rule R1 and R2. In an actual case, the system searches for all possible combinations of premises from inputs.

Step 2: Compute the degree part of rules R1, R2.

$$\begin{aligned} \text{degree}_{\text{Premise R1}} &= \text{degree}_{\text{High}}(\text{CPU level}) \wedge \\ &\quad \text{degree}_{\text{Large}}(\text{Monitor size}) \wedge \\ &\quad \text{degree}_{\text{High}}(\text{VGA card quality}) \\ &= \min \{1, 1, 1\} = 1. \end{aligned}$$

$$\begin{aligned} \text{degree}_{\text{Premise R2}} &= \mu_{\text{High}}(\text{CPU level}) \wedge \\ &\quad \mu_{\text{Small}}(\text{Hard disk capacity}) \\ &= \min \{1, 0.75\} = 0.75. \end{aligned}$$

Step3: Compute the contribution of the rules R1, R2.

$$\begin{aligned} \text{degree}_{\text{rule 1}} &= \text{degree}_{\text{premise 1}} \wedge \text{degree}(\text{Configuration is Almost High}) \\ &= \min \{1, 0.9\} = 0.9. \end{aligned}$$

$$\begin{aligned} \text{degree}_{\text{rule 2}} &= \text{degree}_{\text{premise 2}} \wedge \text{degree}(\text{Configuration is Almost High}) \\ &= \min \{0.75, 0.2\} = 0.2. \end{aligned}$$

Step 4: Compute the total weight of rules R1, R2 leading to the same conclusion:

“Configuration is Almost High”

$$\begin{aligned} \text{degree}_{\text{Total}}(\text{Configuration is Almost High}) &= \text{degree}_{\text{rule1}} \vee \text{degree}_{\text{rule2}} \\ &= 0.9 + 0.2 - (0.9 * 0.2) = 0.92 \end{aligned}$$

Finally, we get the conclusion that the Computer configuration is Almost High with the degree 0.92. Then the Type of Almost High computer configuration is given together with the corresponding price.

Knowledge Acquisition

To gain knowledge about consultations of computer configurations, I applied interview and case study. With this way, I found that

Experts divide computer parts to 3 levels high, medium and low.

Experts divide computer configuration level to 5 levels.

Experts gain information from customers through asking questions.

Some computer parts are more important than others, when experts consider computer configurations.

If experts have a great deal of information about customers' needs, the answer is more precise.

When experts consider the level of computer configurations, they mainly consider 6 parts of a computer. CPU, Memory, Hard Disk, Monitor, VGA Card, and CD-ROM.

How rules were formed.

At first, I made all rules that are derived from the 6 configurations combinations. Each configuration contains 3 levels, high, medium, and low. the derived rules are more than 500. From these rules, first, we consider a special case of rules. Such as:

IF CPU level is high and
Memory capacity is large and

HD capacity is large and

Monitor size is large and

CD-ROM quality is high and

VGA card level is high

THEN Computer configuration is high

With this example, I could say that the configuration will surely belong to high level. So we weight the rule with 1.00.

Next, I found that some rules reach the same computer configuration level even if their last part of configuration is not the same. For example,

IF CPU level is high and
Memory capacity is large and
HD capacity is large and
Monitor size is large and
CD-ROM quality is high and
VGA card level is medium

THEN Computer configuration is high

And

IF CPU level is high and
Memory capacity is large and
HD capacity is large and
Monitor size if large and
CD-ROM quality is high and
VGA card level is low

THEN Computer configuration is high

These rules belong to high with some degree but its weight will not be 1.00. These rules show that VGA card configuration is not considered to define a level of computer configurations because VGA levels are changed but reached the same computer configuration level. We try to find other rules that are in the same circumstances. This work is continued with 3 configurations and 2 configurations.

Weight for rules.

I set 6 questions to gather information about computer parts for inferring. A user should answer at least one question. I ask about these parts, CPU level, Memory capacity, Hard Disk size, Monitor size, VGA card quality, and CD-ROM quality. From the answers given by the user, the system tries to deduce the computer configuration level that will match with the customer's needs. The conclusion is divided into 5 levels, high, almost high, middle, almost low, and low. Each combination of answers is divided into 5 levels. For example,

If we have answers as

High CPU level

High memory capacity

Large HD size

Large Monitor

Medium VGA quality

Low CD-ROM quality

Then conclusion is HIGH with weight 0.86.

On the other hand, if we have answers as,

High CPU level

Low Memory capacity

Small HD size

Small Monitor

Low VGA level

Low CD-ROM level

Then conclusion is LOW with weight 0.91

Now you can see that the user gives 6 answers but reach different conclusions because the given answers are not in the same level.

It is likely that different number of answers will reach the same conclusion, but the weight will be different. For example:

If we got answers as

High CPU level

Low Memory capacity

Small HD size

Small Monitor

Low VGA level

Then conclusion is LOW with weight 0.7969

When we have more information about the problem, we can conclude with higher sureness. So we set weight for rules as below:

user gives 6 answers: range of weight for rules 1.00 – 0.80

user gives 5 answers: range of weight for rules 0.80 – 0.70

user gives 4 answers: range of weight for rules 0.70 – 0.55

user gives 3 answers: range of weight for rules 0.55 – 0.45

user gives 2 answers: range of weight for rules 0.45 – 0.20

user gives 1 answer: range of weight for rules 0.20 – 0.00

These range of weights are only as standards. There are some special cases. (For more

details please see Appendix A Rules)

Knowledge base.

Knowledge base should keep these things.

- Rules and weights
- Computer configurations

The nature of the problem, rules and weights should be changed easily because from time to time and from place to place, people's feelings change. Also, computer configurations are easily maintained. Nowadays the implementation of computer hardware is really rapid. Within three months there are new products.

These reasons let us use Database table to handle them.

I would like to show the required functions for the system.

- Add or remove easily.
- Change existed rule or computer configurations easily.
- User friendly interface
- Create new rules or computer configurations.

5.4 Testing

Knowledge base

First, we examined our knowledge base with two computer sales experts who are working at Panthip plaza. Each of them has experience as a computer sales person for about 5 years. At first, we asked them to write rules they usually considered in actual situations. They listed about 100 rules. These rules are already covered in our rules. Then we showed inputs that are listed in the rules, and then asked them to write their conclusions. After that I showed my conclusions for rules and compared with

their conclusions. They said the rules were enough and conclusions that were reached from inputs were corrects. In this testing, if one of the experts gave an answer that was the same as our knowledge base we concluded that the rule's conclusion was correct.

The System's behavior

After our knowledge base was certified, I asked them to compare the results that were concluded by the system and by the sales experts. To check the system's behavior, we tried to use the system in an actual sales situation. After 50 times of comparison, the sales expert said that the system worked well. About 90% of the conclusions were the same as that of the experts'.

Some benefits of the system.

The experts said that some benefits of the system are:

- When human experts are busy, it can still give advice.
- It reduces time for advising
- It can be used for training new employees
- It uses natural expressions for needs which is easy to understand.

For time comparison, we tried to compare the time consumed by the system and that by an expert. From asking questions to suggest were considered in case of the expert. From choosing inputs to deduce conclusion time was considered for the system. The system tested for more than 100 cases of requirements in actual situations. For each case, the test was repeated 10 times as an operation check. we took the average time as results. Figure 5-2 shows the results. The expert consumed a lot of time when he was asking questions and waiting for the answers. Also less information took the expert to deep thinking because they should deduce most of the parts.

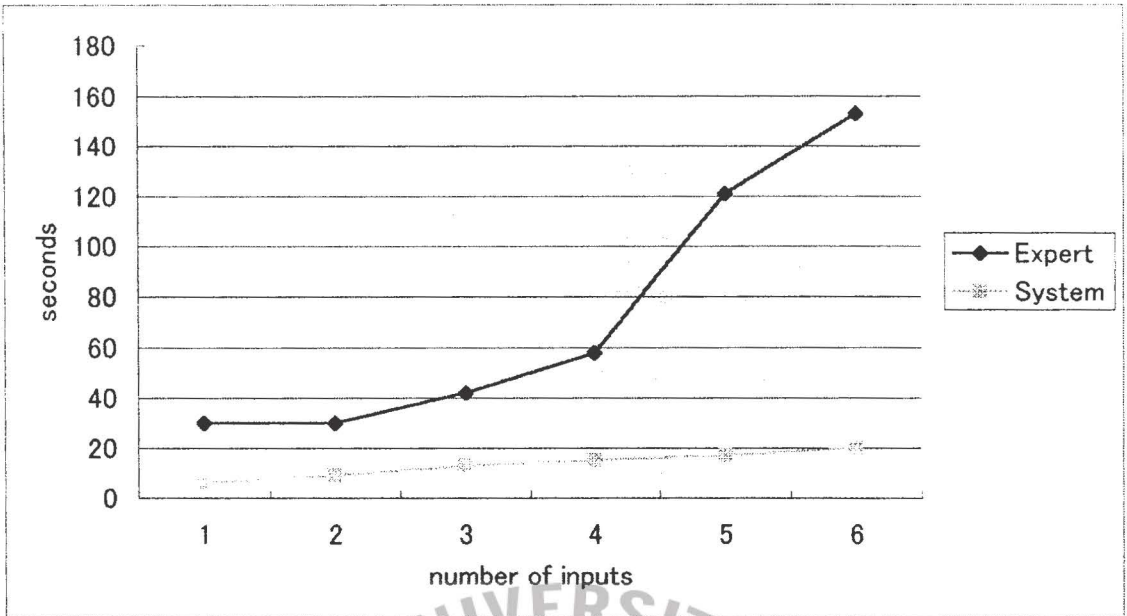


Figure 5-2: Comparing consumed time for advising by the expert and by the system



Chapter Six: IMPLEMENTATION AND PERFORMANCE OF THE SYSTEM

6.1 Programming Language [13], [28]

For this system, I chose Visual Basic 6. At first I thought about the required components for the system. Then I recognized that we mainly needed a database for the knowledge base and the interface that is easy to use. Visual Basic 6 is easy to use in making user-friendly interface. Also, this language is suitable for developing programs that work on Windows O.S., and it supports Microsoft Access file.

C and Perl were considered too, but difficulties to make a user-friendly interface were realized.

6.2 Performance of the system

a) Starting the system.

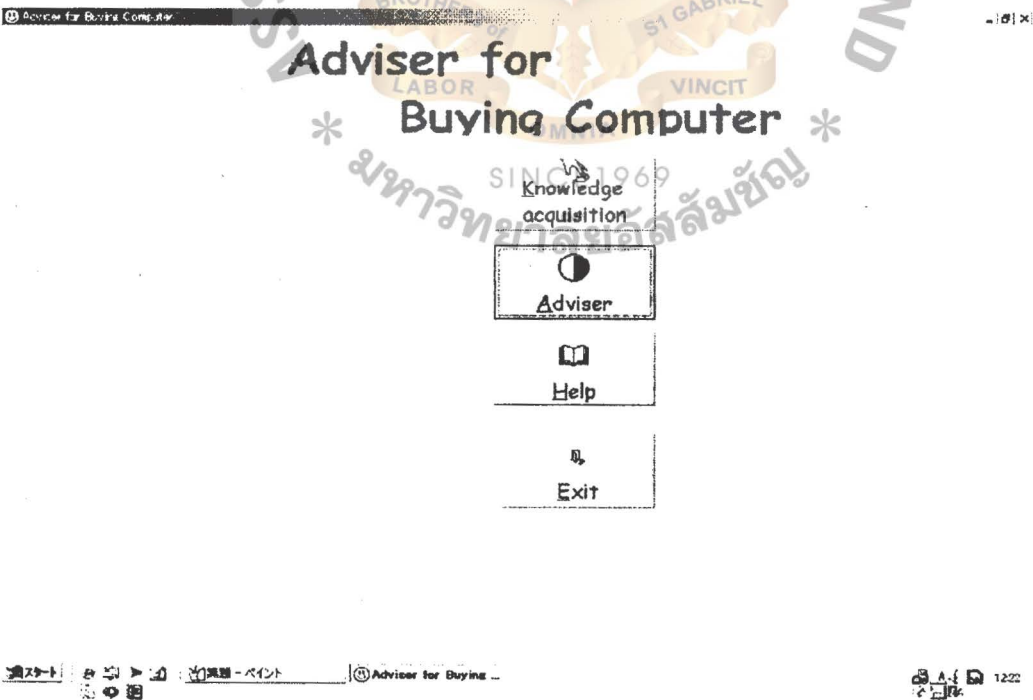


Figure 6-1: top of the system.

When starting the system interface shown in figure 6-1 appears. If you want to handle knowledge base, push “knowledge acquisition” button. If you want to start inference about computer configuration, just push “Adviser” button. Exit button for quit.

b) Knowledge acquisition.

How to edit computer units

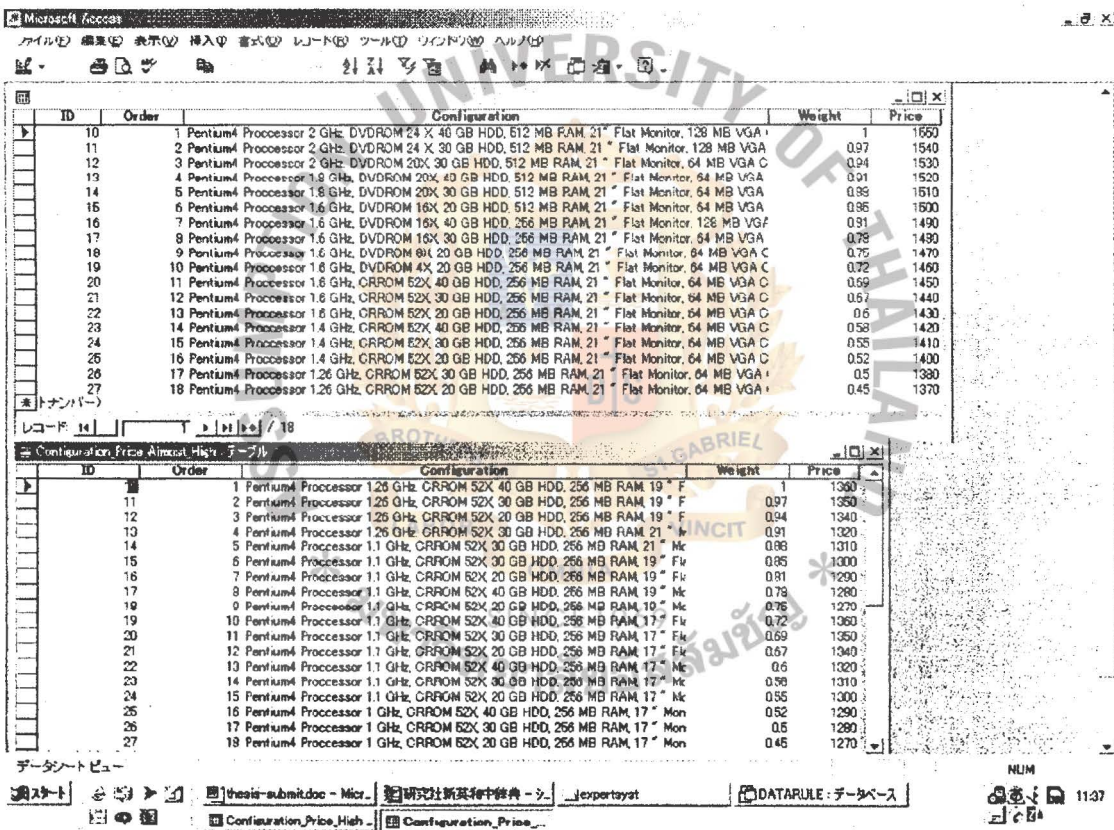


Figure 6-2: Edit computer unit using Microsoft Access.

To edit computer units, we use Microsoft Access. At first, open the database named “datarule.mdb”. Then choose the table where you want to change units.

List of table for unit were here:

“Configuration_Price_Almost_High”

- “Configuration_Price_Almost_Low”
- “Configuration_Price_High”
- “Configuration_Price_Medium”
- “Configuration_Price_Low”

You can directly change computer units in each level of configuration.

How to add rules.

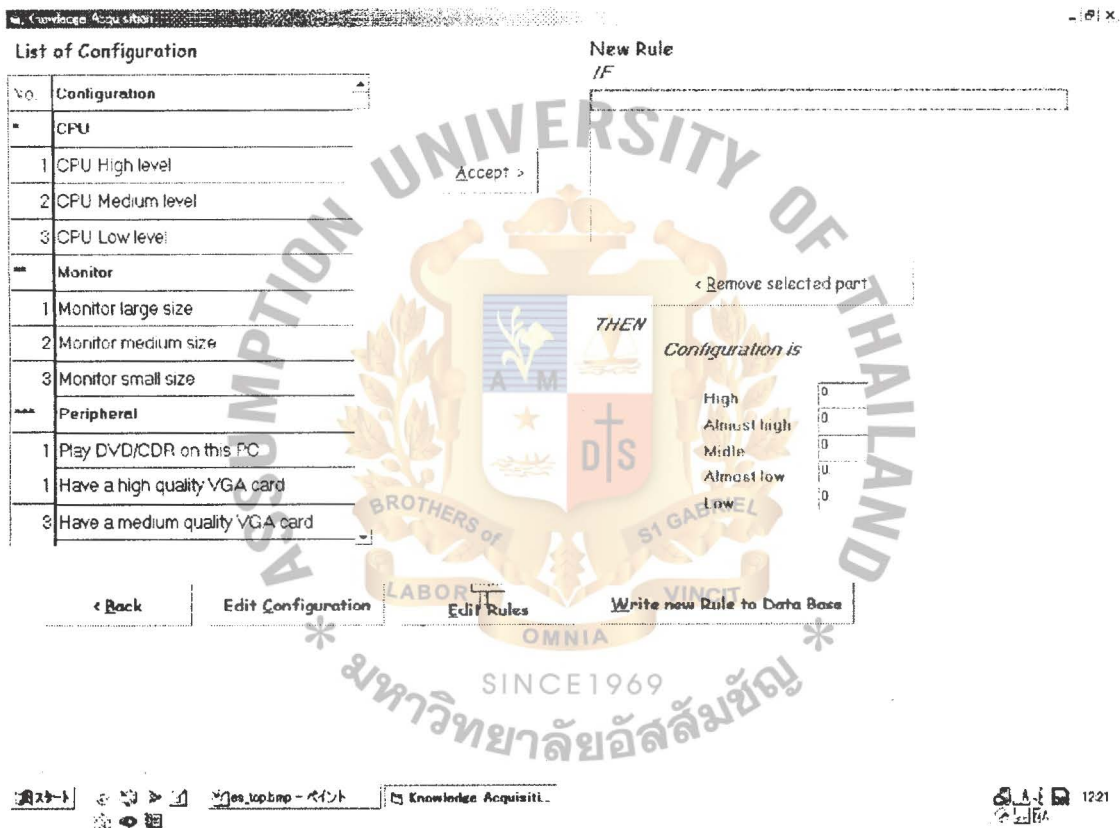


Figure 6-3: Top view of knowledge Acquisition.

After pushing “Knowledge Acquisition” button, you will reach the interface shown in figure 6-3. The left side of picture contains parts that will appear in premise. Click on a wanted part in premise and press “Accept” button at the center of the picture. The selected part will be added to the right side text box. If you want to deselect parts that you selected, click on a part you want to deselect and push

“Remove selected part”. After you chose all parts you want, then you should define THEN part by putting some degree of weight for each configuration level that appears below of IF-part text box. Then push “Write new Rule to Data Base” button to add the new rule to the knowledge base.

How to edit rules.

You can reach rule edit mode by pushing “Edit rules” button shown in figure 6-3. Then the program shows a picture as in figure 6-4.

View Rules		Weight of Configuration				
No.	Premises	High	Almost High	Medium	Almost low	Low
1	[CPU High level] AND [Monitor large size] AND [CD-ROM high quality] AND [high quality VGA card] AND [Large hard disk] AND [High memory capacity]	1	0	0	0	0
2	[CPU High level] AND [Monitor medium size] AND [CD-ROM high quality] AND [high quality VGA card] AND [Large hard disk] AND [High memory capacity]	0.925	0	0	0	0
3	[CPU High level] AND [High memory capacity] AND [Large hard disk] AND [Monitor medium size] AND [high quality VGA card] AND [CD-ROM medium level]	0.9	0	0	0	0
4	[CPU High level] AND [High memory capacity] AND [Large hard disk] AND [Monitor medium size] AND [medium quality VGA card] AND [CD-ROM high quality]	0.9	0	0	0	0
5	[CPU High level] AND [High memory capacity] AND [Medium size hard disk] AND [Monitor large size] AND [CD-ROM high quality] AND [high quality VGA card]	0.9	0	0	0	0
6	[CPU High level] AND [High memory capacity] AND [Large hard disk] AND [Monitor small size] AND [medium quality VGA card] AND [CD-ROM low quality]	0	0	0.91	0	0
7	[CPU High level] AND [High memory capacity] AND [Medium size hard disk] AND [Monitor small size] AND [low quality VGA card] AND [CD-ROM high quality]	0	1	0	0	0

< Back

Edit Rule

Remove Rule

Refresh

Figure 6-4: List of rules in the knowledge base.

This picture contains the list of rules in the knowledge base. The top part of the picture contains the premise of rules and weights for each configuration. The bottom part of the picture contains command buttons named “Edit rules”, “Remove Rules” and “Refresh”. Each button gives a command to the program as below:

Edit Rule button: push this button to edit selected rule.

Remove Rule button: push this button to remove selected rule.

Refresh button: push this button to refresh rule list.

To edit a rule in the list, you should select a rule that you want to edit in the list, then the program shows a dialog box as in figure 6-5. You can change the premise and weights for each level. When you want to write edited rules, push the write button on the dialog box, or, cancel button for cancellation of edit.

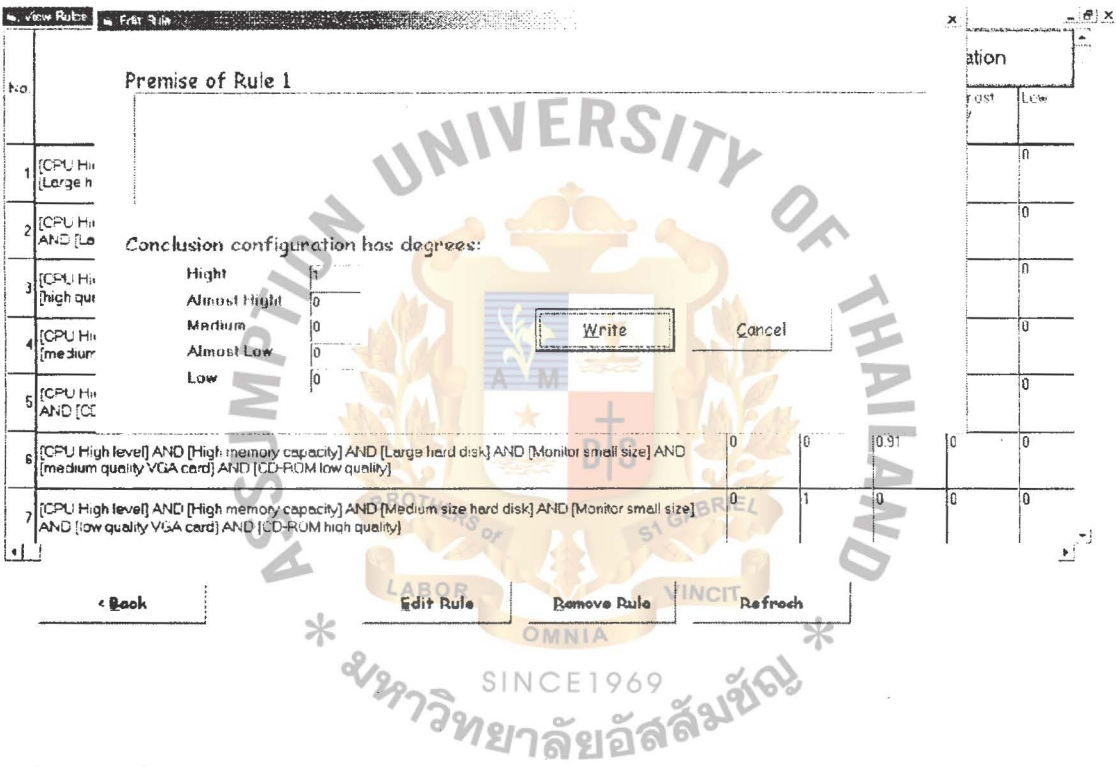


Figure 6-5: Dialog box for rule edit.

How to delete rules

To delete rules, choose a rule you want to delete by clicking on the rule list in figure 6-4, then push “Remove rule” button. The program asks you for confirmation of deleting the rule you selected as in figure 6-6. If you are sure press “yes” to delete the rule. If you don’t want to delete the rule, press no button.

No.	Premises	Weight of Configuration				
		High	Almost High	Medium	Almost low	Low
1	[CPU High level] AND [Monitor large size] AND [CD-ROM high quality] AND [high quality VGA card] AND [Large hard disk] AND [High memory capacity]	1	0	0	0	0
2	[CPU High level] AND [Monitor medium size] AND [CD-ROM high quality] AND [high quality VGA card] AND [Large hard disk] AND [High memory capacity]	0.925	0	0	0	0
3	[CPU High level] AND [High memory capacity] AND [Large hard disk] AND [Monitor medium size] AND [high quality VGA card] AND [CD-ROM medium level]	0.9	0	0	0	0
4	[CPU High level] AND [High memory capacity] AND [Large hard disk] AND [Monitor medium size] AND [medium quality VGA card] AND [CD-ROM high quality]	0.9	0	0	0	0
5	[CPU High level] AND [High memory capacity] AND [Medium size hard disk] AND [CD-ROM high quality] AND [high quality VGA card]	0.9	0	0	0	0
6	[CPU High level] AND [High memory capacity] AND [Large hard disk] AND [medium quality VGA card] AND [CD-ROM low quality]	0	0	0.91	0	0
7	[CPU High level] AND [High memory capacity] AND [Medium size hard disk] AND [Monitor small size] AND [low quality VGA card] AND [CD-ROM high quality]	0	1	0	0	0

Back
Edit Rule
Remove Rule
Refresh

Figure 6-6: Confirmation for deleting rule.

How to edit configurations

In this program we applied weight for each part of premise and these parts are also editable. When you want to edit weight, push “Edit configuration” button on figure 6-3. Then the program shows a picture as in figure 6-7. Each command button corresponds to the operations as below:

Edit button: to edit selected part.

Remove button: to remove selected part from the knowledge base.

Refresh button: to refresh list of parts

Back button: move to knowledge acquisition picture.

The right side of picture shows configurations and related type of expression and weights. When you want to edit, select configuration by clicking on the list. Then the selected configurations details will appear on the left side. The left side shows

description of part, type of the part and group of the part to which they belong. After editing, you should choose the group of parts “write” button.

Update configuration

New Configuration

Description of part

Type of the part

Type

Degree of type

Group of Parts

☐ CPU

☐ Peripheral

☐ Monitor

☐ Hard disk and memory capacities

Write

Back

Edit

Remove

Refresh

Configuration on the knowledge base

No	Configuration	Type 1	Degree 1	Type 2	Degree 2	Type 3
-	CPU					
1	CPU High level	sure high		1 Almost high	0.75	middle
2	CPU Medium level	sure medium		1 almost med	0.75	middle
3	CPU Low level	sure low		1 almost low	0.75	middle
-	Monitor					
1	Monitor large size	sure large		1 almost large	0.75	middle
2	Monitor medium size	sure medium		1 almost med	0.75	middle
3	Monitor small size	sure small		1 almost sma	0.75	middle
-	Peripheral					
1	CD-ROM high quality	sure high		1 almost high	0.75	middle
2	CD-ROM medium low	sure medium		1 almost med	0.75	middle
3	CD-ROM low quality	sure low		1 almost low	0.75	middle
4	high quality VGA card	sure high		1 almost high	0.75	middle

Figure 6-7: Configuration view.

How to remove configuration part

Just select from the list on the right hand side by clicking and pushing remove button on figure 6-7. Then the program asks for confirmation of deleting configuration as in figure 6-8.

How to add new configuration part

Fill out Description part, type of the part and its degree, and group of parts. Then push write button. They are automatically added to knowledge base.

Update Configuration

New Configuration

Description of part

Type of the part

Type

Degree of type

Group of Parts

☐ CPU
☐ Peripheral

☐ Monitor
☐ Hard disk and memory capacities

< Back

Edit

Remove

Refresh

Configuration on the knowledge base

No	Configuration	Type 1	Degree 1	Type 2	Degree 2	Type
▲	CPU					
1	CPU High level	sure high	1	Almost high	0.75	middle
2	CPU Medium level	sure medium	1	almost med	0.75	middle
3	CPU Low level	sure low	1	almost low	0.75	middle
▲	Monitor					
1	Monitor large size	sure large	1	almost large	0.75	middle
2	Monitor medium size	sure medium	1	almost med	0.75	middle
	are small		1	almost sma	0.75	middle
	are high		1	almost high	0.75	middle
2	CD-ROM medium lev	sure medium	1	almost med	0.75	middle
3	CD-ROM low quality	sure low	1	almost low	0.75	middle
4	high quality VGA card	sure high	1	almost high	0.75	middle

Delete Symptom 1: 'CPU High level' ?

Write

Figure: 6-8: Confirmation for deleting selected configuration part.

c) Inferring advice

In the inference interface, these are contained as in figure 6-9. The details for each part are:

1. Questions about user's requirements. Questions are about CPU, Memory, Hard Disk, Monitor, CD-ROM and VGA cards.
2. Ranges: Show ranges of each level of requirements.
3. Accept button: Add the selected requirement to the list of requirements.
4. Deselect button: Remove the selected-requirement from the list of requirements.
5. Inference button: Start the inference based on current requirements.
6. List of requirements. Selected requirements are stored here.
7. Results are shown here.

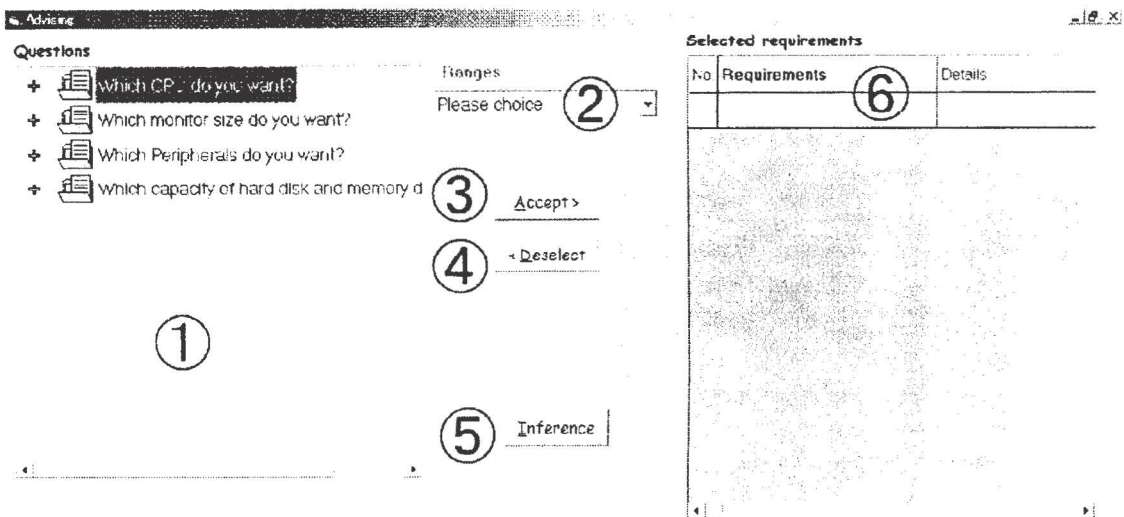


Figure 6-9: Inference interface

How to select inputs (requirements)

To select your requirements for computer configuration, first you should double click a question or click + marks on the left of the questions (see figure 6-10). Second, choose your requirement for the question and select the range of level. Then push "Accept" button to add your requirements to the list of requirements. Selected requirements are shown as in figure 6-11. Here we tried to search for very high level CPU, very large monitor, very high quality CD-ROM, very high quality VGA card, very large hard disk, and very high capacity memory.

Questions

Which Peripherals do you want?

- ☐ Play DVD/CD/R on this PC
- ☐ Have a high quality VGA card
- ☐ Have a medium quality VGA card
- ☐ Have a low quality VGA card

Which capacity of hard disk and memory

- ☐ Large hard disk
- ☐ Medium size hard disk
- ☐ Small size hard disk
- ☒ High memory capacity
- ☐ Medium memory capacity

Rangees

Please choice

Accept

Deselect

Inference

Selected requirements

No.	Requirements	Details
1	CPU High level	Very high
2	Monitor large size	Very large
3	Play DVD/CD/R on this PC	Very high quality
4	Have a high quality VGA ca	Very high quality
5	Large hard disk	Very large
6	High memory capacity	Very high

53

Outputs

Outputs contain:

- 1. Recommended configuration’s level
- 2. Computer configuration for recommended configuration level
- 3. Price for computer configuration that is recommended.
- 4. “Yes” and “No” button: Are you satisfied with shown price or not?

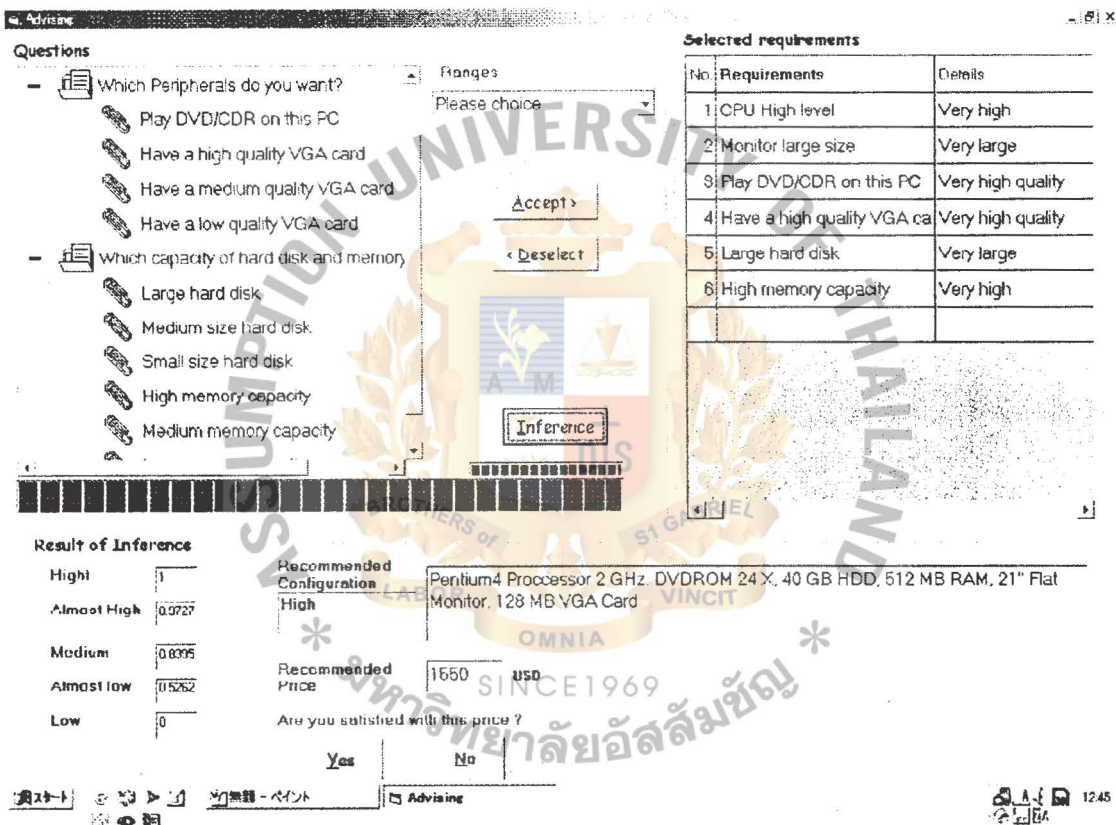


Figure 6-12: Result for requested configuration.

In this example, we get results for requests as High-level configuration and the price of US\$1550. The computer configuration is “Pentium4 processor 2Ghz, DVD-ROM 24x, 40GB of Hard Disk, 512MB of RAM, 21’ Flat monitor, 128MB VGA card.”

If you are satisfied with the results, push yes to finish. If you are not satisfied,

press no to re-inference with new requests.

Changing inputs

When you press “No”, a text box appears on the bottom to fill your budget. Please see figure 6-13.

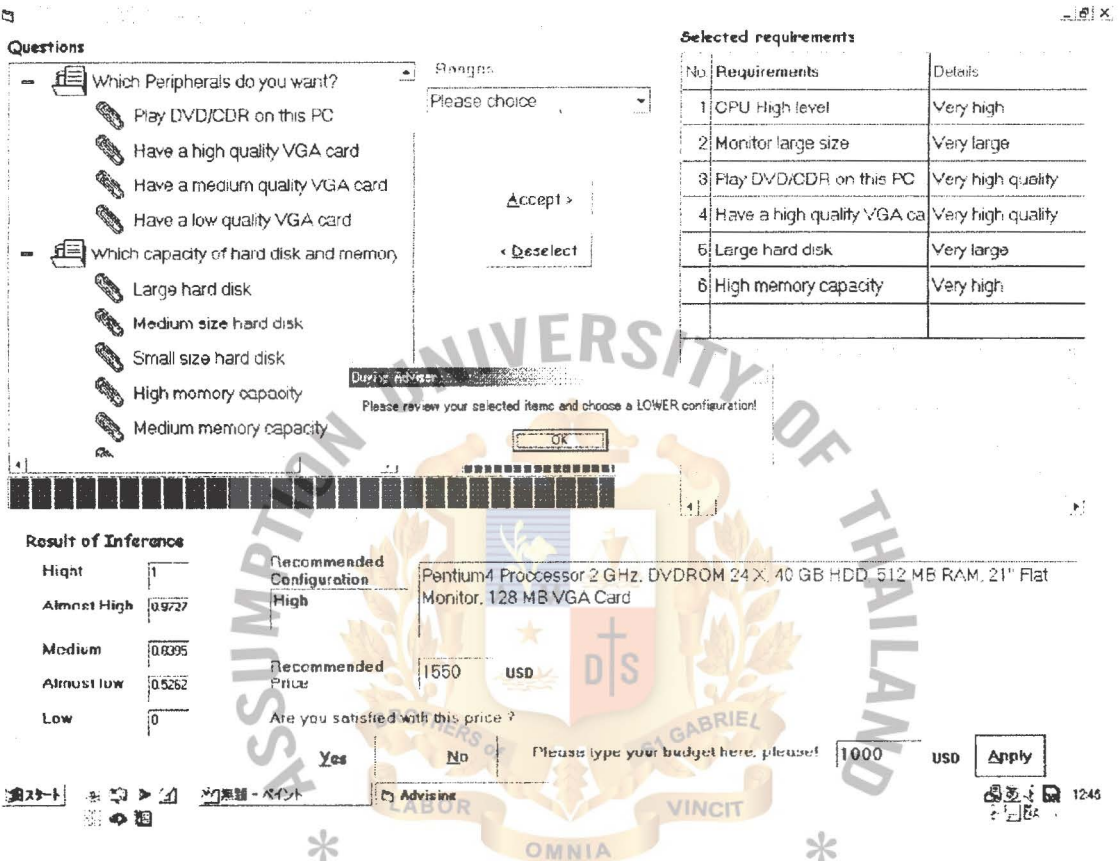


Figure 6-13: Did not satisfy with reached result.

For example, you put US\$1000 for your budget and then press “apply” button. The system advises, “Please review your selected items and choose a LOWER configuration”.

So deselect a part you want to change, and select a new requirement for that part. For example, we choose “very high CPU, above medium size monitor, medium quality of CD-ROM, Medium quality of VGA, very large hard disk and high capacity memory” as new requirements. We get the result as requested configuration belonging to “Almost high configuration” and cost is US\$1360. The computer configuration is

“Pentium4 processor 1.26Ghz, 52x CD-ROM, 40GB hard disk, 256MB memory, 19’ flat monitor and 64MB VGA card”. See figure 6-14.

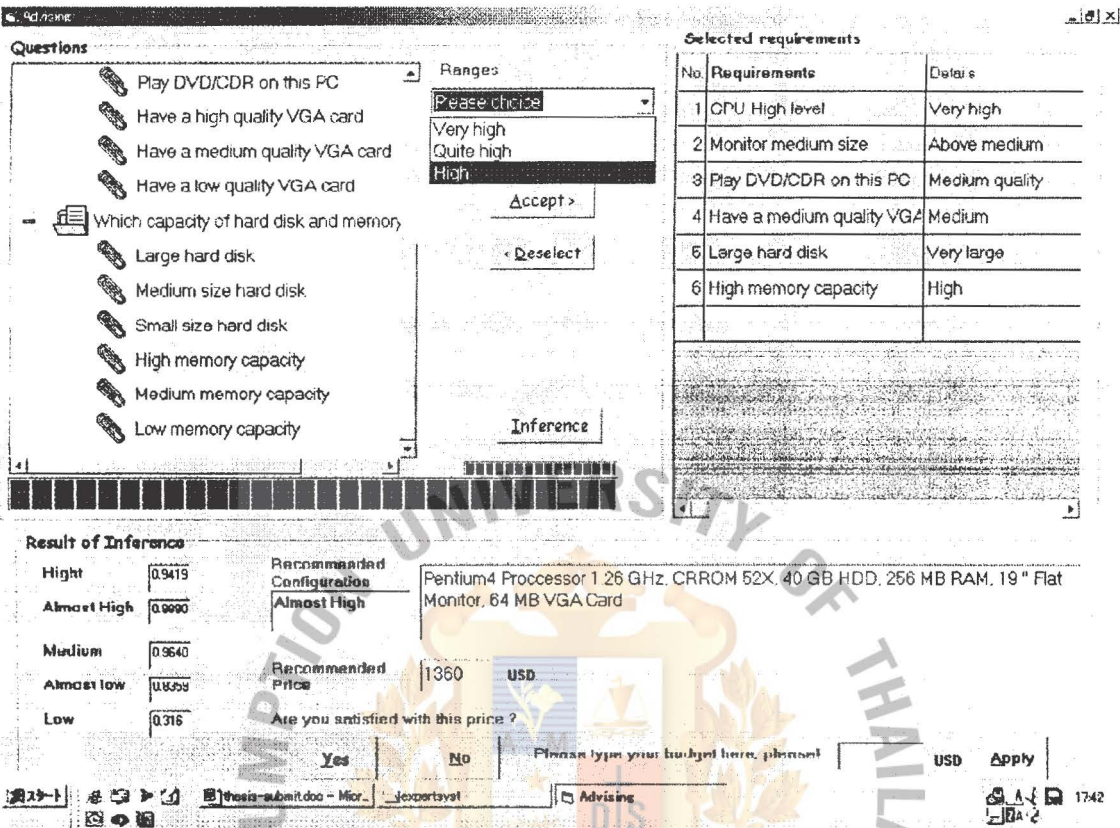


Figure 6-14: New result with lower requests configuration.

Chapter Seven: CONCLUSION

7.1 Summary

In this thesis, I have:

- 1 Studied the problem of business and computer configurations consultation.
- 2 Understood the nature of problems of computer configuration consultations in reasoning with ambiguity concepts.
- 3 Studied expert systems technology and some existing systems such as XCON/XSEL, APEX, and so on. These systems, which were in the same problem, were applied to real situations.
- 4 Developed the rule-based system with a degree of belief model for computer configuration consultation problems.
- 5 Established the rule base for the system prototype. It contains more than 300 rules, using an Inference Engine with forward chaining.
- 6 Implemented the system in the computers.

7.2 Future works

Future researchers should focus on the following:

- Testing the system with other experts to improve knowledge base and inference strategy.
- Developing the explanations module to show HOW the configuration advice is obtained, tracing all rules need in the inference process of the system.
- Establishing an explanation module for the system.

References

- [1] Zadeh L.A. Fuzzy Sets, Information and Control, Vol. 8, 338-353.
- [2] Waterman, D.A., A Guide to Expert Systems, Addison-Wesley, 1986.
- [3] Kouichirou Akita Guide of Introduction to Expert System (Japanese), Denki Shoin, 1988.
- [4] Hans Bandemer, and Siegfried Gottwald. Fuzzy Sets, Fuzzy Logic, Fuzzy Methods – with Application, John Wiley and Sons, 1995.
- [5] Annabel Beerel, Expert Systems in Business: Real World Applications, Ellis Horwood, 1993
- [6] M. A. Bramer. Research and Development in Expert System (Japanese), ED, Cambridge University Press, 1987..
- [7] Megginson Byrd, and Scott Megginson Small Business Management – An Entrepreneur's Guide to Success, IRWIN, 1994.
- [8] San-Chyi Chang. "Optimal revenue for fuzzy demand quantity" FUZZY – sets and systems, Volume 111, Number 3, May 1, 2000 IFSA. North-Holland, pp399 – 464.
- [9] C. H. Chen, Fuzzy Logic and Neural Network Handbook, McGraw-Hill, 1996
- [10] John Durkin Expert systems – design and development, Macmillan Publishing Company, 1994.
- [11] Feigenbaum, E., et al. The Rise of the Expert Company, Time-life, 1988
- [12] Paul Harmon and David King. Expert Systems – Information & Computing (Japanese), Science Publishing, 1986.
- [13] Kyoko Iketani, and Tomoaki Masuda 500 Techniques for Visual Basic (Japanese), Shuwa System. 2001
- [14] J. -S. Jang, C.T. Sun, E. Mizutani, Neuro-Fuzzy and Soft Computing,

Prentice-Hall, Inc., 1997.

- [15] Abraham Kandel. Fuzzy Expert Systems, CRC, 1992
- [16] George J. Klir and Tina A. Folger. Fuzzy Sets, Uncertainty, and Information, Prentice-Hall International Edition, 1988..
- [17] George J. Klir and Bo Yuan Fuzzy Sets And Fuzzy Logic – Theory and Applications, Prentice-Hall International Edition, 1995.
- [18] George J. Klir., Ute H. St. Clair., Bo Yuan Fuzzy Set Theory – Foundations and Applications, Prentice Hall International Edition, 1997.
- [19] Philip Kptler., Marketing Management –analysis, planning, implementation and control., Prentice-Hall International Edition, 1994.
- [20] Jay Liebowitz, Expert Systems for Business & Management, ED, Printice-Hall, 1990.
- [21] F. Martin McNeill and Ellen Thro. Fuzzy logic – a practical approach, AP Professional, 1994.
- [22] Witold Pedrycz and Fernando Gomide, An Introduction to Fuzzy Sets: Analysis and Design, Massachusetts Institute of Technology, 1998
- [23] D.V. Pigford, and Greg Baur, Expert Systems for Business: Concepts and Applications, Boy & fraser, 1995.
- [24] Dr. William M. Siler Theory and Practice of Fuzzy Expert System (Japanese), Denki Shoin, 1990.
- [25] Barry G. Silverman. Expert systems for business, ED, Addison Wesley, 1987.
- [26] Jun-ichi Touchi. Introduction to Expert System (Japanese), Nihon Rikou Shuppankai, 1997.
- [27] Tuthill and Levy. Knowledge – Based Systems – A Manager’s Perspective, TPR, 1991.
- [28] Wallace Wang. Visual Basic 6 For Dummies, Hungry Minds, 1998

APPENDIX A



NO	IF PART						THEN	MF
	CPU	Memory	HD	Monitor	VGA	CD-ROM		
1	high	high	large	large	high	high	high	1
2	high	high	large	medium	high	high	high	0.925
3	high	high	large	medium	high	medium	high	0.9
4	high	high	large	medium	medium	high	high	0.9
5	high	high	medium	large	high	high	high	0.9
6	high	high	medium	small	medium	low	medium	0.91
7	high	high	medium	small	low	high	almost high	1
8	high	high	small	medium	medium	high	almost high	0.95
9	high	high	small	medium	low	high	almost high	0.97
10	high	medium	large	small	high	low	medium	0.92
11	high	medium	large	small	medium	high	almost high	0.95
12	high	medium	medium	large	low	low	medium	0.95
13	high	medium	medium	medium	high	high	almost high	0.8
14	high	low	large	large	low	low	medium	0.8
15	high	low	large	medium	high	high	almost high	0.83
16	high	low	small	small	medium	low	almost low	0.82
17	high	low	small	small	low	high	medium	0.82
18	medium	high	large	medium	low	low	medium	0.8
19	medium	high	large	small	high	high	almost high	0.99
20	medium	high	medium	large	medium	low	medium	0.82
21	medium	high	medium	large	low	high	almost high	0.83
22	medium	medium	large	large	high	low	medium	0.89
23	medium	medium	large	large	medium	high	almost high	0.83
24	medium	medium	medium	medium	medium	medium	medium	1
25	medium	medium	small	medium	low	low	almost low	0.9
26	medium	medium	small	small	high	high	medium	0.81
27	medium	low	medium	medium	medium	low	almost low	0.95
28	medium	low	medium	medium	low	high	medium	0.79
29	medium	low	small	large	medium	high	medium	0.82
30	medium	low	small	large	low	high	medium	0.8
31	low	high	large	large	medium	high	almost high	0.93
32	low	high	small	medium	low	low	almost low	0.9
33	low	high	small	small	high	high	medium	0.8
34	low	medium	large	small	low	low	almost low	1
35	low	medium	medium	medium	medium	high	medium	0.85
36	low	medium	small	large	high	medium	medium	0.84

NO	IF PART						THEN	MF
	CPU	Memory	HD	Monitor	VGA	CD-ROM		
37	low	medium	small	large	medium	high	medium	0.85
38	low	low	large	medium	high	low	almost low	0.86
39	low	low	large	medium	medium	high	medium	0.83
40	low	low	medium	large	high	high	medium	0.81
41	low	low	medium	small	low	low	low	0.9
42	low	low	small	medium	medium	low	low	0.92
43	low	low	small	medium	low	high	almost low	0.84
44	low	low	small	small	low	low	low	1
45	high	high	large	large	high		high	0.8
46	high	high	medium	small	low		almost high	0.78
47	high	high	small	medium	high		almost high	0.8
48	high	medium	large	small	high		almost high	0.76
49	high	low	small	small	low		almost low	0.75
50	medium	high	medium	large	high		almost high	0.79
51	medium	high	medium	large	medium		almost high	0.74
52	medium	medium	medium	medium	medium		medium	0.8
53	medium	medium	large	large	high		almost high	0.73
54	medium	low	medium	medium	low		almost low	0.7
55	medium	low	small	large	high		medium	0.7
56	low	high	large	large	high		almost high	0.7
57	low	medium	medium	medium	high		medium	0.75
58	low	low	small	small	low		low	0.8
59	low	low	large	medium	high		medium	0.72
60	high	high	small	large			almost high	0.59
61	high	medium	large	small			medium	0.55
62	high	medium	medium	large			almost high	0.6
63	high	low	large	large			almost high	0.64
64	medium	high	large	small			medium	0.6
65	medium	medium	small	small			almost low	0.57
66	medium	low	medium	small			almost low	0.55
67	low	high	small	small			almost low	0.58
68	low	medium	medium	large			medium	0.55
69	low	low	large	large			medium	0.55
70	high	high	large	large			high	0.69
71	high	high	large	medium			high	0.67
72	high	high	large	small			almost high	0.69

NO	IF PART						THEN	MF
	CPU	Memory	HD	Monitor	VGA	CD-ROM		
73	high	high	medium	large			almost high	0.67
74	high	high	medium	medium			almost high	0.65
75	high	high	medium	small			medium	0.6
76	high	high	small	large			almost high	0.67
77	high	high	small	medium			almost high	0.68
78	high	high	small		medium		medium	0.59
79	high	high	small	small			medium	0.58
80	high	medium	large	large			almost high	0.65
81	high	medium	large	medium			almost high	0.66
82	high	medium	medium		high	high	almost high	0.75
83	high	medium	medium	medium			medium	0.59
84	high	medium	medium	small			medium	0.58
85	high	medium	small	large			medium	0.6
86	high	medium	small	medium			medium	0.61
87	high	medium	small	small			medium	0.63
88	high	low	large		high		almost high	0.57
89	high	low	large	medium			medium	0.63
90	high	low	large	small			medium	0.62
91	high	low	large	large			medium	0.64
92	high	low	medium	medium			medium	0.58
93	high	low	medium	small			medium	0.57
94	high	low	small	large			medium	0.55
95	high	low	small	medium			medium	0.56
96	high	low	small	small			almost low	0.67
97	medium	high	large	large			almost high	0.56
98	medium	high	large	medium			almost high	0.57
99	medium	high	medium	large			almost high	0.59
100	medium	high	medium	medium			medium	0.61
101	medium	high	medium	small			medium	0.62
102	medium	high	small	large			medium	0.63
103	medium	high	small	medium			medium	0.61
104	medium	high	small	small			medium	0.64
105	medium	medium	large	large			medium	0.63
106	medium	medium	large	medium			medium	0.65
107	medium	medium	large	small			medium	0.63
108	medium	medium	medium	large			medium	0.67

NO	IF PART						THEN	MF
	CPU	Memory	HD	Monitor	VGA	CD-ROM		
109	medium	medium	medium	medium			medium	0.69
110	medium	medium	medium	small			medium	0.67
111	medium	medium	small	large			medium	0.65
112	medium	medium	small	medium			medium	0.66
113	medium	low	large	large			medium	0.67
114	medium	low	large	medium			medium	0.64
115	medium	low	large	small			medium	0.63
116	medium	low	medium	large			medium	0.61
117	medium	low	medium	medium			medium	0.62
118	medium	low	small	large			almost low	0.56
119	medium	low	small	medium			almost low	0.57
120	medium	low	small	small			almost low	0.56
121	low	high	large	large			medium	0.59
122	low	high	large	medium			medium	0.58
123	low	high	large	small			medium	0.56
124	low	high	medium	large			medium	0.55
125	low	high	medium	medium			medium	0.57
126	low	high	medium	small			medium	0.58
127	low	high	small	large			medium	0.56
128	low	high	small	medium			medium	0.56
129	low	medium	large	large			medium	0.56
130	low	medium	large	medium			medium	0.58
131	low	medium	large	small			medium	0.54
132	low	medium	medium	medium			almost low	0.59
133	low	medium	medium	small			almost low	0.6
134	low	medium	small	large			almost low	0.67
135	low	medium	small	medium			almost low	0.69
136	low	medium	small	small			almost low	0.68
137	low	low	large	medium			almost low	0.67
138	low	low	large	small			almost low	0.65
139	low	low	medium	large			almost low	0.67
140	low	low	medium	medium			almost low	0.62
141	low	low	medium	small			almost low	0.63
142	low	low	small	large			almost low	0.64
143	low	low	small	medium			almost low	0.65
144	low	low	small	small			low	0.69

NO	IF PART						THEN	MF
	CPU	Memory	HD	Monitor	VGA	CD-ROM		
145	high	high		large			high	0.53
146	high	high		medium			almost high	0.53
147	high	high		small			medium	0.51
148	high	medium		large			almost high	0.52
149	high	medium		medium			medium	0.47
150	high	medium		small			medium	0.48
151	high	low		large			medium	0.47
152	high	low		medium			medium	0.48
153	high	low		small			medium	0.49
154	medium	high		large			almost high	0.51
155	medium	high		medium			medium	0.5
156	medium	high		small			medium	0.51
157	medium	medium		large			medium	0.52
158	medium	medium		medium			medium	0.53
159	medium	medium		small			medium	0.52
160	medium	low		large			almost low	0.49
161	medium	low		medium			almost low	0.47
162	medium	low		small			almost low	0.48
163	low	high		large			medium	0.46
164	low	high		medium			medium	0.47
165	low	high		small			medium	0.46
166	low	medium		large			medium	0.48
167	low	medium		medium			almost low	0.49
168	low	medium		small			almost low	0.51
169	low	low		large			almost low	0.53
170	low	low		medium			low	0.52
171	low	low		small			low	0.53
172	high		large		high		high	0.51
173	high		large		medium		almost high	0.51
174	high		large		low		almost high	0.5
175	high		medium		high		high	0.5
176	high		medium		medium		almost high	0.49
177	high		medium		low		medium	0.46
178	high		small		high		medium	0.46
179	high		small		medium		medium	0.47
180	high		small		low		medium	0.48

NO	IF PART						THEN	MF
	CPU	Memory	HD	Monitor	VGA	CD-ROM		
181	medium		large		high		almost high	0.49
182	medium		large		medium		almost high	0.48
183	medium		large		low		almost high	0.47
184	medium		medium		high		almost high	0.48
185	medium		medium		medium		medium	0.51
186	medium		medium		low		medium	0.49
187	medium		small		high		medium	0.48
188	medium		small		medium		medium	0.47
189	medium		small		low		almost low	0.47
190	low		large		high		medium	0.47
191	low		large		medium		medium	0.46
192	low		large		low		medium	0.48
193	low		medium		high		medium	0.46
194	low		medium		medium		medium	0.47
195	low		medium		low		medium	0.46
196	low		small		high		almost low	0.51
197	low		small		medium		low	0.5
198	low		small		low		low	0.51
199	high	high	large				high	0.54
200	high	high	medium				almost high	0.54
201	high	high	small				medium	0.45
202	high	medium	large				almost high	0.53
203	high	medium	medium				medium	0.44
204	high	medium	small				medium	0.43
205	high	low	large				medium	0.47
206	high	low	medium				medium	0.45
207	high	low	small				almost low	0.44
208	medium	high	large				almost high	0.48
209	medium	high	medium				medium	0.49
210	medium	high	small				medium	0.48
211	medium	medium	large				medium	0.51
212	medium	medium	medium				medium	0.54
213	medium	medium	small				medium	0.51
214	medium	low	large				medium	0.49
215	medium	low	medium				almost low	0.49
216	medium	low	small				almost low	0.48

NO	IF PART						THEN	MF
	CPU	Memory	HD	Monitor	VGA	CD-ROM		
217	low	high	large				almost high	0.49
218	low	high	medium				medium	0.44
219	low	high	small				almost low	0.51
220	low	medium	large				medium	0.46
221	low	medium	medium				almost low	0.53
222	low	medium	small				almost low	0.54
223	low	low	large				almost low	0.53
224	low	low	medium				low	0.51
225	low	low	small				low	0.54
226	high						medium	0.21
227	medium						almost low	0.21
228	low						low	0.21
229		high					medium	0.19
230		medium					almost low	0.2
231		low					almost low	0.15
232			large				medium	0.17
233			medium				almost low	0.17
234			small				low	0.19
235				large			medium	0.16
236				medium			almost low	0.15
237				small			almost low	0.16
238					high		medium	0.14
239					medium		almost low	0.14
240					low		almost low	0.15
241						high	medium	0.13
242						medium	almost low	0.14
243						low	almost low	0.16
244	high	high					almost high	0.3
245	high	medium					medium	0.3
246	high	low					medium	0.31
247	high		large				almost high	0.29
248	high		medium				medium	0.3
249	high		small				medium	0.34
250	high			large			almost high	0.3
251	high			medium			medium	0.32
252	high			small			medium	0.33

NO	IF PART						THEN	MF
	CPU	Memory	HD	Monitor	VGA	CD-ROM		
253	medium	high					medium	0.35
254	medium	medium					medium	0.31
255	medium	low					almost low	0.37
256	medium		large				medium	0.33
257	medium		medium				medium	0.38
258	medium		small				almost low	0.3
259	medium			large			medium	0.37
260	medium			medium			medium	0.35
261	medium			small			almost low	0.28
262	medium				high		medium	0.22
263	medium				medium		medium	0.21
264	medium				low		medium	0.23
265	medium					high	medium	0.3
266	medium					medium	medium	0.26
267	medium					low	medium	0.23
268	low	high					medium	0.34
269	low	medium					almost low	0.29
270	low	low					almost low	0.3
271	low		large				medium	0.34
272	low		medium				almost low	0.2
273	low		small				almost low	0.25
274	low			large			medium	0.29
275	low			medium			almost low	0.23
276	low			small			almost low	0.21
277	low				high		medium	0.21
278	low				medium		almost low	0.23
279	low				low		almost low	0.25
280	low					high	medium	0.2
281	low					medium	almost low	0.25
282	low					low	almost low	0.2
283		high	large				almost high	0.31
284		high	medium				medium	0.34
285		high	small				medium	0.29
286		high		large			almost high	0.3
287		high		medium			medium	0.33
288		high		small			medium	0.37

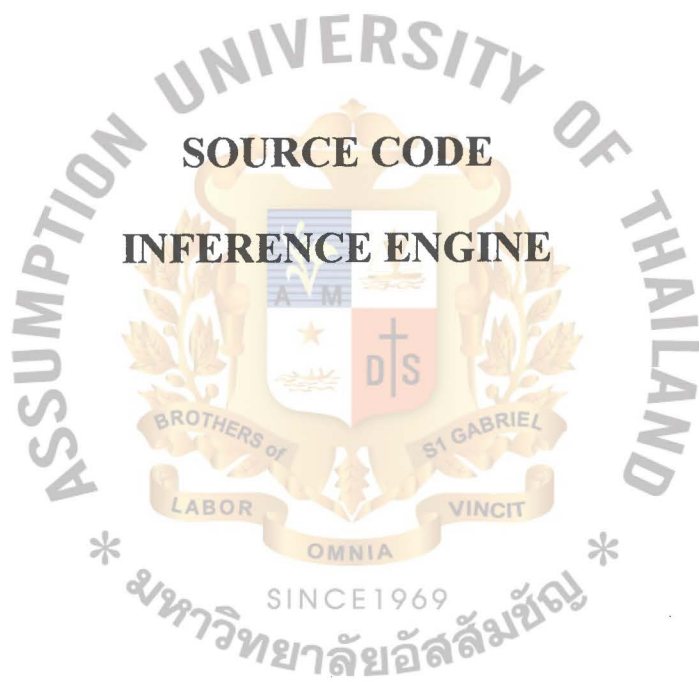
NO	IF PART						THEN	MF
	CPU	Memory	HD	Monitor	VGA	CD-ROM		
289		high			high		almost high	0.34
290		high			medium		medium	0.33
291		high			low		medium	0.35
292		high				high	almost high	0.28
293		high				medium	medium	0.3
294		high				low	medium	0.33
295		medium	large				medium	0.34
296		medium	medium				medium	0.35
297		medium	small				almost low	0.3
298		medium		large			medium	0.29
299		medium		medium			medium	0.34
300		medium		small			almost low	0.31
301		medium			high		medium	0.3
302		medium			medium		medium	0.3
303		medium			low		medium	0.3
304		low	large				medium	0.3
305		low	medium				almost low	0.34
306		low	small				almost low	0.32
307		low		large			medium	0.29
308		low		medium			almost low	0.24
309		low		small			almost low	0.21
310		low			high		medium	0.27
311		low			medium		almost low	0.21
312		low			low		almost low	0.22
313			large	large			almost high	0.33
314			large	medium			medium	0.32
315			large	small			medium	0.33
316			large		high		almost high	0.32
317			large		medium		medium	0.34
318			large		low		medium	0.35
319			medium	large			medium	0.34
320			medium	medium			medium	0.35
321			medium	small			medium	0.33
322			medium		high		medium	0.39
323			medium		medium		medium	0.32
324			medium		low		medium	0.34

NO	IF PART						THEN	MF
	CPU	Memory	HD	Monitor	VGA	CD-ROM		
325			small	large			medium	0.33
326			small	medium			almost low	0.28
327			small	small			almost low	0.32
328			small		high		medium	0.36
329			small		medium		almost low	0.34
330			small		low		almost low	0.36



APPENDIX B

SOURCE CODE INFERENCE ENGINE



```
Private Sub CmdAccept_Click()
```

```
Dim i As Integer
```

```
Dim SelectItem As Integer
```

```
Dim NodX As Node
```

```
Dim row As Integer
```

```
'-----
```

```
' Check if selected item on treeview already on MSFlexGrid1
```

```
For i = 1 To MSFlexGrid1.Rows - 1 Step 1
```

```
MSFlexGrid1.row = i
```

```
MSFlexGrid1.col = 1
```

```
If TreeView1.SelectedItem.Text = MSFlexGrid1.Text Then
```

```
Dim txt As String
```

```
MSFlexGrid1.col = 0
```

```
txt = "This Part of Configuration is selected, number " + MSFlexGrid1.Text
```

```
MsgBox (txt)
```

```
Exit Sub
```

```
End If
```

```
If (TreeView1.SelectedItem.Index > 1) And (TreeView1.SelectedItem.Index < 5) Then
```

```
MSFlexGrid1.col = 4
```

```
If MSFlexGrid1.Text = "4" Then
```

```
Dim Txt2 As String
```

```
MSFlexGrid1.col = 0
```

```
Txt2 = "You have selected CPU low level already, number " + MSFlexGrid1.Text
```

```
MsgBox (Txt2)
```

```
Exit Sub
```

```
End If
```

```
If MSFlexGrid1.Text = "3" Then
```

```
Dim Txt3 As String
```



```

MSFlexGrid1.col = 0
Txt3 = "You have selected CPU medium level already, number " +
MSFlexGrid1.Text
MsgBox (Txt3)

Exit Sub
End If

If MSFlexGrid1.Text = "2" Then
    Dim Txt4 As String
    MSFlexGrid1.col = 0
    Txt4 = "You have selected CPU high level already, number " +
MSFlexGrid1.Text
    MsgBox (Txt4)

    Exit Sub
End If

End If

=====
If (TreeView1.SelectedItem.Index > 4) And (TreeView1.SelectedItem.Index < 9) Then
    MSFlexGrid1.col = 4

    If MSFlexGrid1.Text = "6" Then
        Dim Txt6 As String
        MSFlexGrid1.col = 0
        Txt6 = "You have selected Monitor large size already, number " +
MSFlexGrid1.Text
        MsgBox (Txt6)

        Exit Sub
    End If

    If MSFlexGrid1.Text = "7" Then
        Dim Txt7 As String
        MSFlexGrid1.col = 0
        Txt7 = "You have selected Monitor medium size already, number " +
MSFlexGrid1.Text

```

MsgBox (Txt7)

Exit Sub

End If

If MSFlexGrid1.Text = "8" Then

Dim Txt8 As String

MSFlexGrid1.col = 0

Txt8 = "You have selected Monitor small size already, number " +
MSFlexGrid1.Text

MsgBox (Txt8)

Exit Sub

End If

End If

If (TreeView1.SelectedItem.Index > 10) And (TreeView1.SelectedItem.Index < 14) Then

MSFlexGrid1.col = 4

If MSFlexGrid1.Text = "11" Then

Dim Txt9 As String

MSFlexGrid1.col = 0

Txt9 = "You have selected High Quality VGA Card, number " +
MSFlexGrid1.Text

MsgBox (Txt9)

Exit Sub

End If

If MSFlexGrid1.Text = "12" Then

Dim Txt10 As String

MSFlexGrid1.col = 0

Txt10 = "You have selected Medium Quality VGA Card, number " +
MSFlexGrid1.Text

MsgBox (Txt10)

```
Exit Sub
End If
```

```
If MSFlexGrid1.Text = "13" Then
    Dim Txt13 As String
    MSFlexGrid1.col = 0
    Txt13 = "You have selected Low Quality VGA Card, number " +
MSFlexGrid1.Text
    MsgBox (Txt13)
```

```
Exit Sub
End If
```

```
End If
```

```
=====
If (TreeView1.SelectedItem.Index > 14) And (TreeView1.SelectedItem.Index < 18) Then
```

```
    MSFlexGrid1.col = 4
```

```
    If MSFlexGrid1.Text = "15" Then
```

```
        Dim Txt14 As String
```

```
        MSFlexGrid1.col = 0
```

```
        Txt14 = "You have selected Large size Hard Disk, number " + MSFlexGrid1.Text
```

```
        MsgBox (Txt14)
```

```
Exit Sub
End If
```

```
If MSFlexGrid1.Text = "16" Then
```

```
    Dim Txt15 As String
```

```
    MSFlexGrid1.col = 0
```

```
    Txt15 = "You have selected Medium size Hard Disk, number " +
MSFlexGrid1.Text
```

```
    MsgBox (Txt15)
```

```
Exit Sub
End If
```

If MSFlexGrid1.Text = "17" Then

Dim Txt16 As String

MSFlexGrid1.col = 0

Txt16 = "You have selected Small size Hard Disk, number " + MSFlexGrid1.Text

MsgBox (Txt16)

Exit Sub

End If

End If

=====

If (TreeView1.SelectedItem.Index > 17) And (TreeView1.SelectedItem.Index < 21) Then

MSFlexGrid1.col = 4

If MSFlexGrid1.Text = "18" Then

Dim Txt18 As String

MSFlexGrid1.col = 0

Txt18 = "You have selected High Memory Capacity, number " +
MSFlexGrid1.Text

MsgBox (Txt18)

Exit Sub

End If

If MSFlexGrid1.Text = "19" Then

Dim Txt19 As String

MSFlexGrid1.col = 0

Txt19 = "You have selected Medium Memory Capacity, number " +
MSFlexGrid1.Text

MsgBox (Txt19)

Exit Sub

End If

If MSFlexGrid1.Text = "20" Then

Dim Txt20 As String


```

MSFlexGrid1.col = 0
Txt20 = "You have selected Low Memory Capacity, number " +
MSFlexGrid1.Text
MsgBox (Txt20)

Exit Sub
End If

```

```

End If

```

```

Next i

```

```

If TreeView1.SelectedItem.Index > 1 Then
    SelectItem = TreeView1.SelectedItem.Index

    row = MSFlexGrid1.Rows - 1 - SelectedMonitor + SelectedPeripheral +
SelectedHDD_MEM + 1
    If CboDegree.ListIndex = -1 Then
        MsgBox ("You should choose Range of The Part")
        Exit Sub
    End If

```

```

'Add item on CPU

```

```

If (SelectItem <= ItemCPU + 1) And (SelectItem > 1) Then
    SelectedCPU = SelectedCPU + 1
    MSFlexGrid1.AddItem ""
    MSFlexGrid1.col = 0
    MSFlexGrid1.row = row
    MSFlexGrid1.Clip = row

    MSFlexGrid1.col = 1
    MSFlexGrid1.row = row
    MSFlexGrid1.Clip = TreeView1.SelectedItem.Text

```

```
MSFlexGrid1.col = 2
MSFlexGrid1.row = row
MSFlexGrid1.Clip = CboDegree.List(CboDegree.ListIndex)
```

```
MSFlexGrid1.col = 3
MSFlexGrid1.row = row
MSFlexGrid1.Clip = CboDeg.List(CboDegree.ListIndex)
```

```
MSFlexGrid1.col = 4
MSFlexGrid1.row = row
MSFlexGrid1.Clip = TreeView1.SelectedItem.Index
```

End If

'Add item on Monitor

If (SelectedItem <= ItemMonitor + ItemCPU + 2) And (SelectedItem > ItemCPU + 2)

Then

```
SelectedMonitor = SelectedMonitor + 1
```

```
MSFlexGrid1.AddItem ""
```

```
MSFlexGrid1.col = 0
```

```
MSFlexGrid1.row = row
```

```
MSFlexGrid1.Clip = row
```

```
MSFlexGrid1.col = 1
```

```
MSFlexGrid1.row = row
```

```
MSFlexGrid1.Clip = TreeView1.SelectedItem.Text
```

```
MSFlexGrid1.col = 2
```

```
MSFlexGrid1.row = row
```

```
MSFlexGrid1.Clip = CboDegree.List(CboDegree.ListIndex)
```

```
MSFlexGrid1.col = 3
```

```
MSFlexGrid1.row = row
```

```
MSFlexGrid1.Clip = CboDeg.List(CboDegree.ListIndex)
```

```
MSFlexGrid1.col = 4
```

```
MSFlexGrid1.row = row
```

```
MSFlexGrid1.Clip = TreeView1.SelectedItem.Index
```

End If

'Add item on Peripheral

If (SelectedItem <= ItemCPU + ItemMonitor + ItemPeripheral + 3) And (SelectedItem > ItemCPU + ItemMonitor + 3) Then

SelectedPeripheral = SelectedPeripheral + 1

MSFlexGrid1.AddItem ""

MSFlexGrid1.col = 0

MSFlexGrid1.row = row

MSFlexGrid1.Clip = row

MSFlexGrid1.col = 1

MSFlexGrid1.row = row

MSFlexGrid1.Clip = TreeView1.SelectedItem.Text

MSFlexGrid1.col = 2

MSFlexGrid1.row = row

MSFlexGrid1.Clip = CboDegree.List(CboDegree.ListIndex)

MSFlexGrid1.col = 3

MSFlexGrid1.row = row

MSFlexGrid1.Clip = CboDeg.List(CboDegree.ListIndex)

MSFlexGrid1.col = 4

MSFlexGrid1.row = row

MSFlexGrid1.Clip = TreeView1.SelectedItem.Index

End If

'Add item on HDD_MEM

If (SelectedItem <= ItemCPU + ItemMonitor + ItemPeripheral + ItemHDD_MEM + 4) And (SelectedItem > ItemCPU + ItemMonitor + ItemPeripheral + 4) Then

SelectedHDD_MEM = SelectedHDD_MEM + 1

MSFlexGrid1.AddItem ""

MSFlexGrid1.col = 0

MSFlexGrid1.row = row

MSFlexGrid1.Clip = row

```
MSFlexGrid1.col = 1
MSFlexGrid1.row = row
MSFlexGrid1.Clip = TreeView1.SelectedItem.Text
```

```
MSFlexGrid1.col = 2
MSFlexGrid1.row = row
MSFlexGrid1.Clip = CboDegree.List(CboDegree.ListIndex)
```

```
MSFlexGrid1.col = 3
MSFlexGrid1.row = row
MSFlexGrid1.Clip = CboDeg.List(CboDegree.ListIndex)
```

```
MSFlexGrid1.col = 4
MSFlexGrid1.row = row
MSFlexGrid1.Clip = TreeView1.SelectedItem.Index
```

End If

End If

End Sub

Private Sub CmdApplyBudget_Click0

Dim NewPrice, RecommendedPrice As Integer

NewPrice = Text9.Text

RecommendedPrice = Text7.Text

If NewPrice = "" Then

MsgBox ("Please type YOUR BUDGET here, please!")

Exit Sub

End If

'-----

If NewPrice < 310 Then

MsgBox ("I am sorry, your budget is not enough to buy EVEN a low configuration,
please choose a HIGHER budget!")

Exit Sub

End If

'-----

If NewPrice > 1550 Then

MsgBox ("Your budget is too much for buying a PC, Please choose a LOWER budget!")

Exit Sub

End If

'-----

'-----

If NewPrice > RecommendedPrice Then

MsgBox ("Please review your selected requirements and choose a HIGHER configuration!")

Text9.Text = ""

Exit Sub

End If

'-----

If NewPrice < RecommendedPrice Then

MsgBox ("Please review your selected requirements and choose a LOWER configuration!")

Text9.Text = ""

Exit Sub

End If

'-----

'-----

End Sub

Private Sub CmdBack_Click0

MsgBox ("Thank you for using Advising program, please click OK to exit")

FrmStart.WindowState = 2

FrmStart.Show

Unload FrmDiag

End Sub

Private Sub CmdDeSelect_Click0

Dim Message As String
Dim DialogType As Integer
Dim Title As Integer
Dim Response As Integer

Dim row, i As Integer

row = MSFlexGrid1.row

If (row > 0) And (row < MSFlexGrid1.Rows - 1) Then

MSFlexGrid1.col = 1

Message = "Deselect Parts " + CStr(row) + ": " + MSFlexGrid1.Text + "?"

DialogType = MB_YESNO + MB_ICONQUESTION

Response = MsgBox(Message, DialogType, Title)

If Response = IDYES Then

MSFlexGrid1.Move row

MSFlexGrid1.RemoveItem row

For i = row To MSFlexGrid1.Rows - 2

MSFlexGrid1.col = 0

MSFlexGrid1.row = i

MSFlexGrid1.Clip = i

Next i

End If

End If

'Dim NodX As Node

'If (TreeView2.SelectedItem.Index > 1) And (TreeView2.SelectedItem.Text <> "Monitor")
And (TreeView2.SelectedItem.Text <> "Peripheral") And (TreeView2.SelectedItem.Text
<> "HDD_MEM and Palpation") Then TreeView2.Nodes.Remove
(TreeView2.SelectedItem.Index)

End Sub

Private Sub CmdInference_Click()

If CmdApplyBudget.Visible = True Then FrmResult.Visible = False

Text1.Text = ""

Text2.Text = ""

Text3.Text = ""

Text4.Text = ""

Text5.Text = ""

Text6.Text = ""

Text7.Text = ""

TxtMatchRules = "" 'Initiate public variable

Dim i, j, No, k, No1 As Integer

Dim NumberRules As Single

Dim High_conf As Single 'High configuration

Dim Almost_High_conf As Single 'Almost High configuration

Dim Medium_conf As Single 'Middle configuration

Dim Almost_low_conf As Single 'Low configuration

Dim Low_conf As Single 'Almost Low configuration

Dim Reserved_Conf As Single 'undefine

Dim Weight_Config As Single

Dim Choose_level As Single '1 = low, 2 = almost low, 3 = medium, 4 = almost high, 5 = high

Dim Found_Degree As Boolean

Dim High_conf1 As Single ' Gia tri tam thoi de tinh trong mot luat

Dim Almost_High_conf1 As Single

Dim Medium_conf1 As Single

Dim Almost_low_conf1 As Single

Dim Low_conf1 As Single

Dim Reserved_Conf1 As Single

Dim Sunb_config1, Sunb_config2, Sunb_config3, Sunb_config4, Sunb_config5,
Sunb_config6, Sunb_config7, Sunb_config8 As String

'Toi da 8 cau hinh trong mot luat

Dim d1 As Single

Dim d2 As Single

Dim d3 As Single

Dim d4 As Single

Dim d5 As Single

Dim d6 As Single

Dim d7 As Single

Dim d8 As Single

Dim Deg As Single

Dim TxtCol3 As String

Dim TxtCol1 As String

Dim Found As Boolean

MyrecRule.MoveLast

No = MyrecRule.RecordCount

'=====

'Initiate variable

High_conf = 0

Almost_High_conf = 0

Medium_conf = 0

Almost_low_conf = 0

Low_conf = 0

Reserved_Conf = 0

d1 = 1 ' Trong so tam thoi cua Sunb_config 1 trong luat

d2 = 1

d3 = 1

d4 = 1

d5 = 1

d6 = 1

d7 = 1

d8 = 1

'=====

If No > 0 Then

ProgressBar1.Max = No - 1

ProgressBar2.Max = MSFlexGrid1.Rows - 1

ProgressBar1.Value = 0

ProgressBar2.Value = 0

ProgressBar1.Visible = True

ProgressBar2.Visible = True

'=====

MyrecRule.MoveFirst

For i = 0 To No - 1 ' Check on all data rule

Found = False

'-----

' Get Sunb_configs and Degree

'MyrecRule.Move i

NumberRules = MyrecRule.Fields("Order1").Value

Sunb_config1 = MyrecRule.Fields("Sunb_config1").Value 'Get Sunb_config 1 on rules i

Sunb_config2 = MyrecRule.Fields("Sunb_config2").Value

Sunb_config3 = MyrecRule.Fields("Sunb_config3").Value

Sunb_config4 = MyrecRule.Fields("Sunb_config4").Value

Sunb_config5 = MyrecRule.Fields("Sunb_config5").Value

Sunb_config6 = MyrecRule.Fields("Sunb_config6").Value

Sunb_config7 = MyrecRule.Fields("Sunb_config7").Value

Sunb_config8 = MyrecRule.Fields("Sunb_config8").Value

High_conf1 = MyrecRule.Fields("Weight of High_conf").Value

Almost_High_conf1 = MyrecRule.Fields("Weight of Almost_High_conf").Value

Medium_conf1 = MyrecRule.Fields("Weight of Medium_conf").Value

Almost_low_conf1 = MyrecRule.Fields("Weight of Almost_low_conf").Value

Low_conf1 = MyrecRule.Fields("Weight of Low_conf").Value

Reserved_Conf1 = MyrecRule.Fields("Weight of Reserved_Conf").Value

'-----

'Check Sunb_config 1 on rule i

For j = 1 To MSFlexGrid1.Rows - 1 ' loop on working memory

ProgressBar2.Value = j

MSFlexGrid1.row = j

MSFlexGrid1.col = 1

TxtCol1 = MSFlexGrid1.Text

MSFlexGrid1.row = j

MSFlexGrid1.col = 3

TxtCol3 = MSFlexGrid1.Text ' only display

If (Sunb_config1 = TxtCol1) Then

d1 = TxtCol3

Found = True

End If

Next j

'-----

'-----

'Check Sunb_config 2 on rule i

If (Sunb_config2 <> "") And Found Then

```

Found = False
For j = 1 To MSFlexGrid1.Rows - 1 ' loop on working memory
    ProgressBar2.Value = j

    MSFlexGrid1.row = j
    MSFlexGrid1.col = 1
    TxtCol1 = MSFlexGrid1.Text

    MSFlexGrid1.row = j
    MSFlexGrid1.col = 3
    TxtCol3 = MSFlexGrid1.Text

    If Sunb_config2 = TxtCol1 Then
        Found = True
        d2 = TxtCol3
    End If
Next j
'-----
'Check Sunb_config 3 on rule i
If (Sunb_config3 <> "") And Found Then
    Found = False
    For j = 1 To MSFlexGrid1.Rows - 1 ' loop on working memory
        ProgressBar2.Value = j

        MSFlexGrid1.row = j
        MSFlexGrid1.col = 1
        TxtCol1 = MSFlexGrid1.Text

        MSFlexGrid1.row = j
        MSFlexGrid1.col = 3
        TxtCol3 = MSFlexGrid1.Text

        If Sunb_config3 = TxtCol1 Then

```

```

        Found = True
        d3 = TxtCol3
    End If

```

```

Next j

```

```

'-----

```

```

'Check Sunb_config 4 on rule i

```

```

    If (Sunb_config4 <> "") And Found Then

```

```

        Found = False

```

```

        For j = 1 To MSFlexGrid1.Rows - 1 ' loop on working memory

```

```

            ProgressBar2.Value = j

```

```

            MSFlexGrid1.row = j

```

```

            MSFlexGrid1.col = 1

```

```

            TxtCol1 = MSFlexGrid1.Text

```

```

            MSFlexGrid1.row = j

```

```

            MSFlexGrid1.col = 3

```

```

            TxtCol3 = MSFlexGrid1.Text

```

```

            If Sunb_config4 = TxtCol1 Then

```

```

                Found = True

```

```

                d4 = TxtCol3

```

```

            End If

```

```

        Next j

```

```

'-----

```

```

'Check Sunb_config 5 on rule i

```

```

    If (Sunb_config5 <> "") And Found Then

```

```

        Found = False

```

```

        For j = 1 To MSFlexGrid1.Rows - 1 ' loop on working memory

```

```

            ProgressBar2.Value = j

```

```

            MSFlexGrid1.row = j

```

```

            MSFlexGrid1.col = 1

```

```

            TxtCol1 = MSFlexGrid1.Text

```



```

MSFlexGrid1.row = j
MSFlexGrid1.col = 3
TxtCol3 = MSFlexGrid1.Text

```

```

If Sunb_config5 = TxtCol1 Then
    Found = True
    d5 = TxtCol3
End If

```

```

Next j

```

```

'-----

```

```

'Check Sunb_config 6 on rule i

```

```

If (Sunb_config6 <> "") And Found Then

```

```

    Found = False

```

```

    For j = 1 To MSFlexGrid1.Rows - 1 ' loop on working memory

```

```

        ProgressBar2.Value = j

```

```

        MSFlexGrid1.row = j

```

```

        MSFlexGrid1.col = 1

```

```

        TxtCol1 = MSFlexGrid1.Text

```

```

        MSFlexGrid1.row = j

```

```

        MSFlexGrid1.col = 3

```

```

        TxtCol3 = MSFlexGrid1.Text

```

```

    If Sunb_config6 = TxtCol1 Then

```

```

        Found = True

```

```

        d6 = TxtCol3

```

```

    End If

```

```

Next j

```

```

'-----

```

```

'Check Sunb_config 7 on rule i

```

```

If (Sunb_config7 <> "") And Found Then

```

```

    Found = False

```

```

    For j = 1 To MSFlexGrid1.Rows - 1 ' loop on working memory

```

ProgressBar2.Value = j

MSFlexGrid1.row = j

MSFlexGrid1.col = 1

TxtCol1 = MSFlexGrid1.Text

MSFlexGrid1.row = j

MSFlexGrid1.col = 3

TxtCol3 = MSFlexGrid1.Text

If Sunb_config7 = TxtCol1 Then

Found = True

d7 = TxtCol3

End If

Next j

'Check Sunb_config 8 on rule i

If (Sunb_config8 <> "") And Found Then

Found = False

For j = 1 To MSFlexGrid1.Rows - 1 ' loop on working memory

ProgressBar2.Value = j

MSFlexGrid1.row = j

MSFlexGrid1.col = 1

TxtCol1 = MSFlexGrid1.Text

MSFlexGrid1.row = j

MSFlexGrid1.col = 3

TxtCol3 = MSFlexGrid1.Text

If Sunb_config8 = TxtCol1 Then

Found = True

d8 = TxtCol3

End If

Next j

```

'-----

End If ' If Sunb_config8 <> ""
End If ' If Sunb_config7 <> ""
End If ' If Sunb_config6 <> ""
End If ' If Sunb_config5 <> ""
End If ' If Sunb_config4 <> ""
End If ' If Sunb_config3 <> ""
End If ' If Sunb_config2 <> ""

'-----

' CALCULATE
If Found Then
    TxtMatchRules = TxtMatchRules + CStr(NumberRules) + ", "

    ' Degree của mot luat = min của tat ca cac tien de

    Deg = Min(d1, Min(d2, (Min(d3, Min(d4, (Min(d5, Min(d6, Min(d7, d8))))))))))

    'Degree của toan bo cac level ke tu khi co luat nay = min (luat) va tien de

    High_conf1 = Min(Deg, High_conf1)
    Almost_High_conf1 = Min(Deg, Almost_High_conf1)
    Medium_conf1 = Min(Deg, Medium_conf1)
    Almost_low_conf1 = Min(Deg, Almost_low_conf1)
    Low_conf1 = Min(Deg, Low_conf1)

    High_conf = MaxProduct(High_conf, High_conf1)
    Almost_High_conf = MaxProduct(Almost_High_conf, Almost_High_conf1)
    Medium_conf = MaxProduct(Medium_conf, Medium_conf1)
    Almost_low_conf = MaxProduct(Almost_low_conf, Almost_low_conf1)
    Low_conf = MaxProduct(Low_conf, Low_conf1)

End If ' If Found

'=====

ProgressBar1.Value = i
MyrecRule.MoveNext

```

Next i

.....

End If ' If No > 0

Text1.Text = High_conf ' high

Text2.Text = Almost_High_conf ' almost high

Text3.Text = Medium_conf ' medium

Text4.Text = Almost_low_conf ' almost low

Text5.Text = Low_conf ' low

Weight_Config = Max(High_conf, Max(Almost_High_conf, Max(Medium_conf, Max(Almost_low_conf, Low_conf))))

If Weight_Config = 0 Then

MsgBox ("I can not determine which configuration would be bought. Please review your requirements!")

Exit Sub

End If

Choose_level = 0

If (Weight_Config = Low_conf) Then

Choose_level = 5

End If

If (Weight_Config = Almost_low_conf) Then

Choose_level = 4

End If

If (Weight_Config = Medium_conf) Then

Choose_level = 3

End If

If Weight_Config = Almost_High_conf Then

Choose_level = 2

End If

If Weight_Config = High_conf Then

Choose_level = 1

End If

' tính lại trong số tổng nếu các trong số bằng nhau

If (Weight_Config = High_conf) And (Weight_Config = Almost_High_conf) Then
Choose_level = 2

If (Weight_Config = Almost_High_conf) And (Weight_Config = Medium_conf) Then
Choose_level = 3

If (Weight_Config = Medium_conf) And (Weight_Config = Almost_low_conf) Then
Choose_level = 4

If (Weight_Config = Almost_low_conf) And (Weight_Config = Low_conf) Then
Choose_level = 4

If (Weight_Config = High_conf) And (Weight_Config = Almost_High_conf) And
(Weight_Config = Medium_conf) Then Choose_level = 2

If (Weight_Config = Almost_High_conf) And (Weight_Config = Medium_conf) And
(Weight_Config = Almost_low_conf) Then Choose_level = 3

If (Weight_Config = Medium_conf) And (Weight_Config = Almost_low_conf) And
(Weight_Config = Low_conf) Then Choose_level = 4

' ===== *

' Tính giá hợp lý và đưa ra cấu hình

' Hight configurarion

FrmResult.Visible = True

' Public MyrecPrice_High, MyrecPrice_Monitor, MyrecPrice_Medium,
MyrecPrice_Almost_Low, MyrecPrice_Low As Recordset

If Choose_level = 1 Then

MyrecPrice_High.MoveLast

No1 = MyrecPrice_High.RecordCount

MyrecPrice_High.MoveFirst

Found_Degree = False

Text8.Text = "High"

```

For i = 1 To No1
If MyrecPrice_High.Fields("Weight").Value >= Weight_Config Then
    Text6.Text = MyrecPrice_High.Fields("Configuration").Value
    Text7.Text = MyrecPrice_High.Fields("Price").Value
    MyrecPrice_High.MoveNext
End If

Next i

End If

If Choose_level = 2 Then

    MyrecPrice_Monitor.MoveLast
    No1 = MyrecPrice_Monitor.RecordCount
    MyrecPrice_Monitor.MoveFirst
    Text8.Text = "Almost High"

    For i = 1 To No1
    If MyrecPrice_Monitor.Fields("Weight").Value >= Weight_Config Then
        Text6.Text = MyrecPrice_Monitor.Fields("Configuration").Value
        Text7.Text = MyrecPrice_Monitor.Fields("Price").Value
        MyrecPrice_Monitor.MoveNext
    End If

    Next i

End If

If Choose_level = 3 Then

    MyrecPrice_Medium.MoveLast
    No1 = MyrecPrice_Medium.RecordCount
    MyrecPrice_Medium.MoveFirst
    Text8.Text = "Medium"

    For i = 1 To No1
    If MyrecPrice_Medium.Fields("Weight").Value >= Weight_Config Then
        Text6.Text = MyrecPrice_Medium.Fields("Configuration").Value
        Text7.Text = MyrecPrice_Medium.Fields("Price").Value
        MyrecPrice_Medium.MoveNext
    End If

    Next i

End If

```

```

End If

Next i

End If

If Choose_level = 4 Then

    MyrecPrice_Almost_Low.MoveLast
    No1 = MyrecPrice_Almost_Low.RecordCount
    MyrecPrice_Almost_Low.MoveFirst
    Text8.Text = "Almost Low"

    For i = 1 To No1
        If MyrecPrice_Almost_Low.Fields("Weight").Value >= Weight_Config Then
            Text6.Text = MyrecPrice_Almost_Low.Fields("Configuration").Value
            Text7.Text = MyrecPrice_Almost_Low.Fields("Price").Value
            MyrecPrice_Almost_Low.MoveNext
        End If
    Next i

End If

If Choose_level = 5 Then

    MyrecPrice_Low.MoveLast
    No1 = MyrecPrice_Low.RecordCount
    MyrecPrice_Low.MoveFirst
    Text8.Text = "Low"

    For i = 1 To No1
        If MyrecPrice_Low.Fields("Weight").Value >= Weight_Config Then
            Text6.Text = MyrecPrice_Low.Fields("Configuration").Value
            Text7.Text = MyrecPrice_Low.Fields("Price").Value
            MyrecPrice_Low.MoveNext
        End If
    Next i

End If

LblSatisfied.Visible = True
CmdSatisfiedOK.Visible = True

```

End Sub

Private Sub CmdWhy_Click0

FrmExplanation.Show

End Sub

Private Sub CmdSatisfiedOK_Click0

LblBudget.Visible = True

Text9.Visible = True

CmdApplyBudget.Visible = True

Label12.Visible = True

End Sub

Private Sub Form_Load0

Text1.Text = ""

Text2.Text = ""

Text3.Text = ""

Text4.Text = ""

Text5.Text = ""

Text6.Text = ""

Text7.Text = ""

Dim NodX As Node

Dim i, NumberSunb_config, SelectItem As Integer

'Initiate Variable of Selected Item

SelectedCPU = 0

SelectedMonitor = 0

SelectedPeripheral = 0

SelectedHDD_MEM = 0

'Display Grid of Selected Item

MSFlexGrid1.ColWidth(0) = 400

MSFlexGrid1.ColWidth(1) = 3000

MSFlexGrid1.ColWidth(2) = 2400



MSFlexGrid1.ColWidth(3) = 2400

MSFlexGrid1.col = 0

MSFlexGrid1.row = 0

MSFlexGrid1.CellForeColor = &HFF0000

MSFlexGrid1.CellFontSize = 10

MSFlexGrid1.Clip = "No."

MSFlexGrid1.col = 1

MSFlexGrid1.row = 0

MSFlexGrid1.CellForeColor = &HFF0000

MSFlexGrid1.CellFontSize = 11

MSFlexGrid1.CellFontBold = True

MSFlexGrid1.Clip = "Requirements"

MSFlexGrid1.col = 2

MSFlexGrid1.row = 0

MSFlexGrid1.CellForeColor = &HFF0000

MSFlexGrid1.CellFontSize = 11

MSFlexGrid1.Clip = "Details"

MSFlexGrid1.col = 3

MSFlexGrid1.row = 0

MSFlexGrid1.CellForeColor = &HFF0000

MSFlexGrid1.CellFontSize = 10

MSFlexGrid1.Clip = "Degree of requirements"

MSFlexGrid1.col = 4

MSFlexGrid1.row = 0

MSFlexGrid1.CellForeColor = &HFF0000

MSFlexGrid1.CellFontSize = 10

MSFlexGrid1.Clip = "Order"

MSFlexGrid1.Font.Size = 12

'.....

'Update length of Tables

If Myrec.RecordCount > 0 Then Myrec.MoveLast

ItemCPU = Myrec.RecordCount

If MyrecMonitor.RecordCount Then MyrecMonitor.MoveLast

ItemMonitor = MyrecMonitor.RecordCount

If MyrecPeripheral.RecordCount Then MyrecPeripheral.MoveLast

ItemPeripheral = MyrecPeripheral.RecordCount

If MyrecHDD_MEM.RecordCount Then MyrecHDD_MEM.MoveLast

ItemHDD_MEM = MyrecHDD_MEM.RecordCount

.....

Set NodX = TreeView1.Nodes.Add(, "CPU", "Which CPU do you want?")

TreeView1.Nodes(1).Image = 1

'read Sunb_config from CPU

Myrec.MoveLast

NumberSunb_config = Myrec.RecordCount

Myrec.MoveFirst

For i = 1 To NumberSunb_config

Set NodX = TreeView1.Nodes.Add("CPU", tvwChild, "Node " & CStr(i),
Myrec.Fields("Sunb_config"))

TreeView1.Nodes(i + 1).Image = 2

Myrec.MoveNext

Next i

'Sunb_config table 2

Set NodX = TreeView1.Nodes.Add(, "Monitor", "Which monitor size do you want?")

TreeView1.Nodes(ItemCPU + 2).Image = 1

MyrecMonitor.MoveLast

NumberSunb_config = MyrecMonitor.RecordCount

MyrecMonitor.MoveFirst

For i = 1 To NumberSunb_config

```

Set NodX = TreeView1.Nodes.Add("Monitor", tvwChild, "Node1 " & CStr(i),
MyrecMonitor.Fields("Sunb_config"))
TreeView1.Nodes(i + 2 + ItemCPU).Image = 2
MyrecMonitor.MoveNext
Next i

```

'Sunb_config table 3

```

Set NodX = TreeView1.Nodes.Add(, "Peripheral", "Which Peripherals do you
want?")

```

```

TreeView1.Nodes(ItemCPU + ItemMonitor + 3).Image = 1

```

```

MyrecPeripheral.MoveLast
NumberSunb_config = MyrecPeripheral.RecordCount
MyrecPeripheral.MoveFirst

```

```

For i = 1 To NumberSunb_config
Set NodX = TreeView1.Nodes.Add("Peripheral", tvwChild, "Node2 " & CStr(i),
MyrecPeripheral.Fields("Sunb_config"))
TreeView1.Nodes(i + 3 + ItemCPU + ItemMonitor).Image = 2
MyrecPeripheral.MoveNext
Next i

```

'Sunb_config table 4

```

Set NodX = TreeView1.Nodes.Add(, "HDD_MEM", "Which capacity of hard disk and
memory do you want?")

```

```

TreeView1.Nodes(ItemCPU + ItemMonitor + ItemPeripheral + 4).Image = 1

```

```

MyrecHDD_MEM.MoveLast
NumberSunb_config = MyrecHDD_MEM.RecordCount
MyrecHDD_MEM.MoveFirst

```

```

For i = 1 To NumberSunb_config
Set NodX = TreeView1.Nodes.Add("HDD_MEM", tvwChild, "Node3 " & CStr(i),
MyrecHDD_MEM.Fields("Sunb_config"))
TreeView1.Nodes(i + 4 + ItemCPU + ItemMonitor + ItemPeripheral).Image = 2

MyrecHDD_MEM.MoveNext

```

```

Next i
NodX.EnsureVisible
' TreeView1.Style = 6 ' Style 3.
' TreeView1.BorderStyle = vbFixedSingle
TreeView1.Font.Size = 12

Set TreeView1.SelectedItem = TreeView1.Nodes(1)

```

End Sub

```

Private Sub TreeView1_Click()
Dim i, SelectItem As Integer
'If TreeView1.SelectedItem.Index < 2 Then Exit Sub

SelectItem = TreeView1.SelectedItem.Index ' Set Select Item

CboDegree.Text = "Please choice" ' text of Degree

' Clear all on Degree list
For i = 1 To 10
    If CboDegree.List(0) <> "" Then CboDegree.RemoveItem (0)
    If CboDeg.List(0) <> "" Then CboDeg.RemoveItem (0)
Next i

'set items of degree

'case click item on CPU

If (SelectItem <= ItemCPU + 1) And (SelectItem > 1) Then
    Myrec.MoveFirst
    Myrec.Move SelectItem - 2

    If Myrec.Fields("Type1") <> "" Then
        CboDegree.AddItem Myrec.Fields("Type1")
        CboDeg.AddItem Myrec.Fields("Degree1")

```


End If

If Myrec.Fields("Type2") <> "" Then

 CboDegree.AddItem Myrec.Fields("Type2")

 CboDeg.AddItem Myrec.Fields("Degree2")

End If

If Myrec.Fields("Type3") <> "" Then

 CboDegree.AddItem Myrec.Fields("Type3")

 CboDeg.AddItem Myrec.Fields("Degree3")

End If

If Myrec.Fields("Type4") <> "" Then

 CboDegree.AddItem Myrec.Fields("Type4")

 CboDeg.AddItem Myrec.Fields("Degree4")

End If

If Myrec.Fields("Type5") <> "" Then

 CboDegree.AddItem Myrec.Fields("Type5")

 CboDeg.AddItem Myrec.Fields("Degree5")

End If

If Myrec.Fields("Type6") <> "" Then

 CboDegree.AddItem Myrec.Fields("Type6")

 CboDeg.AddItem Myrec.Fields("Degree6")

End If

If Myrec.Fields("Type7") <> "" Then

 CboDegree.AddItem Myrec.Fields("Type7")

 CboDeg.AddItem Myrec.Fields("Degree7")

End If

If Myrec.Fields("Type8") <> "" Then

 CboDegree.AddItem Myrec.Fields("Type8")

 CboDeg.AddItem Myrec.Fields("Degree8")

End If

End If

'case click item on Monitor

If (SelectItem <= ItemMonitor + ItemCPU + 2) And (SelectItem > ItemCPU + 2) Then
MyrecMonitor.MoveFirst
MyrecMonitor.Move SelectItem - ItemCPU - 3

If MyrecMonitor.Fields("Type1") <> "" Then
CboDegree.AddItem MyrecMonitor.Fields("Type1")
CboDeg.AddItem MyrecMonitor.Fields("Degree1")
End If

If MyrecMonitor.Fields("Type2") <> "" Then
CboDegree.AddItem MyrecMonitor.Fields("Type2")
CboDeg.AddItem MyrecMonitor.Fields("Degree2")
End If

If MyrecMonitor.Fields("Type3") <> "" Then
CboDegree.AddItem MyrecMonitor.Fields("Type3")
CboDeg.AddItem MyrecMonitor.Fields("Degree3")
End If

If MyrecMonitor.Fields("Type4") <> "" Then
CboDegree.AddItem MyrecMonitor.Fields("Type4")
CboDeg.AddItem MyrecMonitor.Fields("Degree4")
End If

If MyrecMonitor.Fields("Type5") <> "" Then
CboDegree.AddItem MyrecMonitor.Fields("Type5")
CboDeg.AddItem MyrecMonitor.Fields("Degree5")
End If

If MyrecMonitor.Fields("Type6") <> "" Then
CboDegree.AddItem MyrecMonitor.Fields("Type6")
CboDeg.AddItem MyrecMonitor.Fields("Degree6")
End If

If MyrecMonitor.Fields("Type7") <> "" Then
CboDegree.AddItem MyrecMonitor.Fields("Type7")
CboDeg.AddItem MyrecMonitor.Fields("Degree7")
End If

If MyrecMonitor.Fields("Type8") <> "" Then

 CboDegree.AddItem MyrecMonitor.Fields("Type8")

 CboDeg.AddItem MyrecMonitor.Fields("Degree8")

End If

End If

'case click item on Peripheral

If (SelectedItem <= ItemCPU + ItemMonitor + ItemPeripheral + 3) And (SelectedItem > ItemCPU + ItemMonitor + 3) Then

 MyrecPeripheral.MoveFirst

 MyrecPeripheral.Move SelectedItem - ItemCPU - ItemMonitor - 4

If MyrecPeripheral.Fields("Type1") <> "" Then

 CboDegree.AddItem MyrecPeripheral.Fields("Type1")

 CboDeg.AddItem MyrecPeripheral.Fields("Degree1")

End If

If MyrecPeripheral.Fields("Type2") <> "" Then

 CboDegree.AddItem MyrecPeripheral.Fields("Type2")

 CboDeg.AddItem MyrecPeripheral.Fields("Degree2")

End If

If MyrecPeripheral.Fields("Type3") <> "" Then

 CboDegree.AddItem MyrecPeripheral.Fields("Type3")

 CboDeg.AddItem MyrecPeripheral.Fields("Degree3")

End If

If MyrecPeripheral.Fields("Type4") <> "" Then

 CboDegree.AddItem MyrecPeripheral.Fields("Type4")

 CboDeg.AddItem MyrecPeripheral.Fields("Degree4")

End If

If MyrecPeripheral.Fields("Type5") <> "" Then

 CboDegree.AddItem MyrecPeripheral.Fields("Type5")

 CboDeg.AddItem MyrecPeripheral.Fields("Degree5")

End If

If MyrecPeripheral.Fields("Type6") <> "" Then

 CboDegree.AddItem MyrecPeripheral.Fields("Type6")

```
CboDeg.AddItem MyrecPeripheral.Fields("Degree6")
End If
```

```
If MyrecPeripheral.Fields("Type7") <> "" Then
    CboDegree.AddItem MyrecPeripheral.Fields("Type7")
    CboDeg.AddItem MyrecPeripheral.Fields("Degree7")
End If
```

```
If MyrecPeripheral.Fields("Type8") <> "" Then
    CboDegree.AddItem MyrecPeripheral.Fields("Type8")
    CboDeg.AddItem MyrecPeripheral.Fields("Degree8")
End If
```

```
End If
```

```
'case click item on HDD_MEM
```

```
If (SelectItem <= ItemCPU + ItemMonitor + ItemPeripheral + ItemHDD_MEM + 4) And  
(SelectItem > ItemCPU + ItemMonitor + ItemPeripheral + 4) Then
```

```
    MyrecHDD_MEM.MoveFirst
```

```
    MyrecHDD_MEM.Move SelectItem - ItemCPU - ItemMonitor - ItemPeripheral - 5
```

```
If MyrecHDD_MEM.Fields("Type1") <> "" Then
    CboDegree.AddItem MyrecHDD_MEM.Fields("Type1")
    CboDeg.AddItem MyrecHDD_MEM.Fields("Degree1")
End If
```

```
If MyrecHDD_MEM.Fields("Type2") <> "" Then
    CboDegree.AddItem MyrecHDD_MEM.Fields("Type2")
    CboDeg.AddItem MyrecHDD_MEM.Fields("Degree2")
End If
```

```
If MyrecHDD_MEM.Fields("Type3") <> "" Then
    CboDegree.AddItem MyrecHDD_MEM.Fields("Type3")
    CboDeg.AddItem MyrecHDD_MEM.Fields("Degree3")
End If
```

```
If MyrecHDD_MEM.Fields("Type4") <> "" Then
    CboDegree.AddItem MyrecHDD_MEM.Fields("Type4")
```



```
CboDeg.AddItem MyrecHDD_MEM.Fields("Degree4")  
End If
```

```
If MyrecHDD_MEM.Fields("Type5") <> "" Then  
    CboDegree.AddItem MyrecHDD_MEM.Fields("Type5")  
    CboDeg.AddItem MyrecHDD_MEM.Fields("Degree5")  
End If
```

```
If MyrecHDD_MEM.Fields("Type6") <> "" Then  
    CboDegree.AddItem MyrecHDD_MEM.Fields("Type6")  
    CboDeg.AddItem MyrecHDD_MEM.Fields("Degree6")  
End If
```

```
If MyrecHDD_MEM.Fields("Type7") <> "" Then  
    CboDegree.AddItem MyrecHDD_MEM.Fields("Type7")  
    CboDeg.AddItem MyrecHDD_MEM.Fields("Degree7")  
End If
```

```
If MyrecHDD_MEM.Fields("Type8") <> "" Then  
    CboDegree.AddItem MyrecHDD_MEM.Fields("Type8")  
    CboDeg.AddItem MyrecHDD_MEM.Fields("Degree8")  
End If
```

```
End If
```

```
End Sub
```



APPENDIX C

SOURCE CODE KNOWLEDGE BASE



```

Private Sub cmdAcceptWeight_Click()
If (TxtHigh.Text > 1) Or (TxtHigh.Text = "") Then
    MsgBox ("Your selected Number is Fail")
    Exit Sub
End If

```

```

TxtHigh2.Text = TxtHigh.Text
TxtAlmost_High2.Text = TxtAlmost_High.Text
TxtMedium2.Text = TxtMedium.Text
TxtAlmost_low2.Text = TxtAlmost_low.Text
txtLow2.Text = txtLow.Text
TxtReserved_conf2.Text = TxtReserved_conf.Text

```

```

End Sub

```

```

Private Sub CmdAddSubconfig_Premise_Click()
FrmUpdateSubconfig_Premise.WindowState = 2
FrmUpdateSubconfig_Premise.Show
Unload FrmAcquisition
End Sub

```

```

Private Sub CmdBack_Click()
Unload FrmAcquisition
FrmStart.WindowState = 2
FrmStart.Show

```

```

End Sub

```

```

Private Sub CmdAccept_Click()
'-----
'Check if select Group
If (MSFlexGrid1.row = 1) Or (MSFlexGrid1.row = ItemCPU + 2) Or (MSFlexGrid1.row =
ItemCPU + ItemMonitor + 3) Or (MSFlexGrid1.row = ItemCPU + ItemMonitor +
ItemAlmost_Peripheral + 4) Or (MSFlexGrid1.row = ItemCPU + ItemMonitor +
ItemAlmost_Peripheral + ItemHDD_MEM + 5) Then Exit Sub
'-----
Dim i As Integer

```

```

i = 1
While a(i) <> ""
    i = i + 1
Wend

List1.AddItem MSFlexGrid1.Text ' display
a(i) = MSFlexGrid1.Text

End Sub

```

```

Private Sub CmdEditRule_Click0
FrmViewRules.WindowState = 2
FrmViewRules.Show
Unload FrmAcquisition
End Sub

```

```

Private Sub CmdRemove_Click0
Dim i, j As Integer
i = List1.ListIndex
If i <> -1 Then List1.RemoveItem i
For j = i + 1 To 14
    a(j) = a(j + 1)
Next j
End Sub

```

```

Private Sub CmdWrite_Click0
Dim i, j, k, aa, bb, c, d, e, g 'As Integer
Dim click As Boolean ' the variable represent this rule is update or not
click = False

```

```

    aa = TxtHigh2.Text
    bb = TxtAlmost_High2.Text
    c = TxtMedium2.Text
    d = TxtAlmost_low2.Text
    e = txtLow2.Text
    g = 0 'TxtReserved_conf2.Text

```


If (a(1) = "") And (a(2) = "") And (a(3) = "") And (a(4) = "") And (a(5) = "") And (a(6) = "")
And (a(7) = "") And (a(8) = "") Then click = True

If click Then

MsgBox ("You should choose Subconfig_Premise!")

Exit Sub

End If

Dim error As Boolean

error = False

If (aa > 1) Then error = True

If (bb > 1) Then error = True

If (c > 1) Then error = True

If (d > 1) Then error = True

If (e > 1) Then error = True

If (g > 1) Then error = True

If error Then

MsgBox ("The weight You choice are invalid")

Exit Sub

End If

MyrecRule.MoveLast*

i = MyrecRule.RecordCount

'-----

'Check rule if it is already on data base

For k = 0 To i - 1 Step 1

MyrecRule.MoveFirst

MyrecRule.Move k

If a(1) = MyrecRule("Subconfig_Premise1").Value Then

If a(2) = MyrecRule("Subconfig_Premise2").Value Then

If a(3) = MyrecRule("Subconfig_Premise3").Value Then

If a(4) = MyrecRule("Subconfig_Premise4").Value Then

If a(5) = MyrecRule("Subconfig_Premise5").Value Then

If a(6) = MyrecRule("Subconfig_Premise6").Value Then

If a(7) = MyrecRule("Subconfig_Premise7").Value Then

```

If a(8) = MyrecRule("Subconfig_Premise8").Value Then
    Dim txt As String
    txt = "This rule already on Rule Base, Number " + CStr(k + 1) + " !"
    MsgBox (txt)
    Exit Sub
End If
End If
End If
End If
End If
End If
End If

Next k

'-----
'Write

'MyrecRule.MoveLast

i = MyrecRule.RecordCount
MyrecRule.OpenRecordset
MyrecRule.AddNew*
MyrecRule.Fields("Order1") = i + 1
MyrecRule.Fields("Fired") = 0

MyrecRule.Fields("Weight of Highior") = aa 'TxtHigh2.Text
MyrecRule.Fields("Weight of Almost_Highior") = bb ' TxtAlmost_High2.Text
MyrecRule.Fields("Weight of Mediumiency") = c ' TxtMedium2.Text
MyrecRule.Fields("Weight of Almost_lows") = d ' TxtAlmost_low2.Text
MyrecRule.Fields("Weight of Low") = e ' txtLow2.Text
MyrecRule.Fields("Weight of Reserved_conf") = g ' TxtReserved_conf2.Text
MyrecRule.Fields("Weight of Yin") = 0
MyrecRule.Fields("Weight of Yang") = 0

MyrecRule.Fields("Subconfig_Premise1") = a(1)

MyrecRule.Fields("Subconfig_Premise2") = a(2)

```

MyrecRule.Fields("Subconfig_Premise3") = a(3)

MyrecRule.Fields("Subconfig_Premise4") = a(4)

MyrecRule.Fields("Subconfig_Premise5") = a(5)

MyrecRule.Fields("Subconfig_Premise6") = a(6)

MyrecRule.Fields("Subconfig_Premise7") = a(7)

MyrecRule.Fields("Subconfig_Premise8") = a(8)

MyrecRule.Update

While List1.ListCount > 0

List1.RemoveItem (0)

Wend

For j = 1 To 8

a(j) = ""

b(j) = ""

Next j

TxtHigh2.Text = "0."

TxtAlmost_High2.Text = "0."

TxtMedium2.Text = "0."

TxtAlmost_low2.Text = "0."

txtLow2.Text = "0."

'TxtReserved_conf2.Text = "0."

click = True

End Sub

'=====

'Private Sub Command1_Click0

'Dim i As Integer

'For i = 1 To 37 Step 1

'MyrecRule.MoveFirst

'MyrecRule.Move i - 1

'MyrecRule.Edit

```

'MyrecRule.Fields("Order") = i
'MyrecRule.Update
'Next i
'End Sub

```

```

'Private Sub Command1_Click()
'Text1.Text = a(1)
'Text2.Text = b(1)
'Text3.Text = a(2)
'Text4.Text = b(2)
'Text5.Text = a(3)
'Text6.Text = b(3)

```

```

'End Sub

```

```

Private Sub Form_Load()

```

```

TxtHigh2.Text = "0."
TxtAlmost_High2.Text = "0."
TxtMedium2.Text = "0."
TxtAlmost_low2.Text = "0."
txtLow2.Text = "0."
'TxtReserved_conf2.Text = "0."

```

```

'Update length of Tables

```

```

Myrec.MoveLast

```

```

ItemCPU = Myrec.RecordCount

```

```

MyrecMonitor.MoveLast

```

```

ItemMonitor = MyrecMonitor.RecordCount

```

```

MyrecAlmost_Peripheral.MoveLast

```

```

ItemAlmost_Peripheral = MyrecAlmost_Peripheral.RecordCount

```

```

MyrecHDD_MEM.MoveLast

```



```
ItemHDD_MEM = MyrecHDD_MEM.RecordCount
```

```
'-----
```

```
'=====
```

```
a = Array("", "", "", "", "", "", "", "", "", "", "", "", "", "", "", "", "")
```

```
b = Array("", "", "", "", "", "", "", "", "", "", "", "", "", "", "", "", "")
```

```
Dim j As Integer
```

```
Dim col As Integer
```

```
Dim row As Integer
```

```
Dim i As Integer
```

```
Dim NumberSubconfig_Premise As Integer
```

```
'=====
```

```
' Display title of Table
```

```
MSFlexGrid1.ColWidth(0) = 600
```

```
MSFlexGrid1.ColWidth(1) = 4500
```

```
MSFlexGrid1.col = 0
```

```
MSFlexGrid1.row = 0
```

```
MSFlexGrid1.CellForeColor = &HFF0000
```

```
MSFlexGrid1.Clip = "No."
```

```
MSFlexGrid1.col = 1
```

```
MSFlexGrid1.row = 0
```

```
MSFlexGrid1.CellForeColor = &HFF0000
```

```
MSFlexGrid1.CellFontBold = True
```

```
MSFlexGrid1.CellFontSize = 11
```

```
MSFlexGrid1.Clip = "Configuration"
```

```
'=====
```

```
'Display Subconfig_Premise of CPU
```

```
'Display Name of Group
```

```
col = 0
```

```
row = 1
```

```
MSFlexGrid1.col = col
```

```
MSFlexGrid1.row = row
```

```
MSFlexGrid1.CellFontSize = 11
MSFlexGrid1.CellFontBold = True
MSFlexGrid1.CellForeColor = &HFF0000
MSFlexGrid1.Clip = ""
```

```
MSFlexGrid1.col = col + 1
MSFlexGrid1.row = row
MSFlexGrid1.CellFontSize = 11
MSFlexGrid1.CellFontBold = True
MSFlexGrid1.CellForeColor = &HFF0000
MSFlexGrid1.Clip = "CPU"
```

'-----

'Display Subconfig_Premise of CPU

If ItemCPU > 0 Then

Myrec.MoveFirst

For i = 1 To ItemCPU Step 1

MSFlexGrid1.AddItem i

row = row + 1

MSFlexGrid1.col = col

MSFlexGrid1.row = row

MSFlexGrid1.Clip = Myrec.Fields("Order")

MSFlexGrid1.col = col + 1

MSFlexGrid1.row = row

MSFlexGrid1.Clip = Myrec.Fields("Subconfig_Premise")

Myrec.MoveNext

Next i

End If 'If ItemCPU > 0

'-----

'Display Subconfig_Premise of Monitor

col = 0

row = ItemCPU + 2

'Display name of Group Monitor

```

MSFlexGrid1.col = col
MSFlexGrid1.row = row
MSFlexGrid1.CellFontSize = 11
MSFlexGrid1.CellFontBold = True
MSFlexGrid1.CellForeColor = &HFF0000
MSFlexGrid1.Clip = "***"

MSFlexGrid1.col = col + 1
MSFlexGrid1.row = row
MSFlexGrid1.CellFontSize = 11
MSFlexGrid1.CellFontBold = True
MSFlexGrid1.CellForeColor = &HFF0000
MSFlexGrid1.Clip = "Monitor"

col = 0
row = ItemCPU + 3

If ItemMonitor > 0 Then
    MyrecMonitor.MoveFirst

    For i = 1 To ItemMonitor Step 1
        MSFlexGrid1.AddItem i
        MSFlexGrid1.col = col
        MSFlexGrid1.row = row
        MSFlexGrid1.Clip = MyrecMonitor.Fields("Order")

        MSFlexGrid1.col = col + 1
        MSFlexGrid1.row = row
        MSFlexGrid1.Clip = MyrecMonitor.Fields("Subconfig_Premise")

        MyrecMonitor.MoveNext
        row = row + 1

    Next i

End If ' If ItemMonitor > 0

```

'-----

'Display Subconfig_Premise of Almost_Peripheral

'Display name of Group Almost_Peripheral

col = 0

row = ItemCPU + ItemMonitor + 3

MSFlexGrid1.col = col

MSFlexGrid1.row = row

MSFlexGrid1.CellFontSize = 11

MSFlexGrid1.CellFontBold = True

MSFlexGrid1.CellForeColor = &HFF0000

MSFlexGrid1.Clip = "***"

MSFlexGrid1.col = col + 1

MSFlexGrid1.row = row

MSFlexGrid1.CellFontSize = 11

MSFlexGrid1.CellFontBold = True

MSFlexGrid1.CellForeColor = &HFF0000

MSFlexGrid1.Clip = "Peripheral"

'-----

'Display Subconfig_Premise of Almost_Peripheral

If MyrecAlmost_Peripheral.RecordCount > 0 Then

MyrecAlmost_Peripheral.MoveFirst

row = ItemCPU + ItemMonitor + 4

For i = 1 To ItemAlmost_Peripheral Step 1

MSFlexGrid1.AddItem i

MSFlexGrid1.col = col

MSFlexGrid1.row = row

MSFlexGrid1.Clip = MyrecAlmost_Peripheral.Fields("Order")

MSFlexGrid1.col = col + 1

MSFlexGrid1.row = row

MSFlexGrid1.Clip = MyrecAlmost_Peripheral.Fields("Subconfig_Premise")

MyrecAlmost_Peripheral.MoveNext

row = row + 1

Next i

End If

'-----

'Display Subconfig_Premise of HDD_MEM

'Display name of Group HDD_MEM

col = 0

row = ItemCPU + ItemMonitor + ItemAlmost_Peripheral + 4

MSFlexGrid1.col = col

MSFlexGrid1.row = row

MSFlexGrid1.CellFontSize = 11

MSFlexGrid1.CellFontBold = True

MSFlexGrid1.CellForeColor = &HFF0000

MSFlexGrid1.Clip = "*****"

MSFlexGrid1.col = col + 1

MSFlexGrid1.row = row

MSFlexGrid1.CellFontSize = 11

MSFlexGrid1.CellFontBold = True

MSFlexGrid1.CellForeColor = &HFF0000

MSFlexGrid1.Clip = "Hard Disk and Memory"

'Display Subconfig_Premise of HDD_MEM

If MyrecHDD_MEM.RecordCount > 0 Then

row = ItemCPU + ItemMonitor + ItemAlmost_Peripheral + 5

MyrecHDD_MEM.MoveFirst

For i = 1 To ItemHDD_MEM Step 1

MSFlexGrid1.AddItem i

MSFlexGrid1.col = col

MSFlexGrid1.row = row

MSFlexGrid1.Clip = MyrecHDD_MEM.Fields("Order")

MSFlexGrid1.col = col + 1

```

MSFlexGrid1.row = row
MSFlexGrid1.Clip = MyrecHDD_MEM.Fields("Subconfig_Premise")

MyrecHDD_MEM.MoveNext

row = row + 1

Next i

End If ' If MyrecHDD_MEM.RecordCount > 0

'-----
'Display end of Grid
MSFlexGrid1.col = col
MSFlexGrid1.row = row
MSFlexGrid1.CellForeColor = &HFF0000
MSFlexGrid1.Clip = "===="

MSFlexGrid1.col = col + 1
MSFlexGrid1.row = row
MSFlexGrid1.CellFontSize = 11
MSFlexGrid1.CellFontBold = True
MSFlexGrid1.CellForeColor = &HFF0000
MSFlexGrid1.Clip = "End of Configuration"

MSFlexGrid1.Font.Size = 13 ' Set Font size on all Grid
End Sub

```

