CBR for Tourism Destinations

"TravelSolution System"

Name: Ariati Karina Bt. Khalid

Metric No: WEK010017

Supervisor: En. Mohd. Nor Ridzuan B. Daud

Moderator: Pn. Siti Soraya Bt. Abdul Rahman

Abstract

The present paper will outline the creation and evaluation of a tourism destination advisory system named TravelSolutions. Advisory system became a significant tool in the field of tourism; they offer users a convenient opportunity to find a travel bundle or a single travel item such as accommodation. Existing tourism destination advisory system has some shortcomings as they allow no or only very limited flexibility when taking constraints or preferences into account. Therefore, the aim to design a novel type of advisory system including features of Case-Based Reasoning (CBR) will be pursued. The system enables the selection of travel locations and supports the bundling of personalized travel plan. Travel plans are stored in a memory cases, which exploited for ranking travel items extracted from knowledge base. Knowledge intensive session modeling and mixed initiative advisor are introduced in the CBR framework. The application of these ideas to an interactive, case-based tourism destination advisory system, that guides travelers' decision making processes, is described.

Acknowledgement

In the name of Allah The Almighty.

This proposal is finally completed in the period of time given as friends and family have provided financial, technical and emotional support. It is impossible to list here all those who have helped to sustain me during the writing and revising, and I apologize in advance for any omissions.

This success would not have been accomplished if this project were being done individually without any help and guidance from others. Many thanks to those who had helped me by giving ideas and suggestions during the development of this proposal.

Encik Mohd. Nor Ridzuan B. Daud has been a terrific supervisor, suggesting ways to clarify and simplify, giving support and guidance, and helping me to produce an informative and more efficient proposal. Thank you to Puan Siti Soraya Bt. Abdul Rahman for being the moderator of this project and for giving supports, comments and suggestions for this project.

Finally, to my parents and friends who has given me a lot of support and guidance, Encik Khalid B. Jantan, Puan Aminah Bt Abdul Ghani, Haidayu Anuar, Shahrezat Fadlie, Ahmadil Ardi Ishak and everyone involved in developing this project. Thank you very much.

Disclaimer

List of Figures

List of Tables

Table of Contents

Chapter 1 Introduction

1.1 Problem Statement

There are a continuously growing number of web sites that support a traveler in the selection of a travel destination or a travel service (e.g., flight or hotel). Typically, the user is required to input product constraints or preferences, that are matched by the system in an electronic catalogue. The entire major eCommerce websites dedicated to tourism, such as Expedia, Priceline, TISCover, etc, implement this simple and quite effective pattern. Actually, planning a travel towards a tourism destination is a complex problem solving activity. The term "destination" itself refers to a "fuzzy" concept that lacks a commonly agreed definition. For instance, even the destination spatial extension is known to be a function of the traveler distance from the destination. Italy could be a destination for Japanese, but not for a European traveler who may focus on a specific region, such as Tuscany. Secondly, the "plan" itself may vary greatly, in the structure and content. For instance, some people search for prepackaged solutions (all included) while other "free riders" want to select each single travel detail independently. There is a vast literature investigating: how the travel planning decision process unfolds, the main decision variables and their relationships. Several choice models have been proposed (to quote only a few). These models identify two groups of factors that impact on the destination choice: personal and travel features. In the first group there are both socioeconomic factors (age, education, income, etc.) and psychological/cognitive

(experience, personality, involvement, etc.). In the second group, we could list travel purpose, travel party size, the length of travel, the distance, and the transportation mode.

1.2 General View

This project paper present the result of a project on developing a knowledge-based system using case-based reasoning (CBR) for recommendation of tourism destination. The system is planned to be an interactive one, gradually narrows down the possibilities, eventually deciding the recommended place as tourism destination. CBR commonly known as expert or knowledge-based systems. It differs from the conventional way of how a computer solves a problem by using experience and it can support imprecise fuzzy queries. It also facilitates the handling of voluminous quantities of data. The goal of this project is to demonstrate the feasibility of the application of CBR technology in the areas of tourism.

1.3 Definition

Artificial Intelligence (AI) simulates and applies human cognitive thinking toward problem solving. This project is build under the concepts of developing a method for tourism destination advisory system with the aid of CBR technique where knowledge or cases are used in designing the system. One of the goals of AI is to simulate and applies

human cognitive thinking toward problem solving. Researchers incorporate AI with other disciplines in order to develop automated systems for solving problems in these domains. With a huge array of tourism destination being recognized in Malaysia, particularly, it is a difficult for a person other than travel planners or travel agencies or other experts to recommend and advise a specific destination according to preferences, likes and dislikes given. The problem of existing manual systems is they only give tourism destination information and other information such as attractions, famous restaurants and food places, cultural activities and as such. And it is up to the travelers and tourists to make the decision of the destinations to choose. Keeping the problem in mind, I wanted to develop an automated system, which would encode travel agencies' knowledge on how to identify destinations and recommend it according to tourists' preferences, their personal particulars and others so that the system can be use wisely and easily by anyone. But how to represent various types of destinations in a computer system? What can be categorized into travel items? Different packages will be used in different categories. According to the research discovered that the process of identifying tourism destinations could be compiled in term of cases. Compared to conventional rule-based systems, cases are independent from each other. Maintenance of the CBR system can be done easily by just adding or deleting. Domain experts and novices can understand cases quite easily. For this reason, I chose to represent the destinations identification knowledge with the case-based reasoning knowledge representation scheme.

1.4 Objective

This project sets out to develop and validate an advisory system for tourist destination decision-making. It aims to provide personalized recommendations based on individual users' profile and contextual information. This system will support product aggregation for a given destination in the form of a travel plan. The ultimate goal of this web-based system is to represent an invaluable interactive and conversational tool. The project is set to bring innovative results in the area of Case-Based Reasoning Systems. It will be accessible by means of a PC connection to the Internet. Consequently, the objectives for the development of this system are the following:

- ***To provide easy to use and user-friendly system*** – Not all users are familiar in using computer programs. To encourage the use of computer, the user should be able to use the system within a short period.
- ***To provide alternative ways in getting advice on tourism destinations*** – Prospect clients will be able to eliminate cost of traveling as well as their valuable time by just one click at anywhere and anytime that best suits them.
- ***Applying AI techniques in tourism field*** – For many people, AI is still the stuff of science fiction. But for researchers specializing in case-based reasoning systems technology, a sub area of AI, it is very much reality. This project is build practical knowledge based systems using case-based reasoning. Case-based reasoning systems are computer systems that show intelligence or reasoning capabilities based on the knowledge supplied by domain experts so it works like an expert.

- ***To create paperless working environment*** – No more paper will be needed in recording various types of data. The system will automatically generate the most suitable destinations for clients.

1.5 Scope

Given the wide scope of tourism and case-based reasoning from the research, the project will focus on a narrower perspective mainly in the domain of tourism destinations in Malaysia.

1) System will be able to provide advice and recommendation for suitable places and destinations.
2) It will be highly interactive and it will support a sophisticated "question and answer process" mimicking an interaction with a human expert, or a travel agent.
3) System will store a brief record of users' personal particulars and prior consultation.
4) It will provide support for direct manipulation of data entries, which will boost users' confidence in the supplied recommendations. Advanced users will be enable to explore available information, through an array of access tools, for the selection of their preferred items.

1.6 Motivation Factor

The information overload does not allow the people to take advantage of the new opportunity offered by the large availability of contents. Only to have a reasonably preview of all the alternatives it takes so much time that the process of selection doesn't allow to balance the potential benefits of a custom solution. A second factor that promotes the needs of recommendation services is the personalization of products. A large set of goods can be configuring on the fly to meet the user preferences. This edibility has charged the customer of the configuration task. Even though the personalization is an appealing opportunity not always, the users have enough know-how to perform it successfully. For these reasons, a support is required to help the user finding a satisfactory configuration. Consequently, the motivations for the development of this system are the following:

- Shortage of recognize expert in the area of tourism especially in Malaysia.
- The constraints required decisions to be made with limited or inexact information.

Due to the above factors, there is a real demand for an advisory system that would be able to capture the knowledge of current travel agencies to be able to emulate them.

1.7 System Limitation

The following are the few unavoidable constraints that pose a limit to the system:-

1) Not a Multilanguage system.
2) If a destination were not acceptable by the user, the system would not be able to recommend alternative destinations, taking into account the places that are acceptable and not acceptable by the user.
3) System does not provide reservation of accommodation online with the hotels and resorts, online inquiries and online payment.

1.8 Schedule

For the project, I use Gantt chart to determine a clear timeline between starting date and the finishing date. The important advantage of this chart over other time-charting techniques is its simplicity.

There are six major phases in this project, which are:-

1. Literature review
2. System analysis
3. System design
4. System coding
5. System testing

6. Documentation

Activity	Duration	Start	End	June	July	August	September	October
Literature Review	40 days	4th June 2003	14th July 2003					
System Analysis	40 days	15th July 2003	24th August 2003					
System Design	30 days	25th August 2003	6th October 2003					
System Coding	90 days	7th October 2003	7th February 2003					
System Testing	40 days	15th October 2003	7th February 2003					
Documentation	180 days	3rd June 2003	7th February 2003					

Table 1.1 Gantt chart of Project Schedule

1.9 Report Layout

The following layout is to give an overview of the major phases involved during the development of the project. The purpose of this report is to document all the essential information gathered and used to develop this system. Below is the project layout:-

Chapter 1 Introduction

This chapter serves an introduction to the entire project. In this chapter, I have made an overview on the definition, project objectives, project scopes, project assumptions, motivation factors, project limitations and project schedule.

Chapter 2 Literature Review

The chapter covers all the literatures survey done to the project including reviews on the features, capabilities, system architecture, system-designing tool, and so on.

Chapter 3 Methodology and System Analysis

Chapter 3 fairly discusses the development methodology, the functional and nonfunctional requirements and also the tools and technology considerations of this project.

Chapter 4 System Design

This chapter discusses the design considerations including processing design, user interface design and the case-based design on this project.

Chapter 2 Literature Review

In this chapter, research and literature reviews have been done. It is very important to dig deeper for information about the proposed system. With that, the overview of the system will be seen. The materials are concluded following some efforts such as studying and looking for the similar projects from FCSIT (Faculty of Computer Science and Information Technology) document room, looking for reference books from library, extra reference materials from the Internet and purchasing some useful related books.

There are three parts of literature reviews; approaches, findings and analysis. Firstly, literature review will identify all the approaches that were used to find information for the project and then provide sources and summary of the findings. Finally, the strengths and weaknesses of the information that was found are analyzed.

2.1 Information Source and Finding

Many approaches to find information in case-based reasoning (CBR) and travel and tourism destination had been utilized. The information available on CBR is not that much but it is different for the available information on travel and tourism destination, which is so much as it, has become one of the biggest factors towards contribution to the world and Malaysia economy, particularly. Many researches and development have also been

done in most countries in the world. My finding sources are generally divided into few categories:

- Internet
- Article from Medias
- Interviewing staffs from local travel agencies
- Questionnaire method
- Others

2.1.1 Internet

Internet plays the major role in finding all the information about CBR, travel and tourism destination, knowledge-based system and others. Internet can provide me up-to-date information and this is important because we are in the technology world that everything has kept updated. There are plenty of CBR researches available in the Internet and CBR applications are very wide indeed. As a research, I have downloaded many articles about CBR developing system in order to better understanding of CBR.

2.1.2 Articles from Medias

- Newspapers
- In-tech from Star

- Compute Time from News Straits Time

2.1.3 Interviewing Staffs from Local Travel Agencies

The purpose of conducting interviews with local travel agencies is to obtain some useful information for the purposed system. The information that I gathered may help me to understand on what kind of services is required by the customers. Besides, I can also know the common complaints from the customers. From the interviews conducted with the staffs from local travel agencies, I was able to know their marketing strategy. The questions for the interviews are listed as below:-

1) What are the services provided by a travel agency?
2) What is the information required by the travelers in Malaysia?
3) What is the field of interest for the most travelers in Malaysia (for example, place of interest, accommodation)?
4) What are the customers mostly complaint from the travel agency?
5) Does the travel agency provide tourist guide to the travelers?
6) Does the travel agency has its own web sites? If yes, what are the features in the web site? If no, are they interest to publish their travel agency online?

There are several important points that can be summarized for the interview, that are:

1. **Services and information required by the customers**

The customers normally prefer to let the travel agency to settle everything for them after they had decided to join certain trip. The services provided by the travel agency are hotel

reservation , air ticket reservation ,car rental, provide useful tourist guide, manage the travel document and so on. The customers will feel convenient when the travel agency provide all services they needed during the trip .Normally, local travelers are more interested in placed of interest in Malaysia whole foreign travelers would like to know more about the culture ,food, festival, as well as the natural and historical places of interest in Malaysia.

2. **Complaint from customers**

The common complaints from the customers are lack of information about the trip, lack of customer service, difficult to reach to the person in charge and so on. Customers will also like to bargain with the travel agency's staffs because they think the prices are not reasonable. However, the staffs will help the travelers to obtain a ideal trip within their budget.

3. **Marketing strategy**

The travel agencies that I had interviewed do not use web-based travel information system. They are doubtful on the efficiency of web-based travel information system to increase the company's sales. Moreover, they do not have much exposure in the information technology field. However, they are interest to publish their travel agency online and may do so in the future.

2.1.4 Questionnaire Method

The purpose of using questionnaire method is to obtain information from a number of people from different area. The general opinion and suggestion from the public are very useful for developing the system. The questions are well designed and well structured in order to get some adequate and useful information. I used the close-ended questions as well as the open-ended questions in designing the questionnaire.

Close-ended questions are designed so that I can identify the answers from the respondents easily as I can expect the answers given by them. As the number of respondents involved is quite a lot, it is hard to analyze the data if all the questions ae designed in an open-ended form. Besides, respondents do not have to spend a lot of time to answer close-ended questions.

Open-ended questions are designed when I expect a wide scope of answer from the respondents. Besides, some questions are designed to obtain opinion and suggestion from the respondents.

The questionnaire is attached on the Appendix.

The result of the collected questionnaire indicated that most of the people especially Internet users preferred a web-based travel information for a trip. They would like to enjoy the services such as analysis on hotels, online reservation, online inquiries, online payment, tourist consultant, view transport schedule, and so on.

2.1.5 Others

Studies have been done through similar works made of seniors in the FCSIT document room. Although there are no exactly similar work has been found, but I study about how they express CBR for better understand on my CBR applications. Beside that, I had found more CBR application reference website from their work.

2.2 Artificial Intelligence (AI)

AI is a combination of computer science, physiology and philosophy. AI is a broad topic, consisting of different tools, from machine vision to expert systems. The element that the fields of AI have in common is the creation of machines that can "think".

Research into the areas of learning, language, and of sensory perception has aided scientists in building intelligent machines. One of the most challenging approaches facing experts is building systems that mimic the behavior of the human brain, made up of billions of neurons, and arguably the most complex matter in the universe. Perhaps the best way to gauge the intelligence of a machine is British computer scientist, Alan Turing's Test. He stated that a computer would deserve to call intelligent if it could deceive a human into believing that it was human.

Artificial Intelligence has come a long way from its early roots, driven by dedicated researchers. The beginning of AI can reach back before electronics to philosophers and mathematicians such as Boole and others theorizing on principles that were used as the function of AI Logic. AI really began to intrigue researchers with the invention of the computer in 1943. The technology was finally available, or so it seemed, to stimulate intelligent behavior. Over the next four decades, despite many obstacles, AI has grown from a dozen researchers, to thousands of engineers and specialists; and from programs capable of playing checkers, to systems designed to diagnose disease.

AI has always been on the pioneering end of computer science. Advanced-level computer languages, as well as computer interfaces and word processors owe their existence to the research into artificial intelligence. The theory and insights brought about by AI research will set the trend in the future of computing. The products available today are only bits and pieces of what are soon to follow, but there are movements towards the future of Artificial Intelligence.

2.2.1 Introduction to Artificial Intelligence

AI can be broken into several different disciplines. They are each unique, but often they intermix to accomplish a programming tool, and the differences become fuzzy. The different disciplines are expert systems, natural languages, simulation of human sensory capabilities, robotics, and neural networks.

Expert Systems: Expert systems are also known as knowledge-based systems. They are computer systems that rely on a knowledge base of rules that pertain to a specific application area. Experts in the application area give rules of thumb to use in certain situations. The rules of thumb are linked into preset if-then rules to solve the problem. Essentially, the system mimics the expert's thought process in troubleshooting the problem. Expert systems are able to solve problems. Solving problems is their intention. They are able to break a large problem into smaller parts in order to come to a solution. They understand the information given to them by the user. If the information is contradictory or ambiguous, the expert system asks for clarification or more information. Expert systems can learn. If an expert system is facing with a new problem, it will save the information and the solution the user chooses for future use.

Natural Languages Systems: Natural language systems are systems that enable computers to except, interpret, and execute instructions in the natural language of the user. The intent of natural language systems is to create a more natural interaction between human users and computers. Database query is the area that has benefited the most from natural language systems. Other areas appropriate for natural language include machine translation, summarizing and searching bibliographic texts, and analyzing style and sentence structure.

The biggest contribution to the evolution of machine intelligence by natural language and conversation systems is in the introduction between humans and machines.

Human Sensory Simulation: Simulation of human sensory capabilities focuses on seeing, hearing, speaking, and touching. I will focus on two systems: voice recognition and vision input systems.

- **Voice recognition systems** – Allow the user to speak to the computer. Voice recognition systems are used to enable computers to see and understand its environment.
- **Vision input systems** – are still in their infancy. The storage space and processing speed needed for a fully functional vision input system are too restrictive. The human vision process is extremely complex. One commercial use of vision input systems is in assembly line inspection robots. This type of system will have a camera placed over the assembly line to inspect parts for defects. The digitized camera image is compared to an image in a database. If a defect is found, the system alerts the main assembly computer to bring it to the human operator's attention, and appropriate action can be taken.

Robotics: Robotics is the integration of computers and robots. Robots are taught to perform repetitive tasks. Intelligent robots incorporate the other disciplines of AI. They use human sensory simulation for touch, sight and hearing. A useful robot in use is a security robot for warehouses. They are given an internal map of the area they are to patrol. They listen for abnormal sounds, and look for intruders or fire. If an abnormality is found, they contact the authorities and employees. Another useful robot collects aluminum cans and thrash in office buildings after everyone has left for the day.

The basis of a robot's intelligence is in its programming. A knowledge base may be used for the robot to figure out what to do if it finds a fire. They must be aware of their surroundings and where they are in their surroundings. They will faithfully sense their particular purpose.

Neural Networks: Neural networks are a fairly new addition to the field of artificial intelligence. Neural networks are a simulation of the processes of the human brain. As Rao (1995) explain, "The human brain uses a web of interconnected processing elements called neurons to process information."

Simply, neural networks are a system of interconnected computing elements. The individual elements are data objects commonly referred to as nodes or neurons. A weighted signal is sent into the network. Each node has only one output based on its input. The input spreads through the network forming a pattern. Information is stored as the pattern of interconnection that is formed between the nodes.

The behavior of a neural network is more human than conventional computer systems. Rosenfield states, "They learn, forget and most importantly, they can be structured so that they self-organize to represent information and solve problems." (1995). Neural networks are not very good at solving simple problems. They are best at solving complex problems that cannot be solved by a simple algorithm. They are also very good at solving problem whose input is noisy. Neural networks can be built to solve specific problems, but they are most useful for their ability to learn. A string of inputs is introduced to the network along with the desired result. The network will form the solution. For this reason neural networks are the masters of pattern recognition.

Incorporate fuzzy logic and this ability is increased ten fold.

The major fields of interest for neural networks make use of the neural networks ability to filter noise and recognize patterns. These fields are handwriting and speech recognition, and predicting stock patterns. Neural networks are also found in fields such as finance, reading IRS tax forms, defense systems, and vehicle control.

2.2.2 The History of Artificial Intelligence

Evidence of Artificial Intelligence folklore can be traced back to ancient Egypt, but with the development of the electronic computer in 1941, the technology finally became available to create machine intelligence. The term artificial intelligence was first coined in 1956, at the Dartmouth conference, and since then AI has expanded because of the theories and principles developed by its dedicated researchers. Through this short modern history, advancement in the fields of AI have been slower than first estimated, progress continues to be made. From its birth four decades ago, there have been a variety of AI programs, and they have impacted each other technological

advancements.

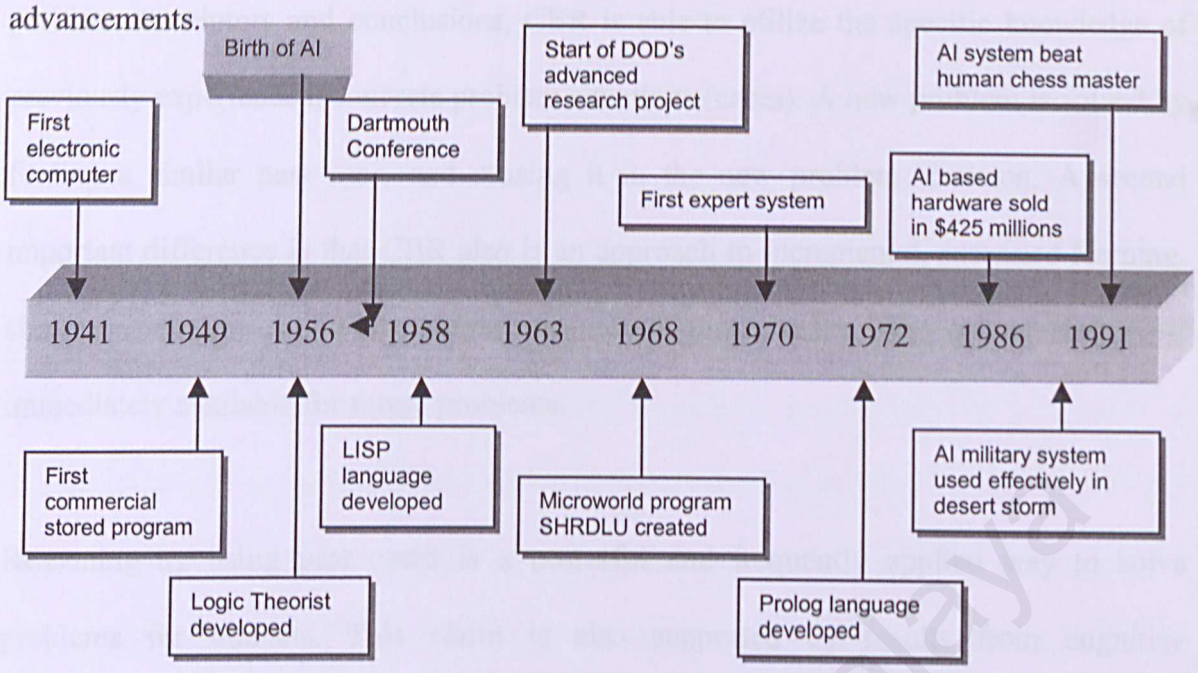


Figure 2.1 Timeline of major AI event

2.3 Case-Based Reasoning (CBR)

2.3.1 Introduction

Over the last few years, case-based reasoning (CBR) has grown from a rather specific and isolated research area to a field of widespread interest, as seen by its increased share of papers at major conferences, available commercial tools, and successful applications in daily use. CBR is a problem-solving paradigm that in many respects is fundamentally different from other major AI approaches. Instead of relying solely on general knowledge of a problem domain, or making associations along generalized relationships between

problem descriptors and conclusions, CBR is able to utilize the specific knowledge of previously experienced, concrete problem situations (cases). A new problem is solved by finding a similar past case, and reusing it in the new problem situation. A second important difference is that CBR also is an approach to incremental, sustained learning, since a new experienced is retained each time a problem has been solved, making it immediately available for future problems.

Reasoning by using past cases is a powerful and frequently applied way to solve problems for humans. This claim is also supported by results from cognitive psychological research. Part of the foundation for the case-based approach, is its psychological plausibility. Several studies have given empirical evidence for the role of specific, previously experienced situations (what we call cases) in human problem solving developed a theory of learning and reminding based on retaining of experience in a dynamic, evolving memory structure. People use past cases as models when learning to solve problems, particularly in early learning. Other results indicate that the use of past cases is a predominant problem solving method among experts a s well.

In CBR terminology, a *case* usually denotes a *problem situation*. A previously experienced situation, which has been captured and learned in a way that it can be reused in the solving of future problems, is referred to as a past case, previous case, stored case, or retained case. Correspondingly, a new case or unsolved case is the description of a new problem to be solved. Case-based reasoning is – in effect - a cyclic and integrated process of solving problem, learning from this experience, solving a new problem, etc.

A very important feature of case-based reasoning is its coupling to learning. The driving force behind case-based methods has to be a large extent come from the machine learning community, and case-based reasoning is also regarded a subfield of machine learning. Thus, the notion of CBR does not only denote a particular reasoning method, irrespective of how the cases are acquired, it also denotes a machine learning paradigm that enables sustained learning by updating the case base after a problem has been solved. Learning in CBR occurs as a natural by-product of problem solving. When a problem is successfully solved, the experience is retained in order to solve similar problems in the future. When an attempt to solve a problem fails, the reason for the failure is identified and remembered in order to avoid the same mistake in the future.

Case-based reasoning favors learning from experience, since it is usually easier to learn by retaining a concrete problem solving experience than to generalize from it. Still, effective learning in CBR requires a well worked out set of methods in order to extract relevant knowledge from the experience, integrate a case into an existing knowledge structure, and index the case for later matching with similar cases.

2.3.2 History of CBR

The word Schank and Abelson in 1977 is widely held to be the origins of CBR. They proposed that our general knowledge about situations is recorded as scripts that allow us

to set up expectations and perform inferences. Scripts were proposed as a structure for conceptual memory describing information about stereotypical events such as, going to a restaurant or visiting a doctor. However, experiments on scripts showed that they were not a complete theory of memory representation because people often confused events that had similar scripts. For example, a person might mix up room scenes from a visit to a doctor's office with a dentist's office. Such observations fell in line with the theories of concept formation, problem solving and experiential learning within philosophy and psychology.

Roger Schank continued to explore the role that the memory of previous situations (i.e. cases) and situation patterns or memory organization packets (MOPs) play both in problem solving and learning. At a similar time Gentler was developing a theoretical framework for analogy which also has relevance to CBR. Perhaps with the benefit with hindsight it is possible to find references of significance to CBR in Wittgenstein's observation that natural concepts such as tables and chairs are in fact polymorphic and can not be classified by a single set of instances (i.e. cases) with family resemblances. This work has been cited by Aamodt and Plaza [94] as a philosophical basis for CBR.

Whilst the philosophical roots of CBR could perhaps be claimed by many what is not in doubt is that it was the work of Roger Schank's group at Yale University in the early eighties that produced both a cognitive model for CBR and the first CBR applications based upon this model. Janet Kolodner developed the first CBR system called CYRUS. CYRUS contained knowledge, as cases, of the travels and meetings of ex-US-Secretary-

of-State Cyrus Vance. CYRUS was an implementation of Schank's dynamic memory model. Its case-memory model later served as the basis for several other CBR systems including MEDIATOR, CHEF, PERSUADER, CASEY, and JULIA.

An alternative approach came from Bruce Porter's work, at the University of Texas in Austin, into heuristic classification and machine learning resulting in the PROTOS system. PROTOS unified general domain knowledge and specific case knowledge into a single case memory model. GREBE, a system operating in legal domain, took this work further.

It is perhaps so surprise that since the practice of law is largely based upon precedence and the notion of cases, that there has been some interest from this sector in CBR. Edwine Rissland is a group in the University of Massachusetts in Amherst who developed HYPO. In HYPO cases representing legal precedents are used to interpret a court situation and to produce arguments for both the defenses and the prosecutors. This system was later combined with rule-based reasoning to produce CABARET.

CBR research is not restricted to the US, but it was slower to get started in Europe. Amongst the first cited European work is that of Derek Sleeman's group from Aberdeen in Scotland. They studied the use of cases for knowledge acquisition, developing the REFINER system. At a similar time Mike Keane, from Trinity College Dublin, undertook cognitive science research into analogical reasoning that has subsequently influenced CBR.

On continental Europe Michael Richter and Klaus Althoff from the University of Kaiserslautern applied CBR to complex diagnosis. This has given rise to the PATDEX system and subsequently to the CBR tool, S3-Case. Agner Aamodt at the University of Trondheim has investigated the learning facet of CBR and the combination of cases and general domain knowledge resulting in CREEK.

In the UK, CBR seems to be particularly applied to civil engineering. A group at the University of Salford is applying CBR techniques to fault diagnosis, repair and refurbishment of buildings. Yang and Robertson in Edinburgh are developing a CBR system for interpreting building regulations, a domain reliant upon the concept of precedence. Whilst another group in Wales applying CBR to the design of motorway bridges.

Further a field there is active CBR groups in Israel, India and Japan. However, the increasing number of CBR papers in AI journals and the increasing number of commercially successful CBR applications is likely to ensure that many more countries take an active interest in CBR in the future. As an indicator the British Computer Society Specialist Group on Expert Systems has held CBR workshops suitable for novices at both it last conference.

2.3.3 How CBR Work?

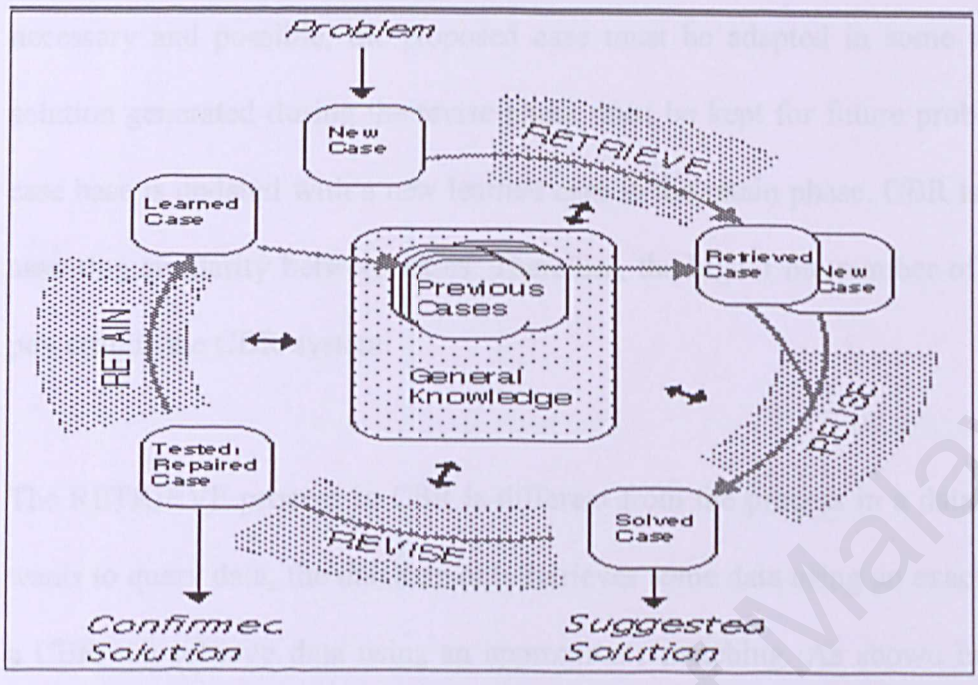


Figure 2.2 CBR Cycle (Aamodt and Plaza, 1994, AI Communication)

The CBR process can be represent by a schematic cycle, as shown in Figure 2.2. Conceptually, the CBR solves problems following the four phases (Aamodt and Plaza, 1994):

- Retrieve the most similar case(s)
- Reuse the case(s) to attempt to solve the problem
- Revise the proposed solution if necessary
- Retain the new solution as a part of a new case

During retrieval the most similar case or cases in the case base are determined, based on the new problem description. During reuse the information and knowledge in the retrieved case(s) is used to solve the new problem. The new problem description is

combined with the information contained in the old case to form a solved case. During revision the applicability of the proposed solution is evaluated in the real world. If necessary and possible, the proposed case must be adapted in some way. If the case solution generated during the revise phase must be kept for future problem solving, the case base is updated with a new learned case in the retain phase. CBR is concerned with assessing similarity between cases. Therefore, the higher the number of cases, the more powerful is the CBR system.

The RETRIEVE process in CBR is different from the process in a database. When user wants to query data, the database only retrieves some data using an exact matching while a CBR can retrieve data using an approximate matching. As shown in Figure 2.2, the CBR cycle starts with the description of a new problem, which can be solved by retrieving previous cases and reusing solved cases, if possible, giving a suggested solution or revising the solution, retaining the repaired case and incorporating it into the case base. However, this cycle rarely occurs without human intervention that is usually involved in many RETAIN step. Many application systems and tools act as a case retrieval system, such as some help desk systems and customer support systems.

CBR systems store knowledge in four different knowledge containers (Richter, 1995):-

- Vocabulary
- Case base
- Similarity assessment
- Solution adaptation

The following describe the CBR systems in more details:

Case representation

A case is a contextualized piece of knowledge representing an experience. It contains the past lesson that is the content of the case and the context in which the lesson can be used. Typically, a case comprises:

- Problem that describes the state of the world when the case occurred
- Solution, which states the derived solution to that problem
- Outcome that describes the state of the world after the case occurred

Cases, which comprise problems and their solutions, can be used to derive solutions to new problems. Cases comprising problems and outcomes can be used to evaluate new situations. If, in addition, such cases contain solutions they can be used to evaluate the outcome of proposed solutions and prevent potential. Cases can be represented in a variety of forms using full range of AI representational formalism is including frames, objects, predicates, semantic nets, and values – the frame/object representation currently being used by the majority of CBR software.

There is a lack of consensus within the CBR community as to exactly what information should be in a case. However, two pragmatic measures can be taken into account in deciding what should be represented in cases: the functionality and the ease of acquisition of the information represented in the case.

Case Indexing

Case indexing involves assigning indices to cases to facilitate their retrieval. Several guidelines on indexing have been proposed by CBR researchers. Indices should be:

- Predictive
- Address the purposes the case will be used
- Abstract enough to allow for widening the future use of the case base
- Concrete enough to be recognized in future

Both manual and automated methods have been used to select indices. Choosing indices manually involves deciding a case's purpose with respect to the aims of the reasoned and deciding under what circumstances the case will be useful. There are an ever increasing number of automated indexing methods including:

- Indexing cases by features and dimensions that tend to be predictive across the centre domain that is, descriptors of the case which are responsible for solving it or which influence its outcome. In this method the domain is analyzed and the dimensions that tend to be important are computed. These are put in a checklist and all cases are indexed by their values along these dimensions.
- Difference-based indexing selects indices that differentiate a case from other cases. During this process, the system discovers which features of a case differentiate a case from other similar cases, choosing as indices those features that differentiate cases best.
- Similarity and explanation-based generalization methods, which produce an appropriate set of indices for abstract cases created from cases that share some

common set of features, whilst the unshared features are used as indices to the original cases.

- Inductive learning methods, which identify, predictive features that are then used as indices.
- Explanation-based techniques, which determine relevant features for each case. This method analyses each case to find which of their features predictive ones are. Those features then indexed cases.

However, despite the success of many automated methods, Janet Kolodner believes that people tend to do better at choosing indices than algorithms, and therefore for practical applications indices should be chosen by hand.

Case Storage

Case storage is an important aspect in designing efficient CBR systems in that, it should reflect the conceptual view of what is represented in the case and take into account the indices that characterize the case. The case base should be organized into a manageable structure that supports efficient search and retrieval methods. A balance has to be found between storing methods that preserve the semantic richness of cases and their indices and methods that simplify the access and retrieval of relevant cases. These methods are usually referred to as **case memory models**. The two most influential case memory models are the **dynamic memory model** and the **category-exemplar model**.

➢ **The Dynamic Memory Model**

The dynamic memory model in this method is comprised of memory organization packets (MOPs). MOPs are a form of frame and are the basic unit in dynamic

memory. They can be used to represent knowledge about classes of events using two kinds of MOPs:

- Instances representing cases, events or objects
- Abstractions representing generalized versions of instances or of other abstractions

The case memory, in a dynamic memory model, is a hierarchical structure of episodic memory organization packets (E-MOPs), also referred to as generalized episode (GEs) developed from Schank's more general MOP theory. The basic idea is to organize specific cases which share similar properties under a more general structure. A GE contains three different types of objects: *norms, cases* and *indices*. Norms are features common to all cases indexed under a GE. Indices are features which discriminate between a GE's cases. An index may point to a more specific generalized episode or to a case, and is composed of an index name and an index value.

The case-memory is a discrimination network where nodes are either a GE, an index name, index value or a case. Index-name value pairs point from a GE to another GE or case. The primary role of a GE is as an indexing structure for storing, matching and retrieval of cases. During case storage when a feature (i.e. index name and index value) of a new case matches a feature of an existing case a new GE is created. The two cases are then discriminated by indexing them under different indices below the new GE (assuming cases are not identical). Thus, the memory is dynamic in that similar parts of two cases are dynamically generalized into a new GE, the cases being indexed under the GE by their differences.

However, this process can lead to an explosive growth in the number of indices as case numbers increase. So for practical purposes most CBR systems using this method limit the number of permissible indices to a limited vocabulary.

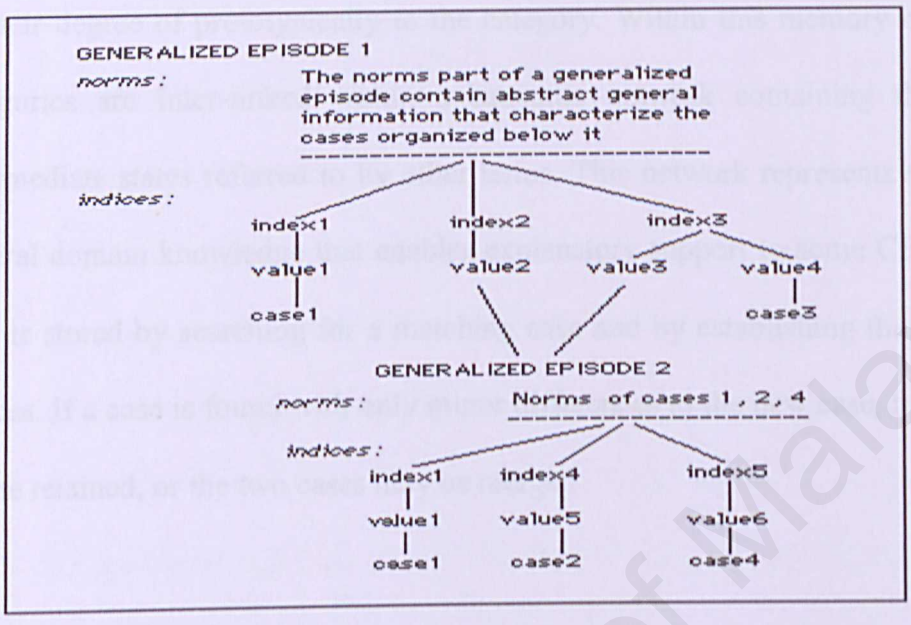


Figure 2.3 The Dynamic Memory Model

➢ ***The Category-Exemplar Model***

This model organizes cases based on the view that the real world should be defined extentionally with cases being referred to as exemplars. The case memory is a network structure of categories, semantic relations, cases and index pointers. Each case is associated with a category. Different case features are assigned different importance in describing a case's membership to a category. Three types of indices are provided, which may point to a case or category:

- *Feature links* – point from problem descriptors (features) to a case or category
- *Case links* – point from categories to its associated cases, and

- *Different links* – pointing from categories to the neighboring cases that only differ in a small number of features

A feature is described by a name-value pair. A category's exemplars are stored according to their degree of prototypically to the category. Within this memory organization, the categories are inter-linked within a semantic network containing the features and intermediate states referred to by other terms. This network represents a background of general domain knowledge that enables explanatory support to some CBR tasks. A new case is stored by searching for a matching case and by establishing the relevant feature indices. If a case is found with only minor differences to the new case, the new case may not be retained, or the two cases may be merged

Retrieval

Given a description of a problem, a retrieval algorithm, using the indices in the case-memory, should retrieve the most similar cases to the current problem or situation. The retrieval algorithm relies on the indices and the organization of the memory to direct the search to potentially useful cases. The issue of choosing the *best* matching case has been addressed by research into analogy. This approach involves using heuristics to constrain and direct the search. Several algorithms have been implemented to retrieve appropriate cases, for example: serial search, hierarchical search and simulated parallel search.

Case-based reasoning will be ready for large-scale problems only when retrieval algorithm s are efficient at handling thousands of cases. Unlike database searchers that

target a specific value in a record, retrieval of cases from the case base must be equipped with heuristics and that perform partial matches, since in general there is no existing case that exactly matches the new case. The following are well known methods for case retrieval:

- Nearest neighbor
- Induction
- Knowledge guided induction
- Template retrieval

These methods can be used alone or combined into hybrid retrieval strategies.

i. Nearest Neighbor

This approach involves the assessment of similarity between stored cases and the new input case, based on matching a weighted sum of features. The biggest problem here is to determine the weights of the features. The limitation of this approach includes problems in converging on the correct solution and retrieval times. In general the use of this methods leads to the retrieval time increasing linearly with the number of cases. Therefore, this approach is more effective when the case base is relatively small.

ii. Induction

Induction algorithms determine which features do the best job in discriminating cases, and generate a decision tree type structure to organize the cases in memory. This

approach is useful when a single case feature is required as a solution, and where that case feature is dependent upon others.

iii. Knowledge guided induction

This method applies knowledge to the induction process by manually identifying case features that are known or thought to affect the primary case feature. This approach is frequently used in conjunctions with other techniques, because the explanatory knowledge is not always readily available for large case bases.

iv. Template retrieval

Similar to SQL-like queries, template retrieval returns all cases that fit within certain parameters. This technique often used before other techniques, such as nearest neighbor, to limit the search space to a relevant section of the case base.

Adaptation

Once a matching case is retrieved, a CBR system should adapt the solution stored in the retrieved case to the needs of the current case. Adaptation looks for prominent differences between the retrieved case and the current case and then applies formula or rules that take those differences into account when suggesting a solution. In general, there are two kinds of adaptation in CBR:

- **Structural adaptation** – adaptation rules are applied directly to the solution stored in cases.
- **Derivational adaptation** – reuses the algorithms, methods or rules that generated the original solution to produce a new solution to the current problem. In this method the planning sequence that constructed the original solution must be stored in memory along with the solution. Derivational adaptation, sometimes referred to a reinstantiation, can only be used for cases that are well understood.

An ideal set of adaptation rules must be strong enough to generate complete solutions from scratch, and an efficient CBR system may need both structural adaptation rules to adapt poorly understood solutions and derivational mechanisms to adapt solutions of cases that are well understood. Several techniques, ranging from simple to complex, have been used in CBR for adaptation. These include:

- ***Null adaptation*** – A direct simple technique that applies whatever solution is retrieved to the current problem without adapting it. Null adaptation is useful for problems involving complex reasoning but with a simple solution.
- ***Parameter adjustment*** – A structural adaptation technique that compares specified parameters of the retrieved and current case to modify the solution in an appropriate direction.
- ***Abstraction and respecialisation*** – A general structural adaptation technique that is used in a basic way to achieve simple adaptations and in a complex way to generate novel, creative solutions.

- ***Critic-based adaptation*** – In which a critic looks for contributions of features that can cause a problem in a solution. Importantly, the critic is aware of repairs for these problems.
- ***Reinstantiation*** – It is used to instantiate features of an old situation with new features.
- ***Derivational replay*** – It is the process of using the method of deriving an old situation or solution piece to derive a solution in the new situation.
- ***Model-guided repair*** – Uses a casual model to guide adaptation, which is used for diagnosis and learning in auto machines, and used in the design of physical devices.
- ***Case-based situation*** – Uses cases to suggest solution adaptation.

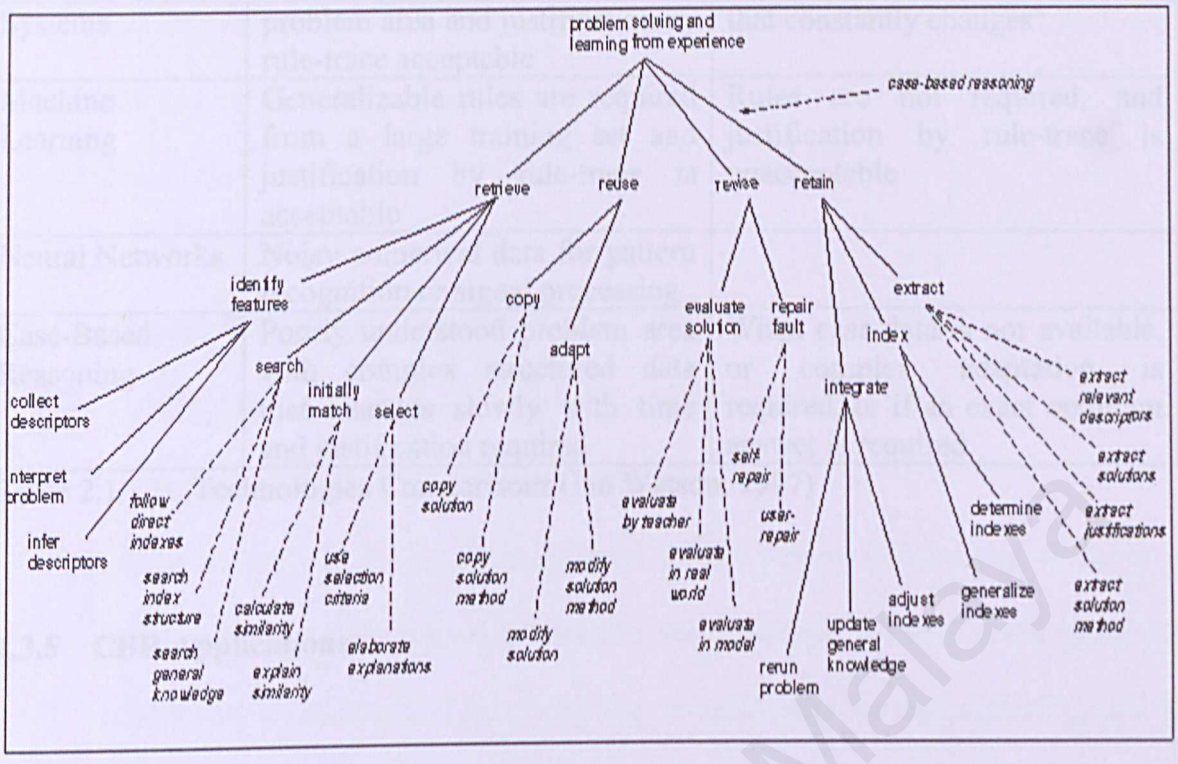


Figure 2.4 A Task-Method Decomposition of CBR

2.3.4 CBR and Other Techniques

Every type of technology has got advantages and disadvantages. The table below shows the appropriate time in using different forms of systems.

Technology Type	When to use	When not to use
Databases	Well-structured, standardized data and simple precise queries possible	Complex, poorly structured data and fuzzy queries required
Information Retrieval	Large volumes of textual data	Non-textual complex data types background knowledge available
Statistics	Large volumes of well-understood data with a well-formed hypothesis	Exploratory analysis of data with dependent variable
Rule-based	Well-understood, stable, narrow	Poorly understood problem area

Systems	problem area and justification by rule-trace acceptable	that constantly changes
Machine Learning	Generalizable rules are required from a large training set and justification by rule-trace is acceptable	Rules are not required, and justification by rule-trace is unacceptable
Neural Networks	Noisy numerical data for pattern recognition or signal processing	
Case-Based Reasoning	Poorly understood problem area with complex structured data that changes slowly with time and justification required	When case data is not available, or complex adaptation is required, or if an exact optimum answer is required

Table 2.1 Technologies Comparisons (Ian Watson, 1997)

2.3.5 CBR Applications

The applications of CBR are divided into two sub-sections. The first CBR applications are primarily the product of academic research. These systems, mostly developed in the US, demonstrate certain features of CBR. They were viewed as demonstrators, as there is little evidence they have been used in real situations. Although CBR is a relatively new method, there have been several successful commercial applications. The second sub-section discusses three such applications. The first developed at Lockheed is usually cited as the first commercial CBR application. Lockheed's CLAVIER system has already become the classic CBR system demonstrating that CBR could be applied to solve a problem where no explicit model existed, and can learn by acquiring new cases and to improve their performance with time.

The second developed by British Airways shows how CBR techniques are easily accepted by users because of CBR's intuitive feel and how retaining the rich context of a case has advantages over the distilled knowledge associated with rule-base systems. The

final system developed for Legal and General shows how CBR can be implemented rapidly and can be maintained by users.

2.3.5.1 Classification of CBR Applications

CBR applications can be classified into two categories:

- ***Analytic tasks*** – Focusing on analyzing situation where classification on the new situation always involved. A new case is matched against those in the case base from which an answer can be given. The solution from the best matching case is then reused. In fact, most commercial CBR tools support analytic tasks.
- ***Synthesis tasks*** – Attempt to get a new solution by combining previous solutions and there are a variety of constraints during synthesis. Usually, they are harder to implement. CBR systems that perform synthesis tasks must make use of adaptation and are usually hybrid systems combining CBR with other techniques [Watson, 1997]. Synthesis task usually will infinite (or at least very large) solution space.

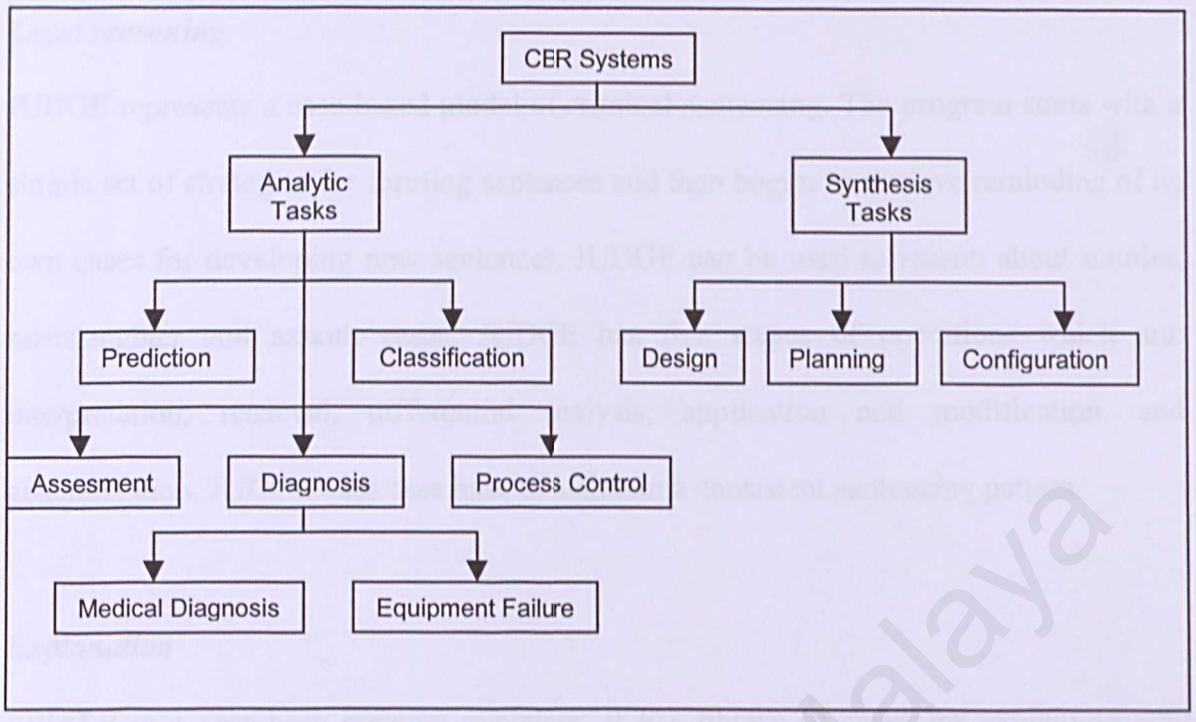


Figure 2.5 A Classification Hierarchy of CBR Applications [Althofff et. El., 1997]

2.3.5.2 Implementation of Academic Research

The following is an indication of the range of problems to which CBR has been applied:

Knowledge acquisition

The REFINER program is a knowledge acquisition system that helps an expert refine his knowledge in a more natural way than extracting rules from an expert. The system has the ability to recognize classifications that are inconsistent and to suggest ways of resolving inconsistency. REFINER uses both inferred rules as well as individual cases.

Legal reasoning

JUDGE represents a case-based model of criminal sentencing. The program starts with a simple set of strategies for forming sentences and then begins to retrieve reminding of its own cases for developing new sentences. JUDGE can be used to reason about murder, manslaughter and assault cases. JUDGE has five stages of operations which are: interpretation, retrieval, differential analysis, application and modification, and generalization. JUDGE uses case base to maintain a consistent sentencing pattern.

Explanation

SWALE is a case-base creative explainer. It has library of cases for explaining why animals and people die. If SWALE is given an anomalous event such as the death of a healthy, young horse, it searches for explanations of death in other context such as rock stars dying from drug overdoses or a spouse being murdered for life insurance, and then SWALE tries to adapt those explanations to fit the current situation.

Diagnosis

CASEY is a system which diagnosis heart failure. As input it uses a patient's symptoms and produces a causal network of possible internal states that could lead to those symptoms. When a new case arises CASEY tries to find cases of patients with similar but not necessarily identical symptoms. If the new case matches, then CASEY adapts the

retrieved diagnosis by considering differences in symptoms between the old and new cases.

Arbitration

MEDIATOR works in the domain of dispute resolution. Given a conflict between several parties, it proposes possible compromises. If one proposal fails to satisfy all the parties involved, it generates new proposals and records the failure for avoiding a similar failure in the future.

Design

JULIA is a case-based designer that works in the domain of meal planning. It uses cases to propose solutions, decomposing the problem as necessary and posting constraints to guide synthesis. It exploits a repertoire of adaptation methods to transform previous dishes in order to meet the constraints of the current problem. These adaptation methods are used in both to modify previous cases and repair previous decisions that have been invalidated by constraints that arrive late.

CADET is a case-based design system that functions as designer's assistant for mechanical design. It retrieves previous successful designs while avoiding previous failures such as poor materials or high costs. CADET transforms abstract descriptors of the derived behavior of the device into a description that can be used to retrieve relevant designs and generate a variety of equivalent alternative designs for a given set of design specifications.

Planning

BATTLE projects the results for plan evaluation in the domain of land warfare planning. The system was built from an existing database of 600 cases. The user describes a battle situation and chosen battle plan. BATTLE retrieves cases that are composed of pieces of battles and experts' evaluations, to project the outcome.

Repair and adaptation

CHEF creates new recipes from old ones. CHEF begins planning by finding a recipe that satisfies as many of its active goals as possible. It uses a set of object critics and modification rules to change the old recipes and satisfy the goal of the new one. One of the important aspects of CHEF is explanation of failures through a causal description of why they occurred. CHEF links recipes to a failure's explanation to predict the failure in similar circumstances. CHEF stores new recipes indexed by the goals that they satisfy and the problems that they avoid.

2.3.6 Advantages of CBR

CBR provides a number of advantages over alternative approaches:

- Does not require extensive analysis of domain knowledge. CBR permits problem solving even if domain knowledge is incomplete. The most important thing is to know how to compare two cases.
- Allows shortcuts in reasoning. If a suitable case is found, a solution can be proposed quickly.

- Lead to improved explanation capability in situations where the most comprehensible explanations are those that involve specific instances.
- Help avoid past errors and learn from the errors and success. In CBR, the system keeps a record of each situation that occurred for future reference.
- High user acceptance.
- Make use of existing data, e.g. Database.
- Reduce the knowledge acquisition effort. In traditional knowledge-based system, acquisition of general knowledge is difficult. CBR system requires less general knowledge because most general knowledge is in case base. Case knowledge is easier to acquire.
- Required less maintenance effort. This is because cases are independent from each other. Furthermore, domain expert and novices understand cases quite easy. Maintenance of CBR system can be done by just adding and deleting cases.

2.3.7 Disadvantages of CBR

- A case-based reasoner might be tempted to use old cases blindly, relying on previous experience without validating it in the new situation.
- A case-based reasoner might allow cases to bias him or her or it too much in solving a new problem.
- Often people, especially novices, are not reminded of the most appropriate sets of cases when they are reasoning.
- Case data can be hard to gather.

- Predictions are limited to the cases that have been observed.

2.3.8 CBR and Other Reasoning Methods

Other than case-based reasoning, there are many other reasoning methods can be used in solving problems. Rule-based reasoning and model-based reasoning are both the most common reasoning methods to be used in this field. How does case-based reasoning compare with other heuristic method?

The major difference between case-based and rule-based reasoning is the size of the chunks used of reasoning. This leads to a number of other differences, most of which is that rule-based reasoning is a process of composing large number of small chunks to get a solution, while case-based reasoning is a process of adapting small of large chunks. Case-based reasoning been shown to be more efficient in several solutions.

The relationship between case-based and model-based reasoning is more complementary. Both were created to avoid reasoning from scratch, and both are committed to reasoning with large chunks of knowledge. The knowledge they use, however, is quite different, with models representing general knowledge and cases representing specific knowledge. The conclusion is that both are needed for reasoning about compex, real-world situations, especially common sense, in which much is unknown.

2.4 Travel and Tourism

Travel and tourism is the leading application field in the b2c e-commerce, it represents nearly 50% of the total b2c turnover. Already in the past travel, applications were at the forefront of Information Technology, i.e., the airline Computerized Reservation Systems in the early 60s. The industry and its product have rather specific features, which explain this circumstance: the product is a confidence good, consumer decisions are solely based on information beforehand; and the industry is highly networked, based on worldwide cooperation of very different types of stakeholders. Consequently, this industry depends on advanced IT applications. As such, travel and tourism may serve as an example of what happens and will happen in the emerging e-markets, pointing at structural changes as well as challenging application scenarios.

Despite unfulfilled business and stock market expectations, in some sectors such as the travel and tourism industry online transactions are rapidly increasing. This industry is the leading application in the b2c arena. The travel and tourism industry is witnessing an acceptance of e-commerce to the extent that the structure of the industry and the way business is conducted, is changing. The Internet used not only for information gathering; there is an obvious acceptance of ordering services over the Internet. A new type of user is emerging; they become their own travel agents and build their travel packages themselves.

This importance of e-commerce can be explained by the features of the industry, but it highlights also another issue, not less important: e-commerce and especially the Web are not only transaction and business oriented, it is also a medium of curiosity, of creating communities or having just fun, all of which may or may not result into business. Especially the tourism product has to do with emotional experiences; it is not just business [Werthner, 2001].

The travel and tourism industry is a global (and a globalization) industry, with very specific features:

- Travel and tourism represents approx. 11% of the worldwide GDP (following the tourism satellite account method of the World Travel & Tourism Council).
- There will be one Billion international arrivals in the year 2010 (following the World Tourism Organization). And tourism grows faster than the other economic sectors.
- It represents a cross-sectoral industry, including many related economic sectors such as culture, sport or agriculture, where over 30 different industrial components have been identified that serve travelers.
- The product is perishable, complex and emotional: i) a hotel bed not sold for one night represents a lost income. Suppliers are in a risky situation, which can be reduced if access to information is available; ii) the tourism product is a bundle of basic products, aggregated by some intermediary. To support the rather complex bundling products must have well defined interfaces with respect to consumer needs, prices or also distribution channels; iii) vacations are emotional experience

structures that involve cognitive and sensory stimulations as well as affective responses to certain events.

Tourism is an information based business, the product is a "confidence good"; an a priori comprehensive assessment of its qualities is impossible. Tourists have to leave their daily environment for consuming the product. At the moment of decision-making only a model of the product, its description, is available. This characteristic of tourism products entails high information search costs and causes informational market imperfections. Consequently, the industry has comparably long information and value chains.

The Web facilitates new ways to meet changing consumer behavior and to reach new market segments, leading to an "informatization" of the overall tourism value chain. This allows different strategies to generate value [Sweet, 2001]:

- Value extraction: increases efficiency and reduces costs, e.g., automation of processes or outsourcing to clients such as self check-in of hotel guests or airline passengers.
- Value capturing: client and sales data are used to support the marketing, e.g., data mining for forecast or yield management.
- Value adding: a linear combination of products / services to create richer product bundles, e.g., new service quality for consumers such as linking mobile services to existing Web services to advise tourists during their travel.
- Value creation: the focus is on network effects, e.g., tourists within a destination participate in service definition and planning.

Such strategies allow for the design of new products and services, enlarging the range of options to customize and to configure products. IT and, more importantly, improved organizational procedures lower the price of customization, enabling individualized offerings based on mass customization. On the other side, configuration points at the bundling of different product or service components to integrated offerings. Core product can be combined with additional service elements in order to create integrated customer solutions. Given the dynamics of the sector and the emerging already very competitive electronic market, nearly all stakeholders have implemented their Internet strategy. Travel and tourism has also become the playing field for new entrants, either start-ups or companies from the media and IT sector. Since tourism is an information-based business, it fits well with their respective background. The overall trend points towards a further specialization and an ongoing deconstruction of the value chain paralleled by an integration of players and products. Companies will compete and cooperate at the same time, boundaries within the industry are blurring.

The described business scenario is based on flexible network configurations and the further integration of consumers into internal business processes. Adding the tourist life cycle – taking into consideration the mobility aspect of travelers – one can draw the following simplified figure of linking tourists' life cycles with companies' business processes.

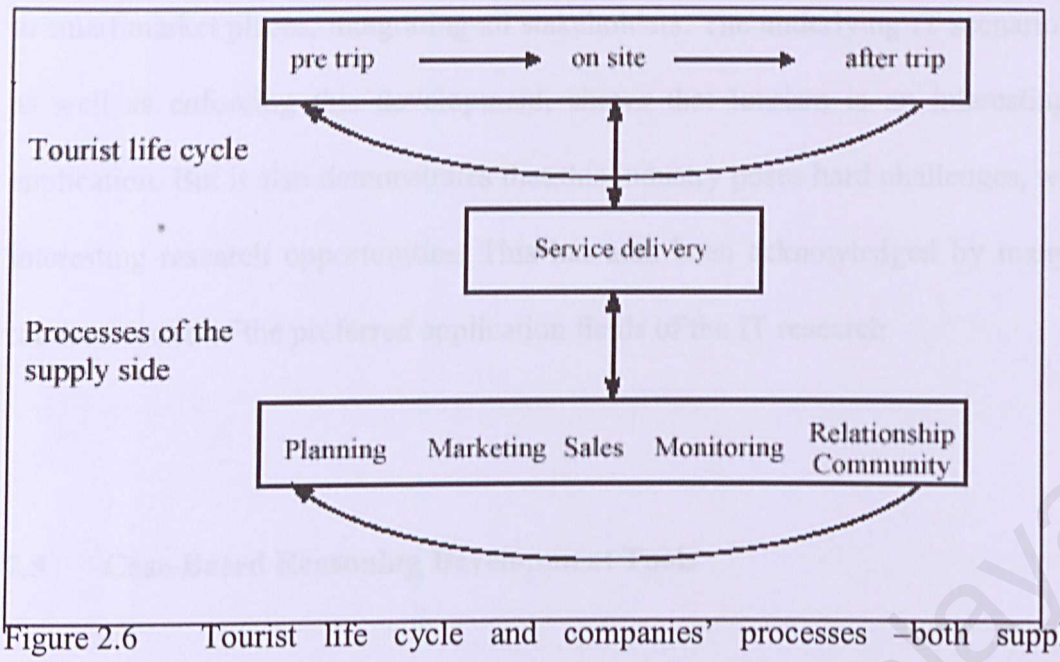


Figure 2.6 Tourist life cycle and companies' processes –both suppliers and intermediaries

Obviously, processes cross company borders, leading to distributed b2b2c applications, requiring cooperation between companies, and supporting also mobile communication with the consumer. Such a future business scenario is based on the assumption that technology – based on a common pervasive infrastructure – will become transparent, invisible for the consumer; information will be available at home, the work place and during travel. Given such a "holistic" approach interleaving business and technology perspectives, what are specific requirements when looking at the user – machine interaction? What is the intelligence needed to support collaboration among companies within a networked industry? These questions lead to interesting considerations regarding system architectures.

The tourism business is changing. More specialized services, flexible network configurations and further consumer integration into internal business processes will lead

to smart market places, integrating all stakeholders. The underlying IT scenario, enabling as well as enforcing this development, shows that tourism is an interesting field of application. But it also demonstrates that this industry poses hard challenges, which offer interesting research opportunities. This has also been acknowledged by many, placing tourism as one of the preferred application fields of the IT research.

2.5 Case-Based Reasoning Development Tools

To build a CBR system, a shell or development tools will be needed. The CBR developer can either build their own or buy the available tools from the market.

A CBR system tool, or Shell, is a software development environment containing the basic components of CBR systems. Associated with a shell is a presented method for building applications by configuring and instantiating these components.

Despite the fact that CBR is a recent innovation, there are at least ten commercially available tools with CBR functionality. This is an impartial review and has no allegiance to any tool vendor.

2.5.1 A Theoretically Ideal Feature Set

A theoretically ideal CBR tool should support each of the main processes of CBR (i.e. retrieval, reuse, revision, and retain). In addition, the tool should support the developers in delivering an efficient solution and the tool must integrate with other systems. The following are the recommended function set:

- ***Representation*** – This must support a full range of date types (e.g. numeric, string, Boolean and symbol) and should be able to structure cases in ways relevant to the application domain. Flat records of value: attribute pairs may be sufficient for some applications, but complex domains may require ordered symbol hierarchies, relationships between features and may benefit from object-oriented inheritance.
- ***Retention*** – The case base should be organized into a manageable structure that supports efficient search and retrieval methods. A balance should be found between storing methods that preserve the semantic richness of cases and their indices and methods that simplify the access and retrieval of relevant cases.
- ***Retrieval*** – Indexing of case will be necessary to make retrieval efficient so indexing must be supported. It may be automatic, but developers should be able to influence the process. If nearest neighbor is used, then case features should be able to weighted and similarity measures customized. If inductive techniques are used the index tree generated should be open to inspection and alteration by developers.
- ***Revision*** – This requires the provision within the tool of a programming language for case adaptation. The language may be procedural or use KBS techniques.

The following are some other issues that are relevance to developers:

- The tool should be able to manage large case-bases with retrieval times increasing linearly (at worst) with the number of cases.
- The tool should support a variety of retrieval mechanisms and allow them to be mixed when necessary.
- The tool should provide the developer with a variety of metrics to both assist the development of an efficient system and the maintenance of the case-base.
- Since organizations may hold case data in existing databases, the tool should be able to import data from the full range of corporate databases.
- A CBR tool should provide a good user interface both for the developers and for operational users.
- Since it may be necessary to embed the developed application the tool should provide a C function library (or similar e.g. a DLL) or support the use of communication protocols such as MS Windows Dynamic Data Exchange (DDE).

2.5.2 CBR Software Tools

Theoreticians might argue that the current surge in interest in CBR is due to the intuitive nature of CBR and because it may closely resemble human reasoning. Software vendors might argue that it is because CBR tools have made the theory practically feasible. There is truth in both views but certainly, the tools have contributed. This section reviews most of the currently available major CBR tools. The tools are dealt with in alphabetical order.

The section concludes with a table that summarizes the functionality of the tools reviewed.

Product	Platforms	Representation	Retrieval	Adaptation	Interface
Art*Enterprise Inference Corporation	A wide variety of PC, workstation, DEC, IBMHP, and mainframe environment	Flat value attribute pairs supporting a full range of variable types	Nearest neighbor but can be using ART's programming environment	Functions, rules and other knowledge-based techniques	Fully featured GUI bulder
CASE-I Astea International	PC Windows	Flat records supporting text and weighted questions	Nearest neighbor and knowledge-guide retrieval	No adaptation features	Interface cannot be customized
Case Power Inductive Solution Inc.	PC Windows, Macs OS/2	MS Excel Spreadsheet ordered symbol hierarchies and nested cases	Nearest neighbor	Via Excel functions and macros	Excel interface
CBR2 Inference Corp.	PC Windows and MVS version	Flat records supporting text and weighted questions	Nearest neighbor and knowledge-guide retrieval	No adaptation features	Tool Book Interface of CBR Express can be customized
Eclipse The Halley Enterprise	Any ANSII C environment	Flat value attribute pairs supporting a full range of variable type	Nearest neighbor	Functions, rules and other knowledge-based techniques	No interface, it is only supplied as C library
ESTEEM Esteem Software Inc.	PC Windows UNIX/X Motif	Cases can be order hierarchically and can be nested	Nearest neighbor and inductive (ID3) retrieval	Functions and rules	GUI builder

KATE *Acknosoft*	PC Windows UNIX	Hierarchical cases	C		Tool Book Interface can be customized
ReCall Soft	PC Windows UNIX	Hierarchical cases with relationships	Nearest neighbor and inductive	Daemons	Graphical development environment
ReMind Cognitive System Inc.		Object-oriented hierarchical cases	Hierarchical cases	Rules	Customizable interface

Table 2.2 Summary of CBR Software Tools

Anyway, of all the mentioned above, I have not chosen any of them. Instead, I have chosen XML as my CBR system development tool.

2.6 Previous Work

The applications of CBR in the tourism field are mostly still in research to be market as commercial product. There are only a few examples of travel recommender systems. The two most successful recommender systems, triplehop.com and vacationcoach.com, can be classified primary as content-based. The user expresses needs, benefits and constraints using the offered language (attributes) and the system matches this in a catalogue of destinations. Neither of these systems can support the user in building a user-defined travel, made of one or more locations to visit, an accommodation and additional attractions (museum, theatre, etc.). Moreover, neither of these exploit the knowledge contained in previous counselling sessions stored as cases.

Chapter 3 Methodology and System Analysis

This chapter will describe the methodology that I had used to complete my application system and the description of system analysis.

3.1 Concept of Methodology

Methodologies provide comprehensive guidelines to follow for completing every activity in the system development life cycle (SDLC), including specific models, tools and techniques. A methodology might be homegrown or purchased from a consulting firm.

3.1.1 Waterfall Model

I have chosen this model because it really provides a high-level view for me to develop this project. This model was chosen because of following reasons:

- **Very structured** – The system is design using a logical flow.
- **Predictable** – It allows estimation of the completion of each stage so that the system can be developed within the time frame given.
- **Involves user participation** – Require information gathering from the user in order to develop a system that meet user needs to a greater extend.
- **Good visibility** – All the requirements can be identified and well defined.

- **More efficiency** – The time and resources can be well determined in order to enable developers to manage the project more efficiently.

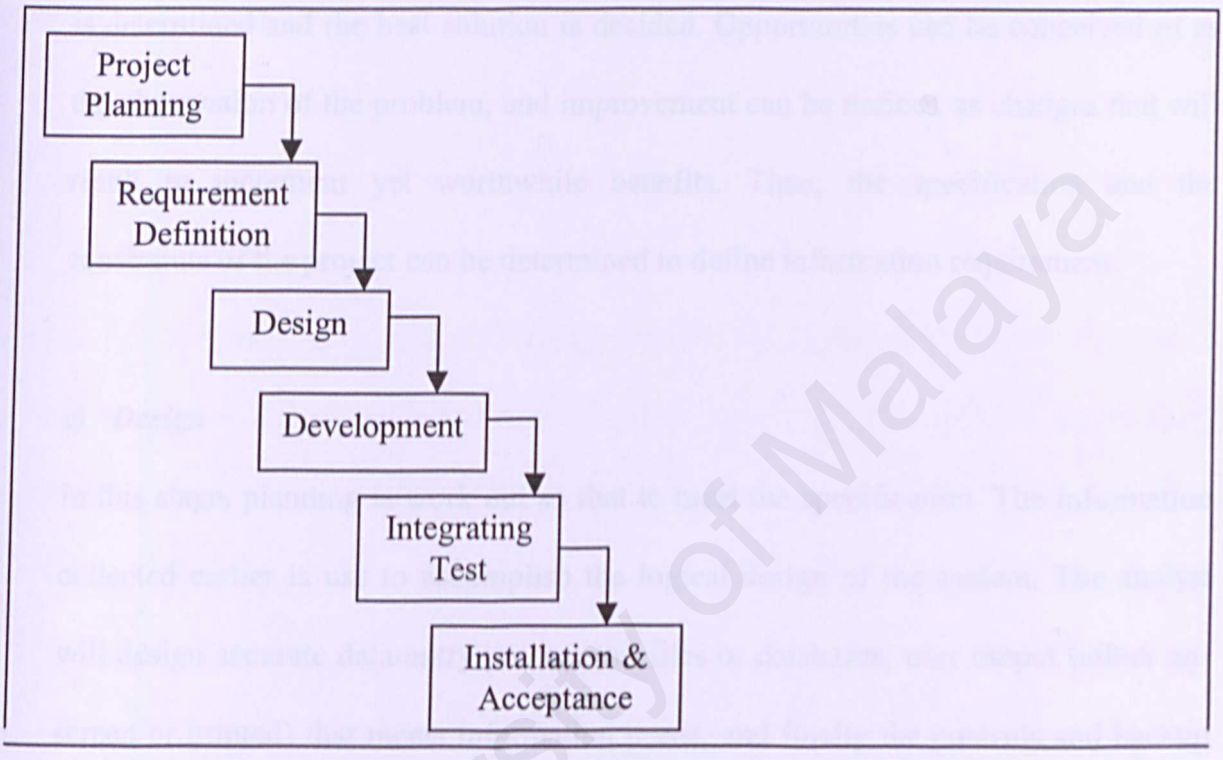


Figure 3.1 Waterfall Model

The following are the descriptions of each process in Waterfall Model that will be involved in this project:

a) Planning

A program management plan is developed that documents the approach to be used and includes the discussion of methods, tools, tasks, resources, project schedule, and users input.

b) Requirement

In the first phase of the system development life cycle, the system analysis is concerned with identifying problems, opportunities and objectives. The real problem is determined and the best solution is decided. Opportunities can be conceived of as the observation of the problem, and improvement can be defined as changes that will result in increment yet worthwhile benefits. Then, the specification and the constraints of the project can be determined to define information requirement.

c) Design

In this stage, planning is work out so that to meet the specification. The information collected earlier is use to accomplish the logical design of the system. The analyst will design accurate data-entry procedures, files or databases, user output (either on-screen or printed) that meets information needs, and finally the controls and backup procedures to protect that system and the data.

d) Coding

During this phase, the analyst works with the programmers to develop any original software that is needed. Programmers have a key role in this phase because they design, code and remove syntactical errors from the program. To ensure quality, a programmer may conduct either a design or a code walk-through to explain complex portions of the program to a team of other programmers.

e) Testing

Before the system can be used, it must be tested. It is much less costly to catch problems before the system is signed over to users. A series of tests to pinpoint problems are run first with sample data and eventually with actual data from the current system.

f) Operation

In this last phase of the system development, the analyst helps implement the system that involves training the users to handle the system. The vendors do some training, but oversight of trainings is the responsibility of the system analyst. Additionally, the analyst needs to plan for a smooth conversion files from old systems to new one. This process includes converting files from old formats to new ones or building a database, installing equipment and bringing the new system into production.

g) Documentation

This integrated process takes place every phase. Activities of each phase are documented in the report form so that to provide a clear view of the progress of each stage.

3.2 Case-Based Reasoning Overview

Case-based reasoning (CBR) is modeling human in solving problem by relying heavily on memory of experiences, which is called cases in CBR. Consider a simple everyday problem such as using an unfamiliar machine. Rather than starting from scratch when learning how to use a new machine, people rely on memory of experiences with other machines. These memories allow them to derive shortcuts and anticipate problems that might arise by having previously observed and solved similar problems.

Although previous cases provide important information for dealing with the current situation, they are not exactly the same. It would be necessary for the person to have to provide a perfect description of the previous case in order to retrieve it from memory. People are very good at recognizing similarities in situations while traditional programming techniques require exact matching.

CBR contains the following features:

- Description of the features of and solution to previous problems to assist in solving a current problem. These descriptions are called cases.
- A description of the features of the current problem. These description is called the presented case.
- A matching operation which provides partial rather than complete matching.

The matching operation matches a presented case with previous cases by accumulating pieces of evidence that the current case is similar to a stored case. A scoring formula converts the accumulated pieces of evidence into a single score for the match between the presented case and any previous case. The stored case with the highest scores has the best match to the presented case. Generally, the cases with the highest scores are returned by the case-based reasoning mechanism.

In XML, cases are represented as objects. The features used by the matching mechanism are attributes of the case objects. These cases are referenced by a case-base which is also an object.

A case-based application has the following components:

- Cases that represent problems and their solutions.
- A case base where the cases are stored.
- A matching operation for matching a new presented case against the cases stored in the case-base.

A typical usage of case-based reasoning follows the sequence of events shown below:

Build a case base

- Create a case base that will hold the pertinent case information.
- Define the significant features that describe the cases.

- Submit cases to the case base as objects, whose attributes represent the case features.
- Save the case-base index to a file.

Query a case base

- Load the case-base index file into XML.
- Specify the match parameters, such as how many cases should be retrieved and how precisely they should match.
- Build a presented-case object that describes a situation you would like to compare for matching against the stored cases of the case base.
- Pass this object to XML's match functions, receiving back a list of the best-matching cases in the case base.
- Refine the presented-case object if necessary, repeating the process from step three until sufficient information has been retrieved to solve the problem.

3.3 CBR Vs Other Reasoning Technique

This section discusses the problems associated with developing model-based reasoning (MBR) and rule-based reasoning (RBR) knowledge based systems. It posts that CBR appears to offer solutions to many of these problems, and presents evidence from the literature to support these claims.

Case-based reasoning (CBR) combines a cognitive model describing how people use and reason from past experience with a technology for finding and presenting such experience. CBR provides a conceptual framework in which to store operator experience and to later provide that experience to other operators to facilitate the situation assessment and solution formulation processes of RPD. This is accomplished by providing a context in which the human operator can view the current state and recent activities of the system and easy access to previous experience.

3.3.1 CBR Vs Rule-Based Reasoning

RBR knowledge-based system typically provides only rule trace backs as an explanation of their activities, which ever an experienced system operator, has difficulty interpreting. In contrast, a case-based system is capable of explaining its activities in the context of the case form, which it was reasoning, thereby, giving the operator much more useful information to guide situation assessment.

Many complex systems are managed in control periods which decompose long durations of system management into control scenarios, or sequences of activities (e.g. aircraft flight plan or a shift plan in a progress control plant). Such control scenarios can be viewed individually as experiences, or cases. Together, they form a case base which is the basis of an automated knowledge-based system.

3.3.2 CBR Vs Model-Based Reasoning

As in MBR knowledge-based system, the last thirty years many knowledge-based systems have been developed that have an explicit model of the problem domain in which they operate. In many such systems the model is implemented by rules, and perhaps more recently by objects. In second-generation systems a deep underlying causal model exists that enables the system to reason from first principles in its application domain. There is little doubt that much MBR systems (whether they are deep or shallow) can be very successful. However, there are five major problems with this approach:

1) Knowledge elicitation is difficult.
2) KBS can be very complex and can take many, many years to develop.
3) KBS are frequently slow.
4) KBS are often poor at managing large volumes of information.
5) Once developed, they are difficult to maintain.

3.3.3 CBR Advantages over Other Reasoning Techniques

People solve problems by using their experience. It is no surprise that expert and experience derive from the same root. We posit that the KBS community was seduced by rules and neglected the truism that experts solve problems by applying their experience, whilst only novices attempt to solve problems by applying rules thay have recently acquired. The application of experience to problem solving is the hallmark of CBR. Thus,

CBR is proposed by some as a psychological theory of human cognition and one that provides a cognitive model of how people solve problems. It offers a paradigm that is claimed to be close to the way people solve problems and one that overcomes the brittleness of MBR and RBR systems.

Hence, there is a strong case for CBR since it has several potential advantages over MBr and RBR:

- CBR systems can be built without passing through the knowledge elicitation bottleneck since elicitation becomes a simpler task out acquiring past cases.
- CBR systems can be built where a model does not exist.
- Implementation becomes a simpler task of identifying relevant case features, and a system can be rolled out with only a partial case-base. This removes one of the bug-bears of KBS – how to tell when a knowledge-base is complete.
- CBR systems can propose a solution quickly by avoiding the need to infer an answer from first principles each time.
- Individual or generalized cases can be used to provide explanation that are perhaps more satisfactory than explanations generated by chain of rules.
- CBR systems can learn by acquiring new cases making maintenance.
- Finally, by acquiring new episodic cases CBR systems can grow to reflect their organization's experience. If a rule-based KBS were delivered to six companies and used for six months, after that time each system would be identical, assuming no maintenance had taken place. If six identical CBR systems are used in a

similar way after six months there would be six different systems as each could have acquired different episodic cases

3.4 System Requirement Analysis

System analysis involves identifying system requirements, which includes functional and non-functional requirements.

In developing a system, it is essential for the developer to identify the system's requirements. A lot of information is needed for this purpose and through series of research and reviews, the developer will finally understand what the system is functional and non-functional requirements are.

3.4.1 Functional Requirements

Functional requirement describes an interaction between the system and its environment. For example, to determine the functional requirements, we decide which states are acceptable ones for the system to be in. Further, functional requirements discuss how the system should behave given certain stimuli [Pfleeger, 2001]

The functional requirements of this project area as follows:

- **Advisor/Recommendation Module**

Be able to recommend and advice appropriate tourism destination for given travel preferences.

- Be able to provide adequate explanation as to how it arrived at a particular conclusion.

3.4.2 Non-Functional Requirement

Non-functional requirement or constraints describes a restriction on the system that limits our choices for constructing a solution to the problem [Pfleeger, 2001]. Non-functional requirements for the system are as follows:

- **Reliability**

The application system, software and hardware shall be reliable and shall not cause unnecessary and unplanned down time of the overall environment.

- **Effectiveness**

The system's ability to meet its objectives and every function has a specific purpose in the system

- **Maintainability**

It is the degree that the system can be cost affectively made to perform its function in a possibly changing operating environment. The system is easy to modify and test in

updating process to meet the new request, correcting errors, or more to a different computer system.

- **Interesting interface**

Interesting interface is a vital aspect needed to encourage users to use the system.

- **Accurate and robust**

3.5 Development Tools

Decision on using the appropriate software for developing the system is important for system implementation. This is to ensure that the software that is going to be used to develop this project is suitable with the concept of the project. This includes the main software needed to develop the system and the software is required to support the system and extend the system's capability.

3.5.1 Software Requirement

3.5.1.1 Server Software Requirements

The software that will be utilized in developing this system are:

- Windows 2000
- Internet Information Server (IIS) 4.0
- Microsoft Access 2000
- Active Server Pages (ASP)
- Adobe Photoshop 6.0
- Internet Explorer 4.0
- XML

3.5.1.2 Clients Software Requirements

Software requirements for client are quite minimal as compared to server. Software requirements are as follows:

- Windows 95/98
- Internet Explorer 4.0

3.5.2 Hardware Requirement

3.5.2.1 Server Hardware Requirements

Server hardware requirements are as follows:

- A server with at least Pentium III 133 MHz processor

- 2GB of Hard Disk space
- 32 Mb of RAM memories
- Network connections
- Other standard computer peripherals

3.5.2.2 Client Hardware Requirements

Client hardware requirements are as follows:

- 32 Mb of RAM memories
- Connected to the network
- Other standard computer peripherals

3.5.3 Development Environment and Tools

3.5.3.1 Operating System Platform – Microsoft Windows 2000 Pro

The reason why Microsoft Windows 2000 Pro is chosen for this project's operating system platform is because Windows 2000 Professional is the preferred 32-bit desktop environment, providing a combination of Windows 98 usability and Windows NT 4 reliability. New features of Windows 2000 include aupport for power management and

plug and play (a great bonus for portable users) and support for new file system features, including EFS.

Windows 2000 server has been created in three different versions, each with features appropriate target audience. All share the same core features as Windows 2000 Pro, but then add additional features. Its key features include:

- Synchronization Manager
- Internet Printing Protocol
- Plug and Play
- EFS
- Enhanced Security Feature
- Multiple Monitor Support
- Support for USB and power management
- Two way SMB and support for 4 GB RAM

3.5.3.2 Web Database – Microsoft Access 2000

As for Web Database, Microsoft Access 2000 is chosen for this project. Microsoft Access 2000 is a good tool to use for storing and managing large amounts or complex sets of data. Access offers many sophisticated and easy-to-use methods for entering and editing items in a database, tracking, sorting, arranging, and viewing the information on screen or summarizing it in printed reports.

3.5.3.3 Web Server – Microsoft Internet Information Servers (IIS) 4.0

Developed by Microsoft, IIS 4.0 is chosen for this project, IIS 4.0 provides World Wide Web server, File Transfer Protocol server, Network News Transfer Protocol server, and Simple Mail Transfer Protocol server operations integrated with the Windows NT server 4.0-network operating system.

Internet Information Servers 4.0 has a completely different interface from prior versions of IIS, as well as substantially enhanced capabilities. The central IIS 4.0 administration tool, the Internet Service Manager (ISM), consists of the new MMC-based console as well as the Web-based Internet Service Manager (HTML).

The new Windows ISM interface uses the Microsoft Management Console (MMC). All IIS services are implemented as MMC-based snap-ins. The ISM centralizes all the configuration options into a hierarchical tree of nodes (containing services, sites, and virtual directories). The Web-based ISM uses a client browser to provide similar administrative access via Active Server Pages.

IIS 4.0 supplies a range of new features to enhance Web site administrator and Web application development and deployment.

Features of IIS 4.0 include:

- Inheritable master properties settings that apply to all Web or all FTP sites.
- Per-site configuration for all operational and security settings for each Web site and FTP site.
- Per-virtual directing configuration for operational and security settings
- Per-directory configuration of directory security and content location, HTTP headers, and error messages for all Web site directories.
- Per-file configuration of file security and locations, HTTP headers, and error messages for all Web site files.
- IIS 4.0 supports HTTP 1.1 capabilities, including persistent connections, pipelining, caching directives, host headers, and HTTP redirects.
- Enhanced security with certificate management and certificate services.

3.5.3.4 Web Application Technology – Active Server Pages

ASP stands for Active Server Pages and it is a program that runs inside Internet Information Server (IIS). Developed by Microsoft, Active Server Pages is a programming environment that gives the ability to generate dynamic html pages with the help of server side scripting.

VBScript is the default scripting for ASP, but if we prefer to use Jscript, Perl or any other scripting language for server side scripting in an ASP page, it is possible. An ASP page is

almost the same as a HTM or HTML page, the only difference is that an ASP page has the '.asp' extension. Active Server Page can include both client and server side scripts.

Features of ASP:

- Dynamically edit, change or add any content of a Web page.
- Respond to user queries or data submitted from HTML forms.
- Access any data or databases and return the results to a browser.
- Customize a Web page to make it more useful for individual users.
- The advantages of using ASP instead of CGI and Perl, are those of simplicity and speed.
- Provides security since your ASP code cannot be viewed from the browser.
- Since ASP files are returned as plain HTML, they can be viewed in any browser.
- Clever ASP programming can minimize the network traffic.

3.5.3.5 XML

XML is simply a means for representing and describing data. The language allows you to define markup tags that can be wrapped around data in order to describe it at a finer granularity that current systems typically do.

By far, the most commonly cited use of XML is the sharing of data. A schema in XML is a conceptual framework that describes the underlying structure of your collection of elements. Schemas serve several important purposes. First, they define the "vocabularies"

of element types and attributes for a given class of documents and allow you to share those documents with other applications. A document may not necessarily be the traditional document you normally think of. It may be a database document, a middleware message, or a Web page. These schemas can be formally defined as document type definitions (DTDs). Once you have these formal rules, applications can use them to validate structures and objects to ensure they are receiving the correct information appropriately formatted. Schemas also play an important role in information retrieval. For instance, metadata defined by an XML schema can provide information to software agents, or may be used as part of the discovery process. XML allows us to describe that data in ways that give meaning, understanding, even intelligence to it. Wherever there is data, XML can be applied. The reason XML is becoming such a vital skill in the Web developer's arsenal is because XML is being sewn into the very fabric of the Web infrastructure. That infrastructure includes all three tiers: Web servers, Web browsers, and back-end databases.

Chapter 4 System Design

This chapter will discuss about the design of the system. Each stage in the process has been disrupted into more details. Data Flow Diagram (DFD) will be used to describe the facets in the proposed system where the module in the system has been drawn out. The details represented in this chapter will serve as a reference and important guidance for the system development phase as well as the system implementation and maintenance phase.

4.1 Introduction

Design is the creative process of transforming the problem into a solution. The description of a solution is also called design. It is viewed as a process that represents data structure, program structure, interface properties and procedural information. Design is a method that translates user requirements into a product or finished system. All of the information gathered during the system analysis phase will be oriented into smaller modules until a system is being developed.

4.2 Structure Design

Structure chart will be used to show the workflow of the system. This workflow of the system is based on the CBR cycle as well as it is mainly a CBR system.

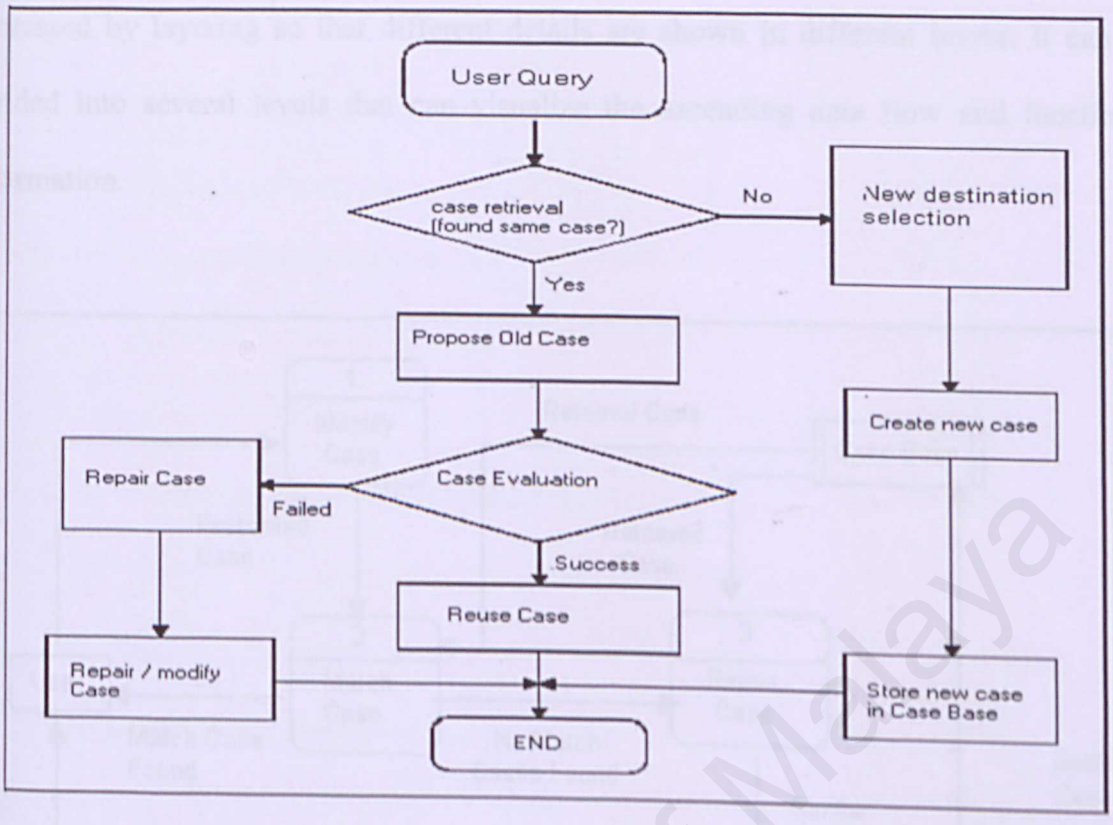


Figure 4.1 Workflow of the system

4.3 Process Design

Process design will be visualize using Data Flow Diagram. It is a graphical technique that will display the data flow in the system. As a transformer of data, the diagram shows the data flow into the system, how they are transformed and how they leave the system. The emphasis in on the flow of the data, not on the flow of control. It will also be able to view the changing process or the converting process that is being implemented to the data once the data goes into the systems, through the system and out of the system. The hierarchy is

expressed by layering so that different details are shown in different layers. It can be divided into several levels that can visualize the ascending data flow and functional information.

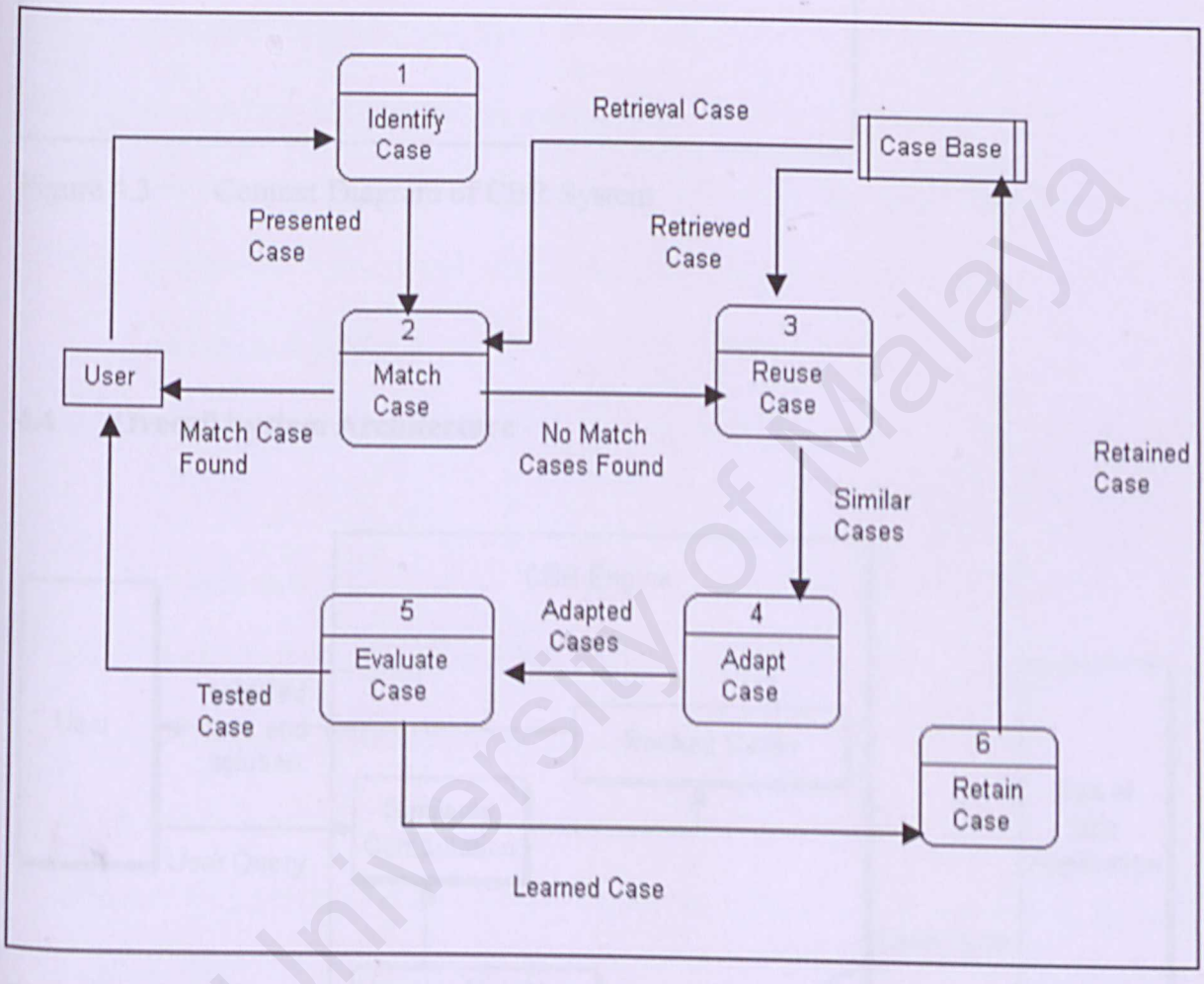


Figure 4.2 DFD for CBR System

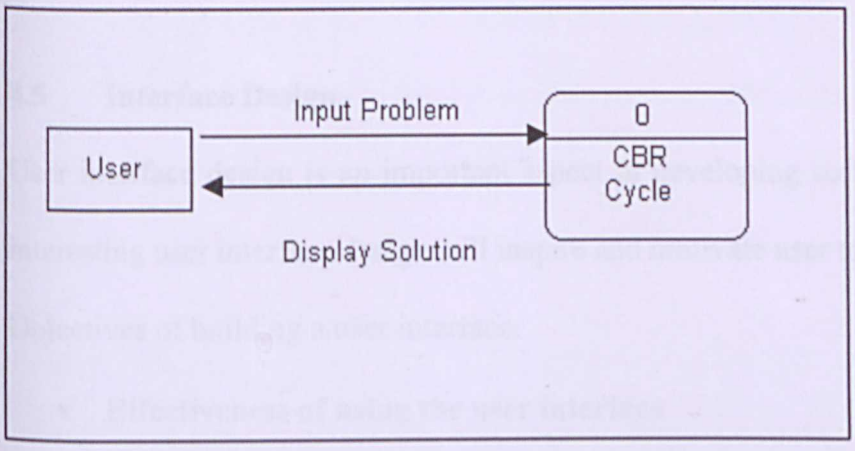


Figure 4.3 Context Diagram of CBR System

4.4 Overall System Architecture

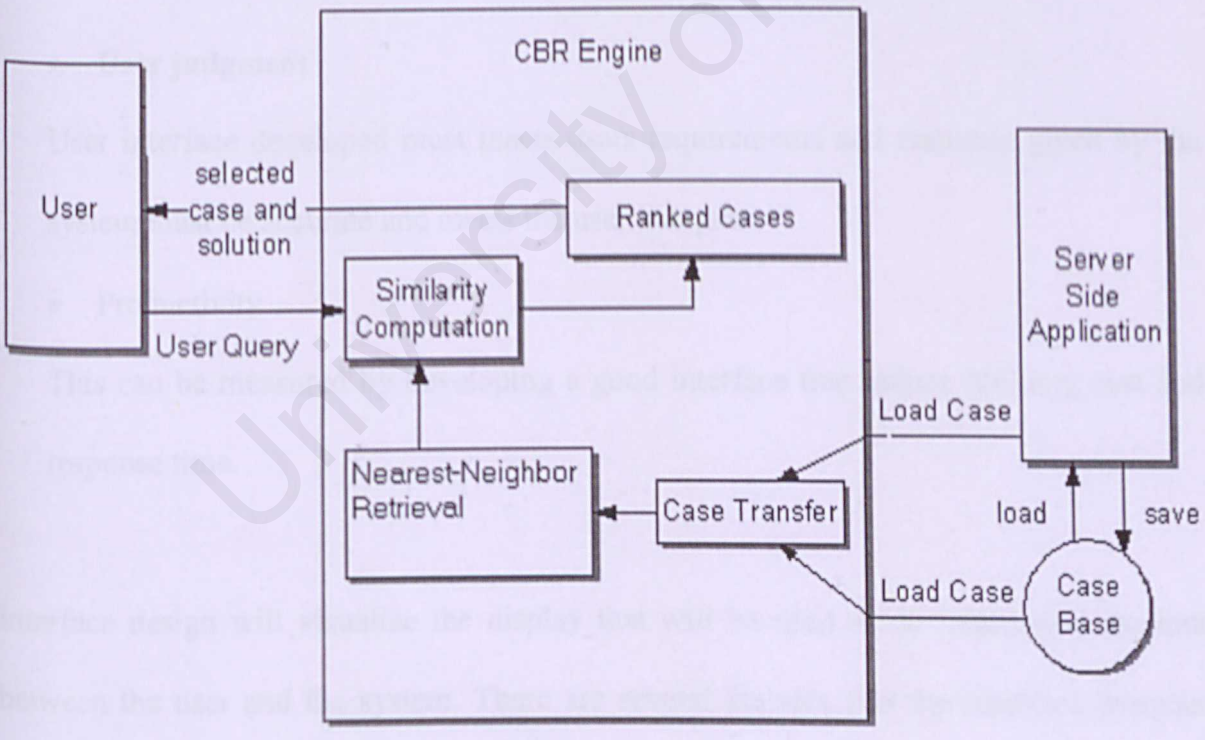


Figure 4.4 Architecture of the System

4.5 Interface Design

User interface design is an important aspect in developing software. This is because an interesting user interface design will inspire and motivate user to test and use the system.

Objectives of building a user interface:

- **Effectiveness of using the user interface**

  This can be accomplish by designing an interface that meets the user requirements and the simplicity of the interface to avoid complication and confusion.

- **Interface reliability**

  Interface accuracy in performing data capturing without errors

  .

- **User judgment**

  User interface developed must meets users requirements and response given by the system must be accurate and meets the user's request.

- Productivity

  This can be measured by developing a good interface that reduce building cost and response time.

Interface design will visualize the display that will be used as an interaction medium between the user and the system. There are several features that the interface designer needs to take into considerations during the interface design. They are:

i. Soft background color, icons, logos, pictures and appropriate fonts.

ii. Consistency between pages to avoid confusion for users.

iii. Suitable type of fonts are used so that the interface would not look too complex and it would be more organized and easier to understand.

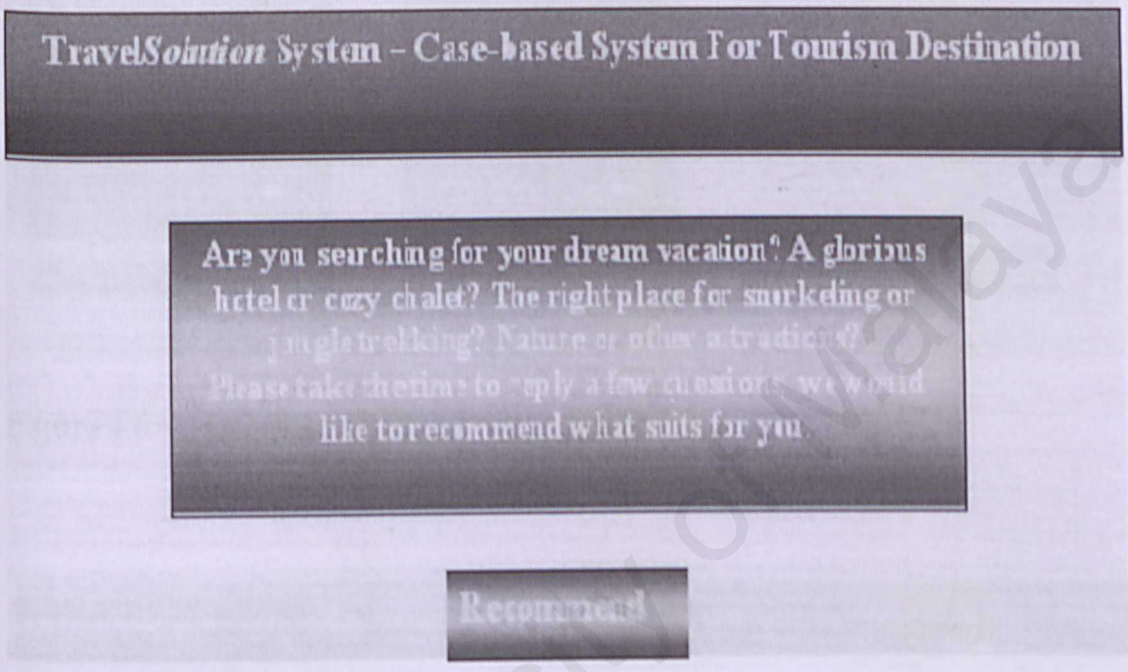


Figure 4.5 Interface for Main Menu

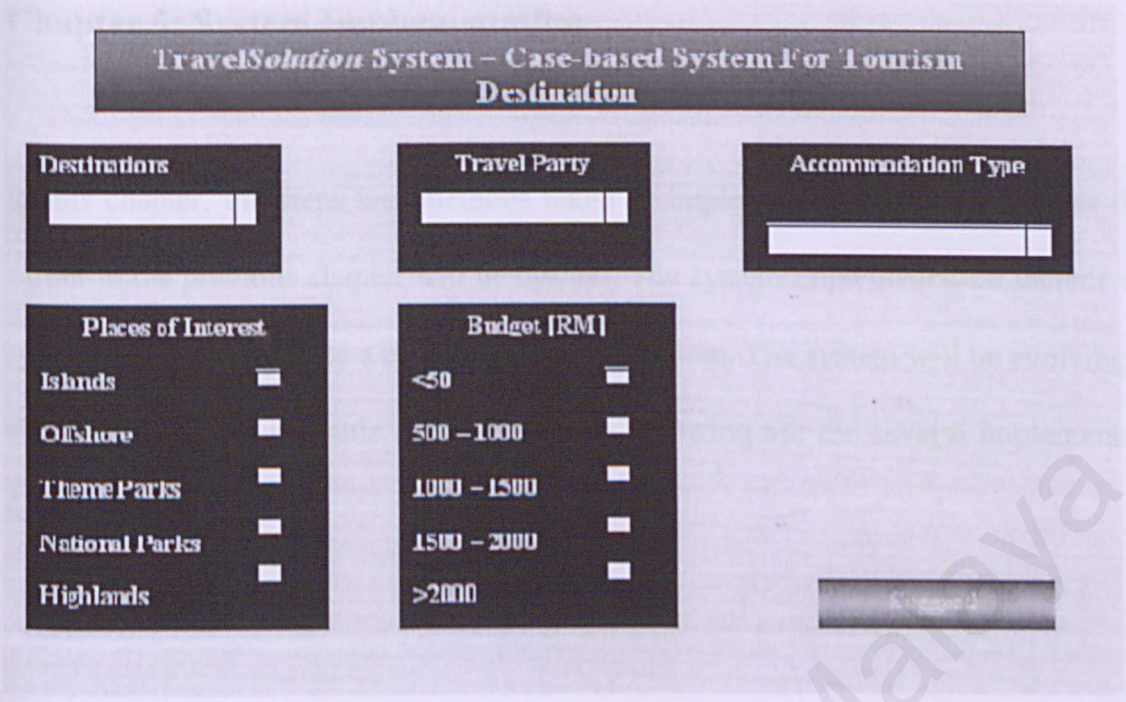


Figure 4.6 Interface for Recommendation Module

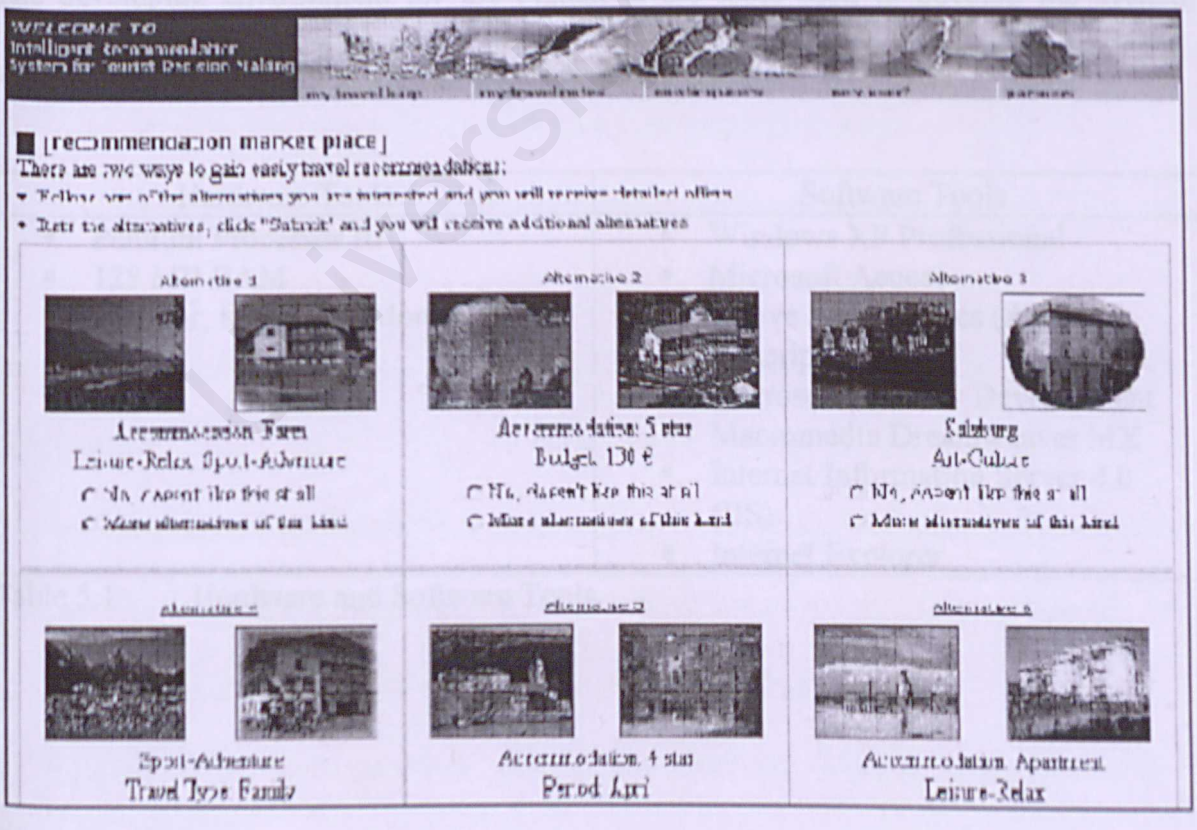


Figure 4.7 Example of Output

Chapter 5: System Implementation

In this chapter, the steps and methods taken to implement the system that was design earlier in the previous chapter will be discuss. The system implementation include of the system design structure to a computer readable system. The system will be evolving from scratch design to a run able application. The following are the several implementations for this system.

5.1 Development Environment

The developing environment for the system is the tools used to develop the system, which includes the hardware tools and the software tools.

Hardware Tools	Software Tools
▪ Pentium Processor III ▪ 128 MB RAM ▪ Monitor, Keyboard, Mouse	▪ Windows XP Professional ▪ Microsoft Access ▪ Active Server Pages (ASP) ▪ Vbscript ▪ Microsoft Interdev Development ▪ Macromedia Dreamweaver MX ▪ Internet Information Server 4.0 (IIS) ▪ Internet Explorer

Table 5.1 Hardware and Software Tools

5.2 Functions Implementation

In this project, graphic user interface has been implemented, as the system will not only be focusing on the application. For this, I used Macromedia Dreamweaver MX to design the user's interface. The functions are created in order to allow users to select their options list and to key in input for the CBR system. The functions will then generate outputs to be displayed by the users.

During the case matching process for this CBR system, users will just need to select and key in their options. Functions made the system easier to use and understand. Below are the functions that have been created for the system:

5.3 Active Server Pages (ASP) and VBscript Implementation

5.3.1 Case-Based Reasoning (CBR) Engine Implementation

As discussed in the previous chapters before, there are four main phase in the CBR cycle, which are:

1. ***Retrieve***. Given inputs from user, retrieve a set of cases stored in the case base, whose problems are evaluated as similar. Indexes, case base partitions, case clusters or other similar tools can be used to speed-up this stage.
2. ***Reuse***. The retrieved cases are reused to build the solution. This stage could be very simple, e.g. only extract the "solution" component from one retrieved case or

much more complex, e.g. to integrate all the solutions extracted from the retrieved cases to build a new candidate solution.

3. ***Revise***. The solution(s) are then adapted to fit the specific constraint of the current situation. For instance a reused therapy for a patient suffering for similar disease must be adapted to the current patient (e.g. considering differences in the weights or ages of the two patients).
4. ***Retain***. The new case is possibly added to the case base. Not all the cases built following this process must be added to the case base. There could be poorly evaluated cases or cases too similar to previous situation and therefore not bringing new knowledge.

Nevertheless, in this system, only three phases from the CBR problem solving process cycle are implemented, which are *retrieve, reuse* and *retain*. The third phase, revise is not implemented due to programming skill problem and time constraint consideration. Later, for future enhancement this function will be added.

A similarity metric is used to compare the problem component of the new case being built with the previous matched cases in the base. For this, I use strings compare function as we deal with many texts. Weighted value is set to one (1) and zero (0). For each attribute in the case base, if user has input is similar with the data contain in the case base, score one (1) will be given, and score zero (0) if otherwise. Below is a part of coding I used to implement this function:

```
If ((StrComp(objRT("Package_Type"), strpackagetype, vbTextCompare) = 0) AND _
    (StrComp(objRT("Travel_Destination"), strtraveldestination, vbTextCompare) = 0)
    AND _
    (StrComp(objRT("Place_Name"),strplacename, vbTextCompare) = 0) AND _
    (intBudget = objRT("Budget")) AND _
    (intstayduration = objRT("Stay_Duration")) AND _
    (StrComp(objRT("Category"), strCategory, vbTextCompare) = 0) AND _
    (StrComp(objRT("Room_Type"), strroomtype, vbTextCompare) = 0) AND _
    (StrComp(objRT("Activities"), strActivity, vbTextCompare) = 0)) Then

    Response.Redirect "temp.asp"
```

As for the attribute 'budget', the scoring function of one (1) and zero (0) will not be implemented because this requires user to input their budget limit themselves without having certain options. Instead, weighted value of 0.8 or 0.5 is set, whichever is the nearest to the budget value inputted by user. Below is part of the coding I used to implement this function:

```
If(intBudget = objRS("budget")) Then
        intberat3 = 1
        Else If((intBudget < objRS("budget") - 20) OR (intBudget > objRS("budget") -
        20)) Then
                intberat3 = 0.8
                Else If((intBudget < objRS("budget") - 50) OR (intBudget >
                objRS("budget") - 50)) Then
                        intberat3 = 0.5
                        Else
                        intberat3 = 0
                        End If
                End If
        End If
```

After all the scores have been computed the system will rank the results in percentage. The process of similarity calculation:

- For each attribute in the presented case, an attribute score is computed for each stored case in the case base.
- For each stored case, the attribute score are combined to produce a case score.

- The match process returns the highest scores.

The attribute score for a stored case is computed from a match contribution as follows:

- If the attribute values in the stored and presented cases match, the match contribution is added to the score.
- For the attribute budget, weighted value is assigned so that even when the user's input do not match, it will not contribute zero score to the similarity calculation but it will always has a value that reflects the most similar with presented value.

The similarity of the case matching is calculated using the formula below:

Total1 = Total of all scored attribute

Total2 = Total1/Number of Attributes

Total3 = Total2 * 100 (Ranking percentage)

Scoring with the above formula will find the most complete matches for the presented case. Ten most similar cases have been set for the system maximum match.

5.3.2 Database

For constructing the database for this system, I used Microsoft Access as it is not a big system which needs large database. It has three main tables. There are:

- **'tourism' table** – Contains data of type of package, name of the islands, name of the destination, budget, duration of staying, category of accommodation, type of room, and main activities that can be done in the particular place.
- **'case' table** – Contains previous matched cases which has the same data as in the 'tourism' table but this case base has extra fields and with the solution for each similarity matching of each attribute.
- **'temp'** – Hold current user's recommended solution. Each time the user complete using the system, the temporary table will be cleared up, ready for other user's recommended solution.

There are seven attributes which are *'package_type'*, *'travel_destination'*, *'place_name'*, *'budget'*, *'stay_duration'*, *'category'*, *'room_type'*, and *'activities'*. These attributes are the main case representation for the system. I had chose these seven attributes that I find will be what user may include in their options to consider when they plan their traveling or holiday.

There is ability or a specific function of the system to add the new matched cases to the case base. Each time when the user used the system, the system will retrieve ten most similar cases as the solutions and this will automatically added to the case base. Hence, there is no need for the user to click the save button.

Below are coding sample I used for database connection:

```
<%
        Dim objConn
        Set objConn = Server.CreateObject("ADODB.Connection")
        objConn.ConnectionString = "FILEDSN=C:\Program Files\Common
        Files\ODBC\Data Sources\travelsolution.dsn"
        objConn.Open
%>
```

Other functions in this system can be seen in the Appendix.

Chapter 6: System Testing

6.1 Introduction

Testing is important in developing a system. The process of testing and debugging are for detecting defects and bugs of a system. These processes are usually done incrementally with system development.

This phase of system testing is sometimes referred as verification and validation. Verification means a set of activities that ensure the system correctly implements a specific function while validation means a different set of activities that ensure the system has been built is traceable to user's requirements. A successful test is one in which no errors are found.

The following are the objectives for system testing:

- To reveal inference error by the case base matching and case retrieval
- To compare the expected outcome with the actual outcome. Eventually, debug it to enhance its functionality and capability
- To ensure the accuracy of the case matching and make sure every case from the case can be retrieved for the matching

6.2 Module Testing

Module testing focuses on verification effort on the smallest unit of system design that is the system component or module.

Each module is tested independently to ensure its workability and error free. The interface module is tested for its ease of use, simplicity and ambiguity, and the system is tested on its knowledge acquisition capabilities efficiency and usability.

6.2.1 Case Matching Accuracy Testing

The accuracy of the case matching is tested by presenting a few known cases for matching. If the score for the matched case is exactly what has been expected, then there should be no problem with the accuracy. If not, then the problem would have either occurred from similarity matching (strings compare), weighting value or retrieval algorithm. If the problems occurred from weight value, then the weight value will be reassigned and weight testing is necessary.

In this project, the system is based on CBR engine algorithm. By using a specific calculation function, the sum value of match contribution will be achieved. A number of known cases are presented to the system with different value for every attribute to define the accuracy.

6.2.2 Case Base Module Testing

This is referring to the case base testing to ensure that every data which user had input is stored accurately and correctly, to the correspondence case base which is in the Microsoft Access database format. A set of sample raw data is created for the testing purpose in this module. The process includes iterate the checking on the duplicated data in the database to ensure that every entered data is valid and ease the redundancy and duplication problems.

6.2.3 Coding Module Testing

Times of testing have been done on the CBR engine algorithm to ensure case retrieval and case matching with highest accuracy and response time.

In this project, the results of CBR engine model are showed to reveal an important improvement in case matching accuracy and faster response time for a large case base by this implementation.

To test whether the case retrieval process is working or not, I must make sure that each of the attribute of cases have been assign a specific data with different priorities. For example, only the budget attribute is assigned with weighted value to ensure more of accuracies during case similarity matching. The CBR engine model has been

implemented to create a case-based retrieval algorithm. This methodology is used to assign relative importance of feature and fast response time for case retrieval.

6.3 Integration Testing

In this project, a bottom-up approach has been used for system testing. It begins with construction and testing with modules at the lowest levels of the system and then moving upward to the modules at the higher levels of the system. Since all the modules have been tested and have been declared bug free, the modules will be combined. The integration is checked again to ensure that there is no error. The matching and retrieving process are tested thoroughly to demonstrate the essence of case-based methodology.

6.4 System Testing

System testing is a series of different tests designed to fully exercise the system to uncover its limitations and measure its capabilities. All modules are combined to complete the system and it is tested as a whole to ensure workability and error free.
The objective is to test the whole system and verify that it meets specified requirements. Testing is concentrated on the whole reasoning process from past case acquisition until evolution in system and retainment of solved case. Several types of system testing are worthwhile for a system.

6.4.1 Function Testing

It is a test during the run time environment where the complete sets of functions are loaded. If ASP can run the rules and functions smoothly without error, then it is considered success.

6.4.2 Performance Testing

The purpose of this testing is to test the run-time performance of the system in case retrieving for matching.

6.4.3 Testing Analysis

The system testing shows that the system able to display the expected match cases for both complete or uncompleted presented case and the rank percentage is being displayed for user references. Besides, the system is easy to use and understand. As a conclusion, all the objectives have been achieved.

Chapter 7: System Evaluation and Conclusion

Various problems were encountered when developing this system. These problems were solved through research and studies in the tourism field such as the Internet, journals and reference books and not forgetting continuous consultation and help from my own supervisor, Encik Mohd Nor Ridzuan Bin Daud and also from friends. The system's strength, limitations and future enhancement were identified.

7.1 Problems Encountered and Solutions

Problems are everywhere and so does a developing system. Several problems had encountered throughout the development of this system. Below are some of the problems along with actions taken in order to overcome these problems.

7.1.1 Difficulty in Choosing Development Tools and Technology

There are many software tools available to develop a case-based reasoning system. Choosing a suitable and most convenient tools and technology was a critical process as all tools possesses their own strength and weaknesses. In addition, the availability of the required tools for development was also a major consideration. Eventually, VBscript and

ASP have been chosen of its strength in programming methods and recommendation by my supervisor and some of my friends.

In first phase of this project, the proposed programming language was Extensible Markup Language (XML). Nevertheless, because of I do not have basic knowledge about XML, after one month studying it; I found it impossible for me to use XML for the project. XML needs a lot of time to master it although it is very powerful for CBR implementation. In addition, considering the system requirement and time constraint, ASP and VBscript are more suitable and easier to learn. Moreover, should there be any problems encountered, help and guidance from who is good at these are easier to get.

7.1.2 Lack of Knowledge in CBR and Implementation of CBR Technique in Tourism

Nothing is too difficult in terms of theoretical. All the problems and doubts will emerge once the development and design started. It is hard to create a good case representation as well as to choose an appropriate algorithm for case retrieval and the type of similarity should be used for case matching.

Tourism is very subjective which means that it does not have certain and concrete solution for traveling problems. Hence, it is very difficult to define the scope of domain for the system and to incorporate CBR technique in tourism destination. It is quite a tedious task to find the best attributes or features that user may have choose in this

domain, tourism and holiday planning. A lot of time has been consumed in doing this task.

I have managed to read many articles and journals for references and many resources have been found through the Internet. Then again, consultation and guidance from my supervisor and friends are the main inspiration to solve the problem.

7.1.3 Lack of Knowledge and Skill in ASP and VBscript

Although ASP is much easier to learn compares to XML, there were still a lot of problems encountered during the coding. Since there was no prior knowledge of programming in ASP, there was an uncertainty on how to write and organize codes. These new programming languages and concepts were never taught before and it takes time to understand the programming of ASP as well in order to use it for writing VBscript and compilation, especially in implementing the nearest neighbor algorithm. Fortunately, previous knowledge in other programming language like C++ did help me in understanding the concept.

7.1.4 Lack of Knowledge in Tourism Field

It has been a very new subject for me to understand in order to create a good case representation for this system. As noted, a successful performance of retrieval mechanism

depends very much on good representation too beside similarity metric. Articles, journals, proceeding papers from the Internet have been my resource in understanding the concept of tourism destination. Besides, some useful information from family and friends has been gathered in addition to help me overcome this problem.

7.2 System Strengths

During the development of this project, several of the system's strengths were identified and described as follows.

7.2.1 Easy To Use Interface

Although I would not consider it has user-friendliness, I claim this system is easily use by the users and not complex to understand to any level of age. It has only one main interface, which from there, user can straight away can use the system to find the most suitable tourism destination for them according to their preferences. User need not to log in and have certain password to memorize in order to use the system. The questions and answers are in the form of drop down list which user just need to click to his/her options.

7.2.2 Implementing CBR Technique To Generate Solutions

It is not an easy task to generate a solution for traveling and holiday planning problems according to user's preferences. However, by implementing CBR technique, no matter how the situation of the user, the system is able to provide the most similar cases as the solutions to the presented case.

7.2.3 Expandable Case Base

There is ability or a specific function of the system to add the new matched cases to the case base. Each time when the user used the system, the system will retrieve ten most similar cases as the solutions and this will automatically added to the case base. Therefore, with the increasing number of cases in the case base, the variety of solutions is increase and the ability of system will be improved in order to provide more accurate matched cases to be the solution to the user.

7.2.4 Percentage Ranking

The solutions are displayed in descending order according to the percentage. System will find ten most similar cases, each which have the rating in percentage that shows the similarity with what user wants. Solution (new matched case) with the highest percentage

ranking shows that I has the most similarity metric with the presented case. With this, it is easier for user to view and to compare each of the solution recommended.

7.2.5 Security

Security in the context of my system is that it has the login part in the main interface for the travel agent that is the system administrator. This means, only the authorize travel agent can log in to the system for adding and deleting data. Authorize travel agent will be assigned specific username and also password that other user do not know. The list of usernames and the passwords assigned are kept in a database called 'pass' that user cannot view.

7.3 System Limitations

Due to the time constraint and the programming language itself, there were some weaknesses in this system. These include:

1) Limited features or scope

   It only provides a few islands to choose, which should be more choices. Besides there should be other feature like attraction type such as art, ethnic or adventure discovery. Other than that, user cannot input their options of number of people going for the holiday (e.g. single, family or party), which is related to type of

room chose. Moreover, the budget considered here does not depend on peak season or off-peak season whereas it should.

2) If a destination were not acceptable by the user, the system would not be able to recommend alternative destinations, taking into account the places that are acceptable and not acceptable by the user. In other words, the third phase of CBR cycle, *Revise*, is not provided.

3) User cannot set priority to the attributes. This means required data that has to be input.

7.4 Future Enhancement

Further development and many new ideas have come about while the system was being implemented but owing to time constraint and other factors, not all of the ideas could be incorporated into the system. For example, the efficiency of the system to recommend the most similar cases as the solutions can be improvised in terms of its accuracies. This means, weighted value should be assigned to all attributes instead of assigning to only one attribute that is budget. The scope of the project which is so confined and restricted that it does not reflect the full requirement of the system and tourism destination itself. The following aspects should be considered as well in future enhancement:

1) Increase features and use bigger scope of domain knowledge

User not only can utilize CBR system but can also access other information such as promotional holiday packages, reservation online with hotels and air plane, interesting destinations, and also about beautiful Malaysia.

2) Use MySQL as database and case base

MySQL provides higher ability to store the cases and it is more efficient to handle huge database.

3) The function of the third phase of CBR cycle, Revise, should be implemented as it is a very important technique in CBR, which allow the user gives feedback and to request other alternatives.

7.5 Conclusion

Recommendation systems provide advice to users about products they might be interested in. Burke distinguishes three types of recommendation systems: collaborative or social filtering; content-based and knowledge-based [8].

Case-Based Reasoning (CBR), as I already commented in the introduction, is a problem solving methodology that faces a new problem or situation by first retrieving a past, already solved similar case, and then reusing that case for solving the current problem. CBR systems typically provide suggestions for a product first asking to the user to specify some preferences, then retrieving from the case base a subset of cases that best match the input description. In general, a case in the memory may represent an item previously suggested or in many existing systems the case base is simply a database of

items. This point out a first difference between my approach and ones those are more conventional. I do exploit both a case base and database. The case base provides complex knowledge about travel products and user navigation preferences, whereas the database of products, provide up-to-date information on the available travel products.

From this project, I managed to learn new programming language like VBscript and Active Server Pages. Although I did not master both of this programming languages, the little bit of new knowledge I got is very much appreciated. Other than that, I learned to use many software tools like Macromedia Dreamweaver and Microsoft Interdev Development. Tourism has become a new interesting subject to me and to incorporate it with the emerging electronic commerce or e-commerce trend is something that we should take seriously into.

The application of CBR to travel and tourism is in a very early stage. Therefore, there are not many CBR applications to this field. I will not consider this system as an excellent success but I would definitely claim it as a fine start for developing better CBR systems in tourism field in the future.

The absence of industry and expert participation is currently one of the constraints faced by this project. However, CBR has been proved an alternative in travel and tourism field and also another alternative for evolving e-commerce implementations in our country through this project.

References

[1] [Fesenmaier *et al.*, 2003] D. Fesenmaier, F. Ricci, E. Schaumlechner, K. Wöber and C. Zanella. DIETORECS: Travel Advisory for Multiple Decision Styles. In *Proceedings of the ENTER 2003 Conference*, pages 232-242, Helsinki, Finland, January 2003.

[2] [Ricci and Werthner, 2002] F. Ricci and H. Werthner. Casebased querying for travel planning recommendation. *Information Technology and Tourism*, 4(3-4):215-226, 2002.

[3] [Werthner and Klein, 1999] H. Werthner and S. Klein. *Information Technology and Tourism – A Challenging Relationship*. Springer Verlag, Wien, New York, 1999.

[4] Agnar Aamodt and Enric Plaza. Case-based reasoning: foundational issues, methodological variations, and system approaches. AI Communications, 7(1):39-59, 1994.

[5] Chin-Liang Chang. Finding prototypes for nearest neighbour classiffier. IEEE Transactions on Computers, C-23(11):1179{1184, 1974.

[6] H. Bunke and B.T Messmer. Similarity measures for structured representations.

In S. Wess, K-D. Althoff, and M. M. Richter, editors, Topics in Case-Based Reasoning, Kaiserslautern, Germany, 1994. Springer-Verlag.

[7] David C. Fallside. Xml schema part 0: Primer. http://www.w3.org/TR/xmlschema-0/, March 2001. W3C Proposed Recommendation.

[8] Robin Burke. Encyclopedia of Library and Information Science, volume 69, chapter Knowledge-based Recommender Systems. Marcel Dekker, 2000.

[9] http://www.ceng.metu.edu.tr/~el26436/Ceng568/CBR-paper.htm

[10] www.ai-cbr.org

[11] www.ecfc.u-net.com/cost/case.htm

[12] www.cbr-web.org

[13] http://www.aiai.edu.ac.uk/links/cbr.html

Appendix

Coding

```
<%
        Dim objConn
        Set objConn = Server.CreateObject("ADODB.Connection")
        objConn.ConnectionString = "FILEDSN=C:\Program Files\Common
        Files\ODBC\Data Sources\travelsolution.dsn"
        objConn.Open
%>

<html>
<head>
<title></title>

<!--#include file="DBConnect.asp"-->
        <U></U>
        <!--#include virtual="/adovbs.inc"-->
        <%
Dim objRS
Dim I           ' Standard looping var
Dim iRecordToDelete ' Id of deleted record
Dim strSQL      ' String variable for building our query
Dim objCleanUpRS
Dim iRecordCount
Set objRS = Server.CreateObject("ADODB.RecordSet")


        iRecordToDelete = Request.QueryString("ID")
        iRecordToDelete = Replace(iRecordToDelete, "'", "''")

        If IsNumeric(iRecordToDelete) Then
                iRecordToDelete = CLng(iRecordToDelete)
        Else
                iRecordToDelete = 0
        End If

        strSQL = "SELECT * FROM tourism WHERE ID=" & iRecordToDelete & ";"
        objRS.Open strSQL, objConn, adOpenKeyset, adLockPessimistic, adCmdText


        If Not objRS.EOF Then
                objRS.MoveFirst
                objRS.Delete adAffectCurrent
```

```
End If


objRS.Close
Set objRS = Nothing


strSQL = "SELECT * FROM tourism ORDER BY ID;"

Set objCleanUpRS = Server.CreateObject("ADODB.Recordset")
objCleanUpRS.Open strSQL, objConn, adOpenStatic, adLockReadOnly,
adCmdText

Response.Write vbTab & "<tr>" & vbCrLf
Response.Write vbTab & vbTab & "<th>ID</th>" & vbCrLf
Response.Write vbTab & vbTab & "<th>package_type</th>" & vbCrLf
Response.Write vbTab & vbTab & "<th>travel_destination</th>" & vbCrLf
Response.Write vbTab & vbTab & "<th>place_name</th>" & vbCrLf
Response.Write vbTab & vbTab & "<th>budget</th>" & vbCrLf
Response.Write vbTab & vbTab & "<th>stay_duration</th>" & vbCrLf
Response.Write vbTab & vbTab & "<th>category</th>" & vbCrLf
Response.Write vbTab & vbTab & "<th>room_type</th>" & vbCrLf
Response.Write vbTab & vbTab & "<th>activities</th>" & vbCrLf
Response.Write vbTab & "</tr>" & vbCrLf

If Not objCleanUpRS.EOF Then
        objCleanUpRS.MoveFirst

        Do While Not objCleanUpRS.EOF
                Response.Write vbTab & "<tr>" & vbCrLf
                For I = 0 To objCleanUpRS.Fields.Count - 1
                        Response.Write vbTab & vbTab & "<td><a
href=""deletetesting.asp?id=" & objCleanUpRS.Fields("id").Value & """>" &
objCleanUpRS.Fields(I) & "</a></td>" & vbCrLf
                Next
                Response.Write vbTab & "</tr>" & vbCrLf
                objCleanUpRS.MoveNext
        Loop
    End If
    Response.Write "</table>" & vbCrLf


    iRecordCount = objCleanUpRS.RecordCount

    objCleanUpRS.Close
```

```
        Set objCleanUpRS = Nothing

    If iRecordCount <= 2 Then
        Set objCleanUpRS = Server.CreateObject("ADODB.Recordset")
        objCleanUpRS.Open strSQL, objConn, adOpenStatic, adLockPessimistic,
    adCmdText

        For I = 1 to 2
            objCleanUpRS.AddNew

            objCleanUpRS.Fields("package_type")
            objCleanUpRS.Fields("travel_destination")
            objCleanUpRS.Fields("place_name")
            objCleanUpRS.Fields("budget")
            objCleanUpRS.Fields("stay_duration")
            objCleanUpRS.Fields("category")
            objCleanUpRS.Fields("room_type")
            objCleanUpRS.Fields("activities")

        Next
        objCleanUpRS.Update

        objCleanUpRS.Close
        Set objCleanUpRS = Nothing
    End If
    %>
        </p>
      </BODY></HTML>

<%@ Language=VBScript %>
<HTML>
<BODY>


</TR>

<TR>
    <TD><%=objRT("Solution_Package_Type") %></TD><TD>  </TD>
    <TD><%=objRT("Solution_Travel_Destination") %></TD><TD>  </TD>
    <TD><%=objRT("Solution_Place_Name") %></TD><TD>  </TD>
    <TD><%=objRT("Solution_Budget") %></TD><TD>  </TD>
    <TD><%=objRT("Solution_Stay_Duration") %></TD><TD>  </TD>
    <TD><%=objRT("Solution_Category") %></TD><TD>  </TD>
    <TD><%=objRT("Solution_Room_Type") %></TD><TD>  </TD>
    <TD><%=objRT("Solution_Activities") %></TD><TD>  </TD>
```

```
        <TD><%=objRT("Rate") & "%"%></TD>

%>
</BODY>
</HTML>

<html>
<head>

<!--#include file="DBConnect.asp"-->
        <!--#include virtual="/adovbs.inc"-->

        <%
Dim objRS,  I, package, destination, placename, PlaceCode
Dim budget, stayduration, category, room, activities
Dim strSQL
Dim objCleanUpRS
Dim iRecordCount
Set objRS = Server.CreateObject("ADODB.Recordset")
objRS.Open "tourism", objConn, , adLockOptimistic, adCmdTable

PlaceCode=Request.Form("txtPlaceCode")
package=Request.Form("txtpackagetype")
destination=Request.Form("txttraveldestination")
placename=Request.Form("txtplacename")
budget=Request.Form("txtbudget")
stayduration=Request.Form("txtstayduration")
category=Request.Form("txtcategory")
room=Request.Form("txtroom")
activities=Request.Form("txtactivities")

If PlaceCode = "" Then
        objConn.Close
        Set objConn = Nothing
        Response.Write "<A HREF='add.asp'>"
        Response.Write "Please Insert  Place Code !"
        Response.Write "</A>"
        Response.End
End If

If package = "" Then
        objConn.Close
        Set objConn = Nothing
        Response.Write "<A HREF='add.asp'>"
        Response.Write "Please Insert  Package Type !"
        Response.Write "</A>"
```

```
        Response.End
End If
If destination = "" Then
        objConn.Close
        Set objConn = Nothing
        Response.Write "<A HREF='add.asp'>"
        Response.Write "Please Insert Travel Destination ! "
        Response.Write "</A>"
        Response.End
End If
If placename = "" Then
        objConn.Close
        Set objConn = Nothing
        Response.Write "<A HREF='add.asp'>"
        Response.Write "Please Insert Place Name !"
        Response.Write "</A>"
        Response.End
End If

If stayduration = "" Then
        objConn.Close
        Set objConn = Nothing
        Response.Write "<A HREF='add.asp'>"
        Response.Write "Please Insert Stay Duration !"
        Response.Write "</A>"
        Response.End
End If
If category = "" Then
        objConn.Close
        Set objConn = Nothing
        Response.Write "<A HREF='add.asp'>"
        Response.Write "Please Insert Category !"
        Response.Write "</A>"
        Response.End
End If
If room = "" Then
        objConn.Close
        Set objConn = Nothing
        Response.Write "<A HREF='add.asp'>"
        Response.Write "Please Insert Room !"
        Response.Write "</A>"
        Response.End
End If
If activities = "" Then
        objConn.Close
        Set objConn = Nothing
```

```
        Response.Write "<A HREF='add.asp'>"
        Response.Write "Please Insert Activities !"
        Response.Write "</A>"
        Response.End
End If
If budget = "" Then
        objConn.Close
        Set objConn = Nothing
        Response.Write "<A HREF='add.asp'>"
        Response.Write "Please Insert Budget !"
        Response.Write "</A>"
        Response.End
End If

objRS.AddNew

objRS("package_type") = Request.Form("txtpackagetype")
objRS("travel_destination") = Request.Form("txttraveldestination")
objRS("place_name") = Request.Form("txtplacename")
objRS("place_code") = Request.Form("txtPlaceCode")
objRS("budget") = Request.Form("txtbudget")
objRS("stay_duration") = Request.Form("txtstayduration")
objRS("category") = Request.Form("txtcategory")
objRS("room_type") = Request.Form("txtroom")
objRS("activities") = Request.Form("txtactivities")
objRS.Update

        strSQL = "SELECT * FROM tourism ORDER BY ID;"

        Set objCleanUpRS = Server.CreateObject("ADODB.Recordset")
        objCleanUpRS.Open strSQL, objConn, adOpenStatic, adLockReadOnly,
        adCmdText

        Response.Write vbTab & "<tr>" & vbCrLf
                Response.Write vbTab & vbTab & "<th>ID</th>" & vbCrLf
                Response.Write vbTab & vbTab & "<th>package_type</th>" & vbCrLf
                Response.Write vbTab & vbTab & "<th>travel_destination</th>" &
                vbCrLf
                Response.Write vbTab & vbTab & "<th>place_name</th>" & vbCrLf
                Response.Write vbTab & vbTab & "<th>place_code</th>" & vbCrLf
                Response.Write vbTab & vbTab & "<th>budget</th>" & vbCrLf
                Response.Write vbTab & vbTab & "<th>stay_duration</th>" & vbCrLf
                Response.Write vbTab & vbTab & "<th>category</th>" & vbCrLf
                Response.Write vbTab & vbTab & "<th>room_type</th>" & vbCrLf
                Response.Write vbTab & vbTab & "<th>activities</th>" & vbCrLf
                Response.Write vbTab & "</tr>" & vbCrLf
```

```
        If Not objCleanUpRS.EOF Then
                objCleanUpRS.MoveFirst

                Do While Not objCleanUpRS.EOF
                        Response.Write vbTab & "<tr>" & vbCrLf
                        For I = 0 To objCleanUpRS.Fields.Count - 1
                                Response.Write vbTab & vbTab & "<td><a
href=""" & objCleanUpRS.Fields("id").Value & """>" & objCleanUpRS.Fields(I)
& "</a></td>" & vbCrLf
                        Next
                        Response.Write vbTab & "</tr>" & vbCrLf
                        objCleanUpRS.MoveNext
                Loop
        End If


        Response.Write "</table>" & vbCrLf

        iRecordCount = objCleanUpRS.RecordCount

        objCleanUpRS.Close
        Set objCleanUpRS = Nothing

If iRecordCount <= 2 Then
        Set objCleanUpRS = Server.CreateObject("ADODB.Recordset")
        objCleanUpRS.Open strSQL, objConn, adOpenStatic, adLockPessimistic,
        adCmdText

        For I = 1 to 2
                objCleanUpRS.AddNew


                objCleanUpRS.Fields("package_type")
                objCleanUpRS.Fields("travel_destination")
                objCleanUpRS.Fields("place_name")
                objCleanUpRS.Fields("place_code")
                objCleanUpRS.Fields("budget")
                objCleanUpRS.Fields("stay_duration")
                objCleanUpRS.Fields("category")
                objCleanUpRS.Fields("room_type")
                objCleanUpRS.Fields("activities")

        Next
        objCleanUpRS.Update
        objCleanUpRS.Close
        Set objCleanUpRS = Nothing
```

```
        End If

        objRS.Update
        objRS.Close
        Set objRS = Nothing
        objConn.Close
        Set objConn = Nothing

        %>

                <HTML>
                <BODY>
                </p>
                <P><U></U></P>
<html>
<head>
<BODY>
<!--#include file="DBConnect.asp"-->
<!--#include virtual="/adovbs.inc"-->

<%

Dim objR, index
index=1
Set objR = Server.CreateObject("ADODB.Recordset")
objR.CursorLocation = adUseClient
objR.Open "temporary", objConn,adOpenStatic, adCmdTable
objR.Sort = "Rate DESC"

Do While Not (objR.EOF OR index=10)
%>
<%
index=index+1
objR.MoveNext
Loop

objR.Close
Set objR = Nothing
objConn.Close
Set objConn = Nothing

%>
<form name="form2" method="post" action="TSD.asp">
```

```
        <div align="center">
          <input type="submit" name="Submit" value="OK">
<HTML>
</div>
</table>
</body>
</html>

<html>
<head>
<HTML>
<BODY>
<!--#include file="DBConnect.asp"-->
<!--#include virtual="/adovbs.inc"-->

<%
Dim objRS, objRT, objR
Dim strpackagetype,intBudget,intstayduration,strtraveldestination, strplacename,
strCategory ,strroomtype, strActivity, intvalue, intvalue2, intvalue3
Dim intberat1, intberat2, intberat3, intberat4, intberat5, intberat6, intberat7, intberat8,
boltest
strpackagetype = Request.Form("txtpackagetype")
strtraveldestination = Request.Form("txttraveldestination")
intBudget  = Cint(Request.Form("txtbudget"))
intstayduration= Cint(Request.Form("txtstayduration"))
strCategory = Request.Form("txtcategory")
strroomtype = Request.Form("txtroomtype")
strActivity = Request.Form("txtactivities")
Set objRS = Server.CreateObject("ADODB.RecordSet")
objRS.Open "tourism", objConn, , , adCmdTable
Set objRT = Server.CreateObject("ADODB.RecordSet")
objRT.Open "case", objConn, , adLockPessimistic, adCmdTable
Set objR = Server.CreateObject("ADODB.RecordSet")
objR.Open "temporary", objConn, , adLockPessimistic, adCmdTable
boltest = False


Do While Not (objRT.EOF OR boltest)


        If ((StrComp(objRT("Package_Type"), strpackagetype, vbTextCompare) = 0)
        AND _
                (StrComp(objRT("Travel_Destination"), strtraveldestination,
                vbTextCompare) = 0) AND _
                (StrComp(objRT("Place_Name"),strplacename, vbTextCompare) = 0)
                AND _
```

```
        (intBudget = objRT("Budget")) AND _
        (intstayduration = objRT("Stay_Duration")) AND _
        (StrComp(objRT("Category"), strCategory, vbTextCompare) = 0) AND _
        (StrComp(objRT("Room_Type"), strroomtype, vbTextCompare) = 0)
        AND _
        (StrComp(objRT("Activities"), strActivity, vbTextCompare) = 0)) Then

        Response.Redirect "temp.asp"

Else

        Do While Not (objRS.EOF)

        If(StrComp(objRS("package_type"), strpackagetype, vbTextCompare) =
        0) Then
                intberat1 = 1
        Else
                intberat1 = 0
        End If

        If(StrComp(objRS("travel_destination"),strtraveldestination,
        vbTextCompare) = 0) Then
                intberat2 = 1
        Else
                intberat2 = 0
        End If

        If(intBudget = objRS("budget")) Then
                intberat3 = 1
                Else If((intBudget < objRS("budget") - 20) OR (intBudget >
                objRS("budget") - 20)) Then
                        intberat3 = 0.8
                        Else If((intBudget < objRS("budget") - 50) OR (intBudget
                        > objRS("budget") - 50)) Then
                                intberat3 = 0.5
                                Else
                        intberat3 = 0
                        End If
                End If
        End If

        If(intstayduration = objRS("stay_duration")) Then
                intberat4 = 1
        Else
                intberat4 = 0
        End If
```

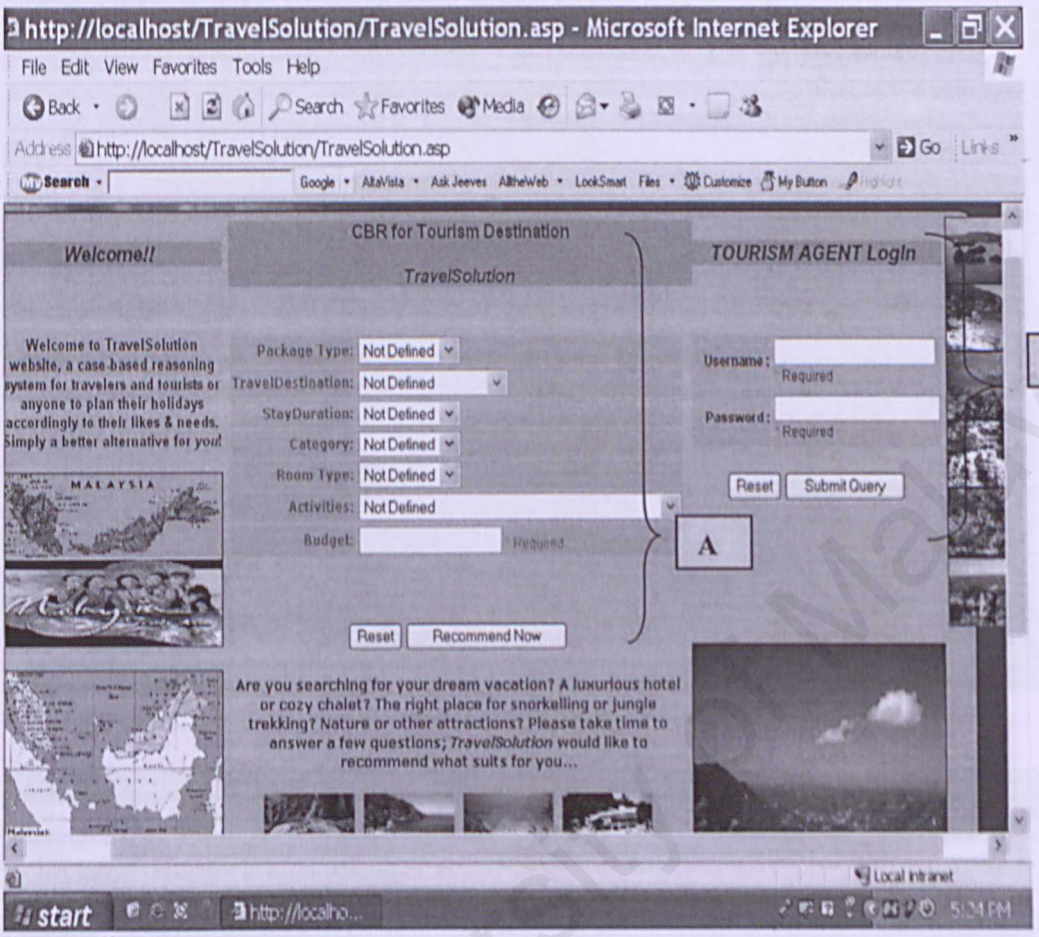
```
If(StrComp(objRS("category"), strCategory, vbTextCompare) = 0) Then
        intberat5 = 1
Else
        intberat5 = 0
End If

If(StrComp(objRS("room_type"), strroomtype, vbTextCompare) = 0)
Then
        intberat6 = 1
Else
        intberat6 = 0
End If

If(StrComp(objRS("activities"), strActivity, vbTextCompare) = 0) Then
        intberat7 = 1
Else
        intberat7 = 0
End If

        intvalue2 = intberat1 + intberat2 + intberat3 + intberat4 + intberat5
        + intberat6 + intberat7
        intvalue3 = intvalue2/7
        intvalue = intvalue3*100


        objRT.AddNew
        objRT("Package_Type") = strpackagetype
        objRT("Travel_Destination") = strtraveldestination
        objRT("Place_Name") = strplacename
        objRT("Budget") = intBudget
        objRT("Stay_Duration") = intstayduration
        objRT("Category") = strCategory
        objRT("Room_Type") = strroomtype
        objRT("Activities") = strActivity
        objRT("Solution_Package_Type") = objRS("package_type")
        objRT("Solution_Travel_Destination") =
        objRS("travel_destination")
        objRT("Solution_Place_Name") = objRS("place_name")
        objRT("Solution_Place_Code") = objRS("place_code")
        objRT("Solution_Budget") = objRS("budget")
        objRT("Solution_Stay_Duration") = objRS("stay_duration")
        objRT("Solution_Category") = objRS("category")
        objRT("Solution_Room_Type") = objRS("room_type")
        objRT("Solution_Activities") = objRS("activities")
        objRT("Rate") = intvalue
```

```
                objRT.Update

                objR.AddNew
                objR("Package_Type") = objRT("Solution_Package_Type")
                objR("Travel_Destination") =
                objRT("Solution_Travel_Destination")
                objR("Place_Name") = objRT("Solution_Place_Name")
                objR("Place_Code") = objRT("Solution_Place_Code")
                objR("Budget") = objRT("Solution_Budget")
                objR("Stay_Duration") = objRT("Solution_Stay_Duration")
                objR("Category") = objRT("Solution_Category")
                objR("Room_Type") = objRT("Solution_Room_Type")
                objR("Activities") = objRT("Solution_Activities")
                objR("Rate") = objRT("Rate")
                objR.Update


%>


<%              objRS.MoveNext
                Loop

                boltest = True
                Response.Redirect "temp.asp"
        End If

objRT.MoveNext
Loop

objRT.Close
Set objRT = Nothing
objRS.Close
Set objRS = Nothing
objConn.Close
Set objConn = Nothing

%>
</BODY>
</HTML>

<%@ Language=VBScript %>
<HTML>
<BODY>
<!--#include file="DBConnect.asp"-->
```

```
<!--#include virtual="/adovbs.inc"-->
<%
Dim objRS, bolFound, strUsername
strUsername = Request.Form("username")

If strUsername = "" Then
        objConn.Close
        Set objConn = Nothing
        Response.Write "<A HREF='TravelSolution.asp'>"
        Response.Write "Please Insert Username"
        Response.Write "</A>"
        Response.End
End If

Set objRS = Server.CreateObject("ADODB.RecordSet")
objRS.Open "pass", objConn, , , adCmdTable
bolFound = False

Do While Not (objRS.EOF OR  bolFound)
        If (StrComp(objRS("username"), strUsername, vbTextCompare) = 0) Then
                BolFound = True
        Else
                objRS.MoveNext
        End If
Loop

If Not bolFound Then
        objRS.Close
        Set objRS = Nothing
        Response.Write "<A HREF='TravelSolution.asp'>"
        Response.Write "Invalid Username<P>"
        Response.Write "</A>"
        Response.End
End If

If Not (StrComp(objRS("password"), Request.Form("password"), vbBinaryCompare) =
0) Then
        objRS.Close
        Set objRS = Nothing
        objConn.Close
        Set objConn = Nothing
        Response.Write "<A HREF='TravelSolution.asp'>"
        Response.Write "Invalid Password<P>"
        Response.Write "</A>"
        Response.End
End If
```

```
Response.Redirect "add.asp"
%>
</BODY>
</HTML>

<HTML>
<BODY>
<!--#include file="DBConnect.asp"-->
<!--#include virtual="/adovbs.inc"-->


<%

Dim objR

Set objR = Server.CreateObject("ADODB.Recordset")
objR.CursorLocation = adUseClient
objR.Open "temporary", objConn,adOpenStatic, adCmdTable

Do While Not objR.EOF

        If (objR("ID") > 0) Then
                objR.Delete
        Else
                Response.Write("tipulah")
        End If



objR.MoveNext
Loop


objR.Close
Set objR = Nothing
objConn.Close
Set objConn = Nothing
%>
</body>
</html>
```

User Manual



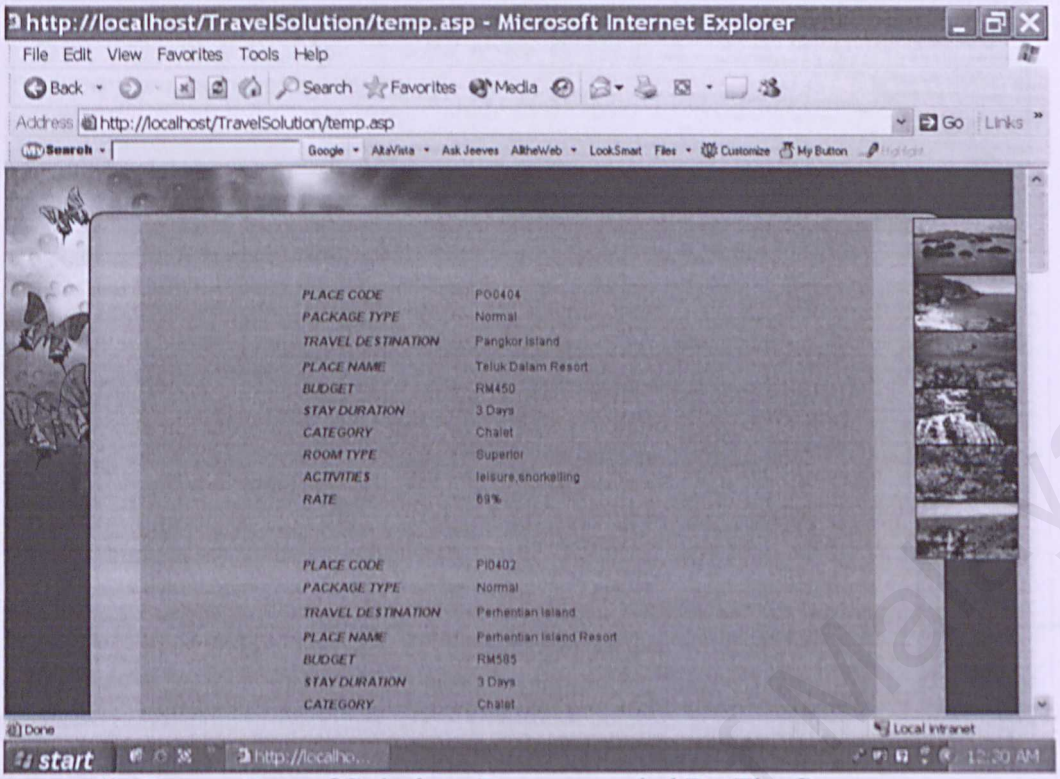
Appendix 1

A

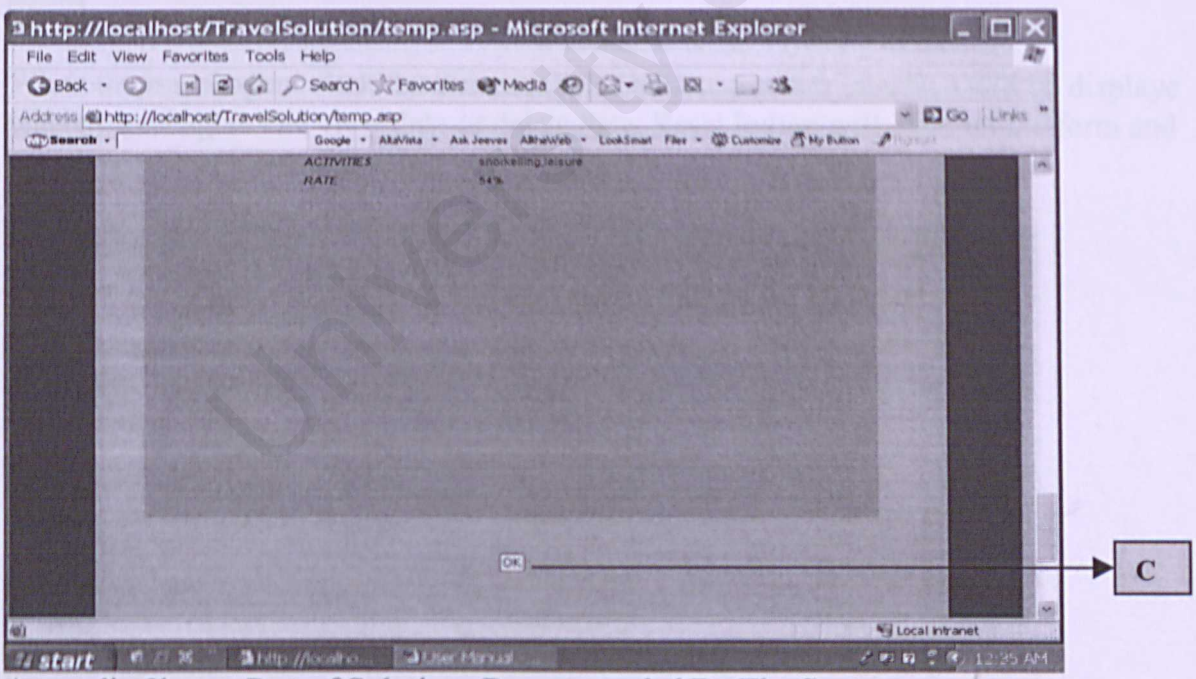
This part is where the users can straight away using the CBR system by selecting their preferences from the options list and to input their budget limit.
Users can click the Reset button to modify their options or they can click the Recommend Now button to view the solutions provided which will be displayed on another page.

B

This part is where the authorize travel agent or the system administrator log in to their usernames and passwords to add or delete the data. When clicking on the Submit Query button, another page will be displayed and travel agent can add in data online and straight away view the added new data in the form of table. From here, if they made mistake adding the wrong data, they can delete it from the table.



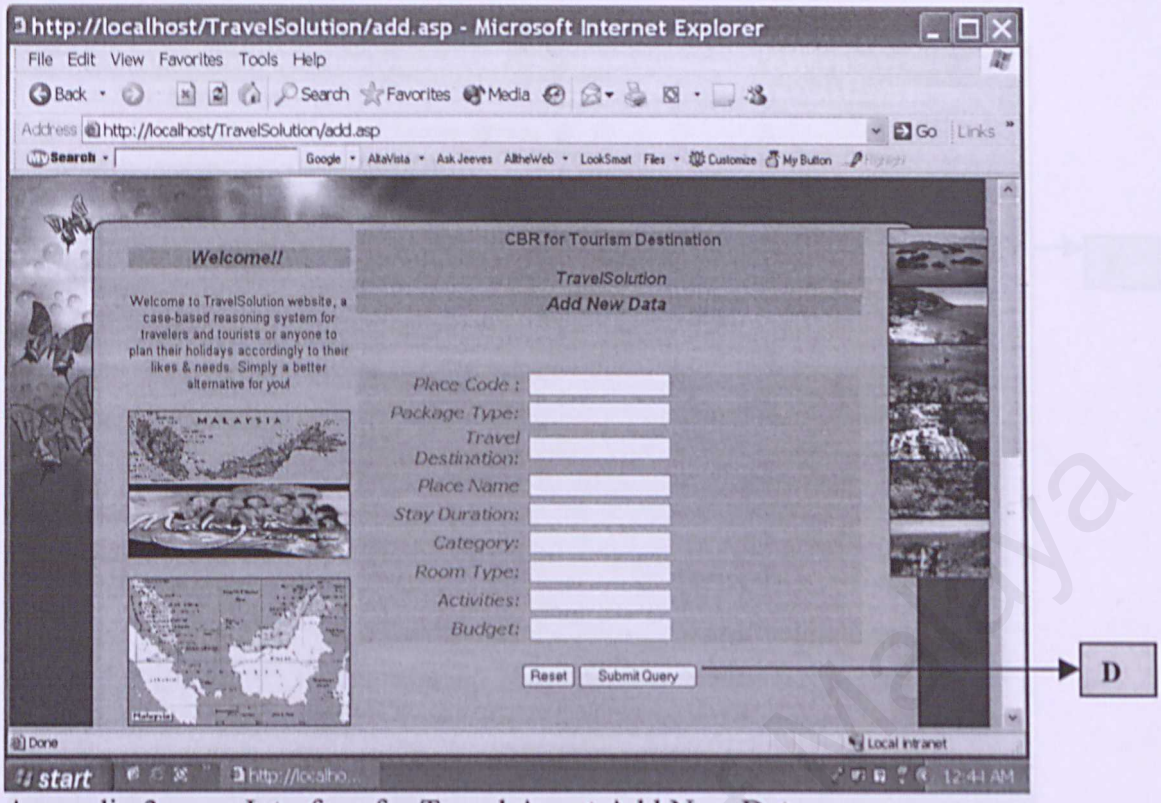
Appendix 2a Part of Solutions Recommended By The System



Appendix 2b Part of Solutions Recommended By The System

C

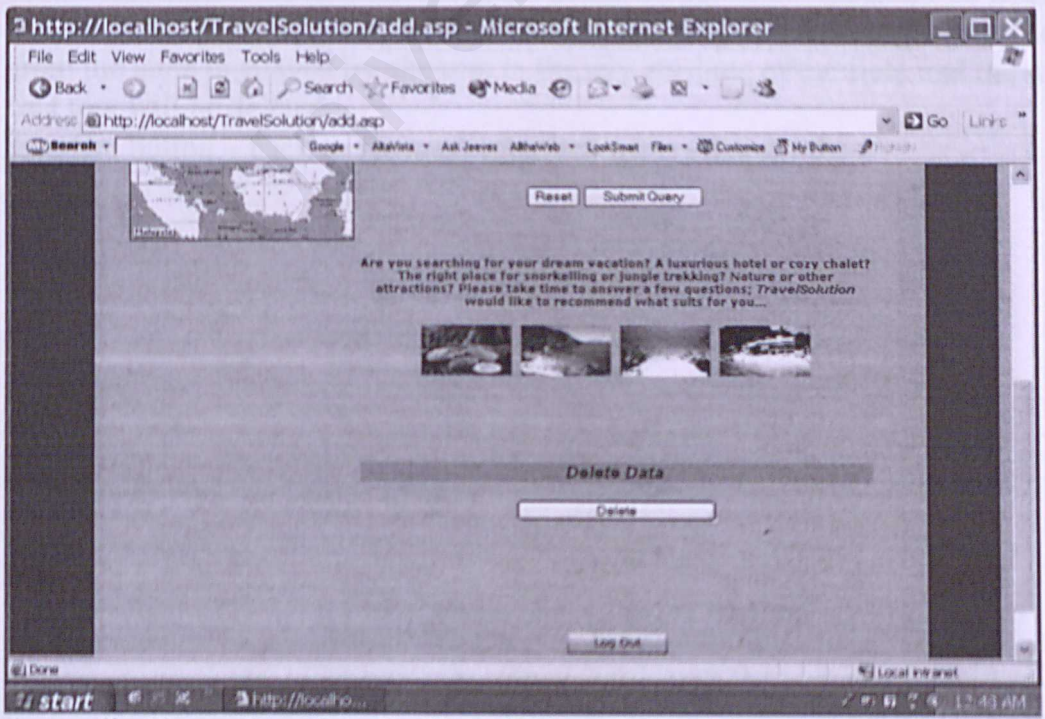
When users click OK button, they will go to another interface and from there they will go back to the main interface when clicking on the picture.



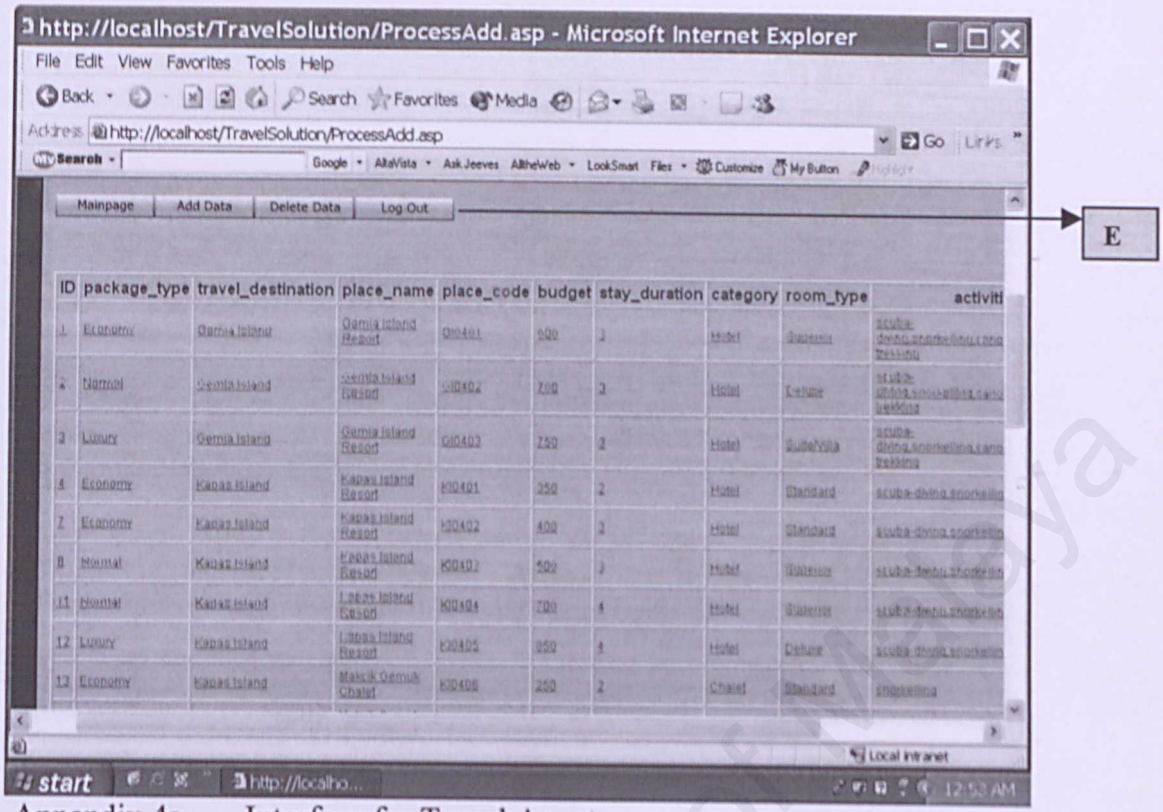
Appendix 3a Interface for Travel Agent Add New Data

D

When the travel agent click the Submit Query button, another interface will be displaye where travel agent can view data or delete data. Reset button will clear up the form and travel agent can input new data again.



Appendix 3b Interface for Travel Agent Delete Data



Appendix 4a Interface for Travel Agent

E

There are:
Mainpage button – back to the main interface when click it.
Add Data button – back to the interface as shown above as Appendix 3a when click it.
Delete Data button – by clicking this button, travel agent can straight away delete the data from the table displayed by clicking to the any attribute of the field and the whole field in that row will be deleted.
Log Out button – back to the main interface when click it.