ADDIS ABABA UNIVERSITY

SCHOOL OF GRADUATE STUDIES

COLLEGE OF NATURAL SCIENCES

SCHOOL OF INFORMATION SCIENCE

APPLICATION OF CASE-BASED REASONING IN LEGAL CASE MANAGEMENT: AN EXPERIMENT WITH ETHIOPIAN LABOR LAW CASES

By

Abebaw Alem

Addis Ababa University

Addis Ababa, Ethiopia

June, 2014

ADDIS ABABA UNIVERSITY
SCHOOL OF GRADUATE STUDIES
COLLEGE OF NATURAL SCIENCES
SCHOOL OF INFORMATION SCIENCE

Application of Case-Based Reasoning in Legal Case Management: An Experiment with Ethiopian Labor Law Cases

A Thesis Submitted to the School of Information Science, Addis Ababa University, in Partial Fulfillment of the Requirements for the Degree of Master of Science in Information Science

By
Abebaw Alem

ADDIS ABABA UNIVERSITY
SCHOOL OF GRADUATE STUDIES
COLLEGE OF NATURAL SCIENCES
SCHOOL OF INFORMATION SCIENCE

Application of Case-Based Reasoning in Legal Case Management: An Experiment with Ethiopian Labor Law Cases

By
Abebaw Alem

Name, responsibility and signature of board members for this work approval**:**

Name	**Responsibility**	**Signature**	**Date**
Dr. Martha Yifru	Advisor	----------	----------
Dr. Million Meshesha	Examiner	----------	----------
Mr. Ermias Abebe	Examiner	----------	----------
Ms. Lemlem Hagos	Chairperson	----------	----------

Acknowledgements

First, I would like to give a special gratitude to the Gracious God and His Mother Saint Marry in helping me to finish my courses in the field of study and to do this final thesis for the MSc degree requirements.

Second, my golden appreciation is forwarding to my advisor Dr. Martha Yifru who has provided her precious and pertinent knowledge about the study from the starting to the end. I haven't words to express her patient and kindness with great motivations when ambiguities faced to me during the research work. Without her continues help and helpful comments, this work could not have been possible to be completed. Once again, I thank Dr. Martha so much!

Third, my grateful thanks go to Dr. Million Meshesha who has provided his precious knowledge of courses, unforgettable advices, comments, and important directions to identify the problem before preparing the research proposal development. Dr. Million is my role model forever!

Fourth, I would like to thank the FSCE workers especially Ato Almaw Wole and Ato Teshager G/Slassie (lawyers), Ato Geresu Gemeda (ICT professional), W/ro Mulualem Gizaw (Data base Administrator) and W/ro Abeba Moges (record officer) who have provided advices and given any relevant information that I want to my work with their kindness.

Fifth, I would appreciate the six selected system evaluators who are students of AAU law school for their willingness and time devotion to evaluate the system performance. Sixth, my unforgettable acknowledge is forwarding to Meseret Ayano who motivated me how to work my tasks and provide educational materials for my work. I would also like to thank all my classmates who share their ideas during teaching-learning process of the course lessons and then after for the research work.

Eighth, I would like to thank Debre Tabor University (DTU) for giving a chance to pursuit my MSc degree at AAU with financial support for educational requirements to complete the program. The last but not the least thank is going to the Information Science department staff who are helping me in different ways in different affairs.

Table of Contents

List of Figures

List of Tables

List of Abbreviations

AAU	Addis Ababa University
AI	Artificial Intelligence
CBR	Case Based Reasoning
CCMS	Court Case Management System
CSV	Comma Separate Value
DL	Description Logic
Four “Re’s”	Retrieve, Reuse, Revise and Retain
FSCE	Federal Supreme Court of Ethiopia
GAIA	Group of Artificial Intelligence Applications
JCOLIBRI	Java Cases and Ontology Libraries Integration for Building Reasoning Infrastructures
KA	Knowledge Acquisition
KBS	Knowledge Based Systems
KE	Knowledge Engineering
LCM	Legal Case Management
LCMS	Legal Case Management System
ODBC	Open Database Connection
PSM(s)	Problem Solving Methods
RBR	Rule Based Reasoning
SQL	Structured Query Language
XML	Extensible Markup Language

Abstract

The labor law domain is an important selection area in AI that is a field to make a machine (computer) simulate human like behavior to enhance consistency, reliable and timely decision making. KBS is one of the major subfield of AI that uses expert (knowledge) to solve a specific problem. CBR is one of the important applications of AI and/or KBS for manipulating previous knowledge (cases) in different areas. It is a problem solving paradigm that uses earlier experiences to solve new problems and is useful to humans when knowledge is incomplete and/or evidence is sparse in the domain of law. Thus, major goal of this study is to design a prototype application of case based reasoning for legal case management in the domain of Ethiopian labor law context at the Federal Supreme Court. Fifty legal texts were selected through document review and discussions from FSCE. These texts then are organized in attribute-values dimension and converted to plain text file for building case base. Attributes are represented using the CBR developmental tool jcolibri framework for implementing the four "Re's" of CBR application tasks (Retrieve, Reuse, Revise and Retain). The performance of the prototype system is evaluated using the statistical analysis (precision and recall) and user acceptance (satisfaction) techniques. Thus, the system has a recall of 71% and precision of 86% performance. Moreover, the system was accepted by domain experts 86% of the time.

*It is concluded that the prototype system is applicable in judicial application to provide fast and quality services. However, general knowledge explanations are lacked and major challenges in CBR for this study when similar cases are not in the case base to solve a problem. So, investigating a prototype system in law domain with integration of CBR and RBR approaches with explanation facility for future investigation is important. Moreover, managing any types of law in a separate section might need more resources. So, an "**all in one**" CBR applications for the three law parts (criminal, labor and civil laws) is recommended to manage large cases and reduce the burdens.*

***Keywords**: Case-based Reasoning (CBR), Legal Case Management, Labor Law*

CHAPTER ONE

INTRODUCTION

1.1. Background of the Study

Human experts not only need the computer systems to be manipulated by them but also need the systems to perform tasks like human experts. As the human intensive needs, knowledge based expert system is required in the globe. Expert system is a branch of artificial intelligence (AI) and a computer system that uses a representation of human expertise in a specialist domain (lawyer) to perform functions similar to those normally performed by a human expert in that domain (Laudon and Laudon, 1997). It is required to mimic the human behavior like to solve complex problems. Thus, legal expert system is required for lawyer with in case based reasoning (CBR) applications because the humans in general and lawyers in specific have incomplete domain knowledge at hand.

The field of AI attempts not just to understand but also to build intelligent entities. AI currently encompasses a huge variety of subfields, ranging from general-purpose areas, such as learning and perception to such specific tasks as playing chess, proving mathematical theorems, writing poetry, and diagnosing diseases so it is truly a universal field (Russell and Norvig, 2003). Therefore, AI is interesting research issue in any domain areas in general and in law domain specifically. Knowledge base system (KBS) is part of AI that contains the previous knowledge stored about human expert domain in the case base to make them interact with computer programs (systems). A DB of past experience, called a case-base, is constructed and then used to solve new problems through CBR application processes of retrieval, adaptation, review, and memorizing. Each experience is stored within a case base, which usually represents a description of the problem, the solution, and an assessment of the outcome of using the case i.e. reviewing. For retrieving the cases within the case base, they should be representing and indexing in structural organization with dimensions and values.

According to Aamodt and Plaza (1994), KBS can be developed using CBR, rule based reasoning (RBR), hybrid (CBR and RBR) based reasoning and ontology (semantic) based reasoning. From

these well known KBS methods, CBR is more popular to the law domain and selected in order to analyze legal cases in this study.

As Aamodt and Plaza (1994) stated, CBR has blown a fresh wind and a well justified degree of optimism into AI in general and knowledge based decision support systems in particular. CBR is very hot subfield of AI which is growing rapidly. Therefore, according to Aamodt and Plaza, (1994), CBR supports the capability of reasoning and learning in advanced decision support systems. Specifically it is a reasoning paradigm that exploits the specific knowledge collected on previously encountered and solved situations, which are known as cases. In this research, the cases are the labors legal texts that stored in paper form from Federal Supreme Court of Ethiopia (FSCE) under the rules of labor law context.

Today, CBR has been applied in different domain areas such as Law, medicine (health), management and others (Ashley and Rissland, 2003). CBR is useful to human, and machines that know a lot about a task and domain. It gives to them a way of reusing reasoning that they have done in the past. In this case both the human and machine can learn cases from the prior knowledge retrieved from the case base. This study is, therefore, dealing with application of CBR for legal case management in Ethiopian labor law context. Because at present, managing legal cases in the court is the most difficult task as cases are increasing from time to time.

Legal case management (LCM) is a subset of law practice management and covers a range of approaches and technologies used by law firms and courts to control knowledge and tasks for managing the life cycle of a case more effectively in CBR applications. LCM in this study is responsible to retrieve previously retained legal cases, revise the retrieved legal cases and retain a new legal case based on the user's general knowledge about the problem at hand.

As Ashley (1987) stated the law as an excellent domain to study CBR since by its very nature, the following points can be pointed out:

- espouses a doctrine of precedent in which prior cases are the primary tools for justifying legal conclusions; and
- employs precedential reasoning to make up for the lack of strong domain models with which to reason deductively about problem situations.

“The law” is a learning system and there are interesting relations between legal reasoning and learning. The law is also a paradigm for adversarial CBR; there are "no right answers", only arguments pitting interpretations of cases and facts against each other (Ashley, 1987).

Legal argument is a paradigm of adversarial CBR. Legal decision-making may seem arbitrary and chaotic, but, with its emphasis on case precedent, the law is an organized chaos.

Law is composed of rules and cases. According to Ashley and Rissland (2003), the legal rules come from a variety of sources: legislatures (government) may fashion them in statutes, agencies may adopt them as regulations, courts may use them as rules for deciding current cases and summarizing the meaning of past cases, and practitioners use them to distill past experience.

Everyone has the right to an effective remedy by the competent national courts for acts violating the fundamental rights granted by the constitution. It is also enshrined in Article 37(1) of the Constitution of the Federal Democratic Republic of Ethiopia (FDRE) that every person has the right to bring justifiable disputes to, and to obtain a decision or judgment by a court of law or where appropriate by another body with judicial power.

A constitution is the basic character for a government; it described that government and the relationships of its different parts of one another. A constitution also limits government powers and describes the basic rights of citizen with which the government cannot interfere.

Law, besides the constitution, is a form of discourse, midway between the “real” and the “ideal,” which both constructs and reflects the social and political realm or situation. Law is not just specific regulations; it is also an interpretive language that establishes expectations of what is considered legal, honorable, rational, and objective. In the FSCE, there are three major categories of law: civil law, criminal law and labor law. To this research, labor law has been taken as a domain area with legal cases in the CBR applications.

Law is systems of rules that govern a society with intend of maintaining social order, upholding justice and preventing harm to people and property. Law systems are often based on ethical or

religious principles and are enforced by justice systems such as the courts. In Ethiopia in Article 79 (1) of Judicial Powers, courts are divided into state and federal courts. In both courts; Supreme Court, high court and first instance court are established and implemented. Among these institutional services, the Federal Supreme Court (FSC) is selected because of the following reasons:

- According to Article 80 (1) of Concurrent Jurisdiction of Courts, the Federal Supreme Court shall have the highest and final judicial power over federal and state matters. Thus, the unsolved cases appeared in the lower courts (high court and first instance court) can be forwarding to this court as a final decision solving technique with its power.
- Previously solved cases are easily accessible in this court which is vital to develop the prototype CBR system regarding to the labor law domain.

The FSCE was established in 1975 for the purpose of providing fair justice to its customers whose rights are violating by other bodies. Within this court, there are more than 10,000 solved labor cases since 1997 E.C as found in DB generated reports.

Labor rules have been established in different proclamation numbers in different times. The major one is under labor proclamation No. 377/2003 which ensures that labors (workers-employers) relations govern the basic principles of rights and obligations to maintain industrial peace and work in spirit of harmony and cooperation towards the all-round development of the country. Employer means a person or an undertaking that employs a person while employee or worker means a person who has an employment relationship with the employer in accordance with the proclamation rules and regulations of the labor law. These rules have to be implemented by courts that facilitate the enhancement of justice.

Labor law has the generic aim of safeguarding employees and employers in the work environment. Such objectives express government policies and the principles which bring labor regulations into existence (Ethiopia, 2002).

Lawyers have no choice but to argue from cases. They argue that a rule of law applies to a dispute by comparing it to similar prior cases.

In CBR terminology, a case usually denotes a problem situation and a previously experienced situation. It has been captured and learned during problem solving or decision making process and it can be reused in solving of future problems. When it is used in later to solve a problem, it is referred to as a past case, previous or earlier case, stored case, or retained case. As a result, this research is aimed at using the retained case to solve new similar case in the four important cycles of CBR application.

The reasoning process can be summarized using the following four basic steps. These are known as the CBR cycle or four ‘‘Re’s’’ CBR tasks such as: retrieve, reuse, revise and retain (Aamodt and Plaza, 1994). The procedure is: first **retrieve** the most similar case(s), with respect to the current input situation, contained in the case repository, which is known as the case base. Then, **reuse** retrieved cases as solution in order to solve the new problem. This is followed by **revising** the proposed new solution (if it is considered necessary). Finally, **retain** the current case for possible future problem solving.

Therefore, a new problem is solved by retrieving one or more previously experienced cases, reusing the case in one way or another, revising the solution based on reusing a previous case, and retaining the new experience by incorporating it into the existing knowledge-base (case-base).

CBR is a reasoning paradigm which consists in solving new problems by adapting solutions of previously solved problems. This process is supported by different valid knowledge used to reason on different cases. Knowledge (employee or human resource) management in CBR is a difficult problem both in public and private organizations in Ethiopia as seen from different organizations. Thus, disputes between employee and employer can be raised which needs to be solved by the labor law. This leads to the mediators, lawyers, to solve these disputes.

Therefore, lawyers need to search cases manually from their sources to solve the disputes which may be taking a long period of time. As a result, the researcher has intended to develop a prototype CBR model that retrieves cases for an input query from the case base to solve the cases happened in the right way within a short period.

Nowadays in information age, retrieving the relevant information (cases) from the source is an important technique that saves time and cost. Legal case retrieving in the domain of law is one of the most important applications in CBR that enables lawyer to retrieve cases from the case base and learn from it to suggest solutions to problems or queries. However, the lawyers in Ethiopian, in FSC specifically, have obligated to use the manual system retrieving technique other than using advanced technology retrieving in a case base. According to Ethiopia (2002), the purpose of legal research is to help a client solve a legal dispute, prevent such a dispute from arising, or prevent the dispute from getting worse. To solve these issues, lawyers have to search solutions to cases or problems through a wealth of case law and doctrinal writings to find relevant guidance on the problem case at hand.

The driving force behind case-based methods has to a large extent come from the machine learning community, and CBR is also regarded as a subfield of machine learning. It favors learning from experience, since it is usually easier to learn by retaining a concrete problem solving experience than to generalize from it. Effective learning in CBR requires a well worked out set of methods 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.

1.2. Statement of the Problem

In this globalization age, industrial organizations are enormously increasing in fast manner in the real world in general and Ethiopia in specific. To make them effective and efficient, there is a need with skilled man powers who could be employed in either of any organizations. But depending on different situations (such as mistreatment of the employees' right), disputes between these employees and employers might be created and the dispute might come to the court to get right solution for the case or problem. To solve these cases, judges have to receive them from the applicants with a form and set a solution by examining each case and the solved cases might be recorded and stored in the FSCE in manual format.

However, reusing the previous solved cases later to solve a problem at hand is difficult and time consuming for judges to search them manually. Thus, providing fast services are impossible in the area due to lack of accessing relevant previous solved cases immediately at the present time.

This is because the lawyers have to use manual searching technique of similar cases to solve a problem of labors. A number of cases are increasing enormously from time to time and lawyers face a problem in searching these vast cases manually. This requires more time, proves to errors and inconsistency to solve a problem.

Thus, managing past cases in the FSCE is so difficult since cases from time to time are increasing enormously. As a result, providing quality services is questionable and time consuming. This problem could be solved if AI system like CBR system (expert system) is developed. However, this system has never been used in law decision support systems in FSCE although it is important to manage past cases and supporting the lawyers.

It is important to employ the appropriate technology that enables the lawyers to access previously solved cases easily. The purpose of computer systems in the modern era is able to support human experts to solve complex problems for a specific task in a given organization. The same is true to the Ethiopian Supreme Court that a CCMS (Court Case Management System) is used. This system was developed by CyberSoft or Cyberaddis Company and deployed in the Supreme Court. But it is simply used to record clients' information and generate the report.

Today, many application scenarios demand advanced technologies that offer intelligent support to the user when searching information in knowledge society. CBR, one of the advanced technologies, is a paradigm for combining problem-solving and learning mechanism to use the earlier knowledge in labor law domain. It is able to use the specific knowledge of previously experienced, concrete problem situations (cases).

As stated by Ashley and Rissland (2003), law is one of the application domain areas of CBR. Different scholars have investigated different out comes in the globe. Such as: Hypo (Ashley, 1987; 1990, 1991), CABARET (Rissland & Skalak, 1991), GREBE (abbreviated for, Generator of Recursive Exemplar-Based Explanation) (Branting, 1991; 1999), CATO (Aleven, 1997; 2003), BankXX (Rissland, Skalak, et al. 1996), and Split-Up (Zeleznikow, Stranieri, et al., 1995-1996) have been investigated in the world using CBR model (see section 2.7).

Likewise, in our country, the Amharic Legal Precedent Retrieval prototype system has been done by Ethiopia (2002) in the domain of labor law at FSCE. However, the research was limited to

show only retrieving similar cases for the current problems or queries with the CBR tool CBR-Works. This ("only retrieving similar cases") is a motivational issue for the current work. Thus, the researcher has focused to extend Ethiopia's work by implementing the four "Re's" applications of CBR: Retrieving, Reusing (adaptation), Revising (evaluation) and Retaining (learning and storing).

In general, the major motivation to identify this research has the following main focuses. The first is to enable the four "Re's" of CBR applications for legal case management as a continuation of Ethiopia (2002) who applied CBR in legal precedents retrieval in labor law domain. The second motivational reason is to enable effective case and time management habits using the CBR applications in providing quality services. As mentioned earlier, a case takes a long period of time to be solved by the lawyers who need ample time to search solved cases manually and analyze decisions for the problems. This affects the quality service in the FSCE. Hence, pursue of the pragmatic solution the researcher redirects the AI research. Because an intelligent learning approach from the retained cases in the case base to solve the problem is an important efficient solution in saving time and producing quality services with effective case management.

Therefore, the researcher in this study has been motivated to explore a prototype CBR system called LCMS. LCMS could be used as a human (manipulating revise application) and machine (manipulating retain application) learning approach. This is done with the Human-machine Interaction approach using the GUI for better management of prior cases. In this situation, applying the four "Re's" of the CBR applications optimizes the decision capacity and time utilization for lawyers to manage cases.

Generally, this study is expected to answer the following basic research questions.

- What are the relevant cases and cases types required for designing CBR in the area of legal law?
- How can the labor legal cases be represented and indexed/mapped with attributes using jcolibri to manage cases from the case base in CBR applications/tasks?
- Is it possible to implement the four "Re's" of CBR tasks using jcolibri framework in managing legal cases?

1.3. Objectives of the Study

This study has both the general and specific objectives which are outlined below.

1.3.1. General objective

The general objective of the study is to develop a prototype CBR system that manages CBR tasks of retrieving, reusing, revising and retaining cases in the domain of Ethiopian labor law.

1.3.2. Specific Objectives

To address the aforementioned general objective and basic research questions, the researcher proposed the following specific objectives.

- To review different global and local researches to understand the basic concept, principle and technologies of cased-based reasoning and labor law.
- To collect the previously solved knowledge (cases) by reviewing documents and with discussion among staffs and domain experts in FSC.
- To represent and index/map legal cases with relevant attributes using jcolibri to manage them from the existed case base in CBR.
- To develop a prototype CBR system, LCMS, that manages legal cases using the four "Re's" of CBR tasks with jcolibri.
- To test and evaluate the performances of the designed prototype LCMS.

1.4. Scope of the Study

The Scope of this study has focused on the application of CBR for legal case management under the Ethiopian labor law at the FSCE. The 50 legal cases collected from the court are implemented with the four "Re's" of CBR applications such as retrieving, reusing, revising and retaining. There are three common fundamental laws implemented in the court: civil, criminal and labor laws but in this research only labor law is considered due to the need of ample time for a better analysis of all or both laws with their corresponding case retrieval.

There are a number of case types under this labor law domain. *"yeseratenya qnesa, yalagbab kesra mesenabet, Yedemwoz ch'imari, t'qmat'qm, yesra wul mequaret', yeamet ereft kfya, yalagbab yesra wul mequaret'ena lyu lyu kfyawoch, yet'ureta abel, yedereja edget, yewuzf demewez kfya, yalagbab yesra wul mequaret', yesra snbt kfya, yedemewez kfya, discipline, yetrf gize kfyana t'qmat'qm, demewez, yewuklna slt'an, yesra bota zwuwur, yet'ureta abel, yalagbab yesra snbtna wuzf demewez and yesra wul mequaret"* case types are used for this study. Because these case types are commonly adopted in any industrial organizations currently.

1.5. Limitations of the Study

This study has faced different constraints (limitations) while working the study. Theses could affect the prototype system to get its better performance. The major limitations or constraints are discussed as follows:

- Although there are more than 10,000 labor legal cases in the court, this research is limited use only 50 cases due to the time constraints and allowing more cases is not permitted.
- Case is a pillar to this research. In order to get legal cases, the researcher planed to discuss with lawyers to gather tacit knowledge in addition to the past cases, to check the relevance of the selected attributes in the FSC and to evaluate the performance of the prototype system by them. However, the lawyers were so busy to be consulted except two of them. This was a trait that the required data (tacit knowledge) in the domain area needs deep analysis for cases as expected. As a result, due to the time constraint and unavailability of lawyers, the technical test and evaluation is performed with small amount (6) test cases by 6 law department students at AAU.
- The legal cases are solved with the local language Amharic. But this language is not compatible to the CBR developmental tool jcolibri accessibility file formats (.csv and .xml) in building the case base. This was a headache to the researcher. So transliteration of legal text into English was the last option.
- The prototype CBR system, LCMS, is developed in showing how to develop legal reasoning system for enabling lawyers using CBR application. However, it is tough to find a fully appropriate one in practical context in the domain area.

1.6. Significance of the Study

The researcher expected that the application of CBR for legal case management could provide different benefits to the target populations in the domain area identified. The researcher is the first beneficiary by fulfilling requirements in the academic program. After evaluation of the results found in this research, the following four major stakeholders could be beneficiary.

- The prototype LCMS, would be applied in FSCE a base to develop legal expert system for general application to satisfy its clients. So, this prototype system could be initials to develop the real legal expert system in the law domain in CBR applications to manage large cases easily.
- The lawyers can manage the stored previous cases using the case base to solve a problem raised at the moment without spending more time. Therefore, the research output could help lawyers to provide adequate and timely advisory services to their clients within a short period of time. This enables the service delivery improvements in the court.
- At the same time, labors (both the employers and the employees) can get the right services provided by the lawyers in solving their cases (problems). This enables to labor that they save time and cost if lawyers solve their cases by using the system in a short period.
- Moreover, the research might be also referred by future researchers in the domain area.

1.7. Research Methodology

This section aims at describing a research methodology to supply with the means and knowledge making operational activities to perform the research through CBR applications. The methodology refers to reviewing literatures, procedures, tool and techniques. This research is therefore designed to develop prototype CBR system to manage legal cases in the Ethiopian labor law context using jcolibri tool. To do so, different research methodology were used. These are presented in the following sections.

1.7.1. Literature Review

For the successful completion of this study, different global and local researches have been thoroughly reviewed starting from the beginning of the study until its completion. Journal articles, proceeding papers, conference papers, manuals, reports, books and Internet resources are consulted.

1.7.2. Data Sources and Sample Collection

The researcher has selected the FSCE, because the other courts are under this court controlling and these lower level courts' cases are accessed in manual paper form in the FSCE.

The most important source of data in a CBR system is the set of previous solved cases at the FSCE. There are more than 10,000 legal cases stored in the court since 1997 E.C. These cases were documented in hard copy (recorded in paper) with Amharic language. 50 legal text cases from the total were collected from the court by reviewing the decided documents that deals about the labor legal cases. The main reasons for taking only 50 legal cases from the large cases is all the recorded documents are not allowed to give the researcher by the workers since it is tiresome. Moreover, organizing and representing more legal texts with extracting the relevant features need ample time.

Therefore, these factors limit to take 50 sample legal texts from about 10,000 recorded documents in this study from the court. In addition to reviewing decided documents, two lawyers were consulted to check the validity of the selected attributes.

1.7.3. Case Representation

The selected 50 cases are represented in the attribute-value dimensions with the help of Microsoft excel application software. The selected Amharic cases were converted in to text format and but jcolibri is not applicable for these texts. As a result, representing a general case description language was a difficult task because the CBR systems with jcolibri have different case representational requirements such as .csv files, xml files and plain text files. But, the hand written Amharic text was not possible to be presented in either of these files. Therefore, the researcher has transliterated the Amharic texts into the English and then represent them in attribute-values dimensions with Microsoft excel 2007 application software. This .xlsx file is

then saved in CSV (comma delimited) (*.**csv**) to convert in to plain text file. This plain text file is used by the jcolibri connector for the functional of each of the four "Re's" CBR tasks. Then the case base is built to access each tasks in CBR applications.

1.7.4. Development Tool

Nowadays, there are different important CBR implementation tools available for teaching and academic research as stated in section 2.5. This research was implemented by using the advanced CBR system tool called jcolibri. Jcolibri framework, developed by Group of Artificial Intelligence Applications (GAIA), is selected to manage legal cases because it supports CBR tasks of retrieving, reusing, revising and retaining and its flexibility that can integrate with other tasks and methods. According to Bello and et al. (2004), jcolibri is an object-oriented framework for developing CBR applications.

Jcolibri is more flexibility because it includes facilities to work with different case representations, namely: first order logics (DL), data based records or XML files. Moreover, it provides a GUI that supports the management of CBR tasks and methods as well as the construction of the particular combination of tasks/methods that defines a CBR system.
Moreover, Microsoft excel 2007 has been used for preprocessing the cases. It is used to convert the text file into plain text file to the accessibility of jcolibri framework.

1.7.5. Testing Techniques

After the prototype system is developed, its performance was tested by phase testing during implementing cases using jcolibri, statistical analysis testing method such as recall and precision and user acceptance test method. The performance of the system was evaluated by domain experts with regard to the following parameters:

- The relevance of the attributes to represent values and case structure.
- The ease of use of the system in retrieving case with a given query.
- The applicability of the prototype CBR in the law domain.
- The system efficiency in terms of time and speed.

1.8. Organization of the Thesis

This thesis was organized into the following chapters: **Chapter one** includes the background of the study, statement of the problem (motivation), objective of the study, the scope and limitation of the study and design, the significance of the study and research methodology (literature review, data source and sample collection, case representation, development tool, and testing techniques) to conduct this study. **Chapter two** describes KBS & AI, CBR, CBR vs. RBR, CBR cycles, CBR tools, application of the CBR system in the law domain and related research work about CBR. **Chapter three** provides the Knowledge Acquisitions and representations of the collected cases to develop LCMS. **Chapter four** presents the implementation of the prototype CBR system, LCMS, development using CBR tools jcolibri framework, tests and evaluating the system performance and discussing about results. Finally, **Chapter five** gives final conclusions and forward recommendations for future studies.

CHAPTER TWO

LITERATURE REVIEW

As Ethiopia (2002) described, CBR combines the knowledge based support philosophy with simulating human reasoning when experience is used, i.e. mentally searching for similar situations happened in the past and reusing the experience gained in those situations. In CBR, the prior knowledge or cases are structured and stored in a DB called case base to be retrieved to user queries requested when trying to solve a problem. This is a usual topic of AI research in the recent years. To do so, specific knowledge in a domain area is required to develop KBS-expert system to support experts in a specific domain area.

To develop such systems, different approaches have been used such as rule based, case based, ontology (semantic) based and fuzzy based approaches. In this research case based approach is selected. Because CBR has become a very popular technique for developing knowledge-based systems that can give rational support using specific prior knowledge stored in case base to lawyers in the domain of labor law. CBR is useful for a variety of problem-solving tasks of applications including classification, planning, diagnosis, design and decision support (Kolodner, 1992; Lenz, et al, 1998).

This chapter provides a conceptual discussion of CBR in the law domain more specifically in labor law cases and gives a highlight about related researches to the study.

2.1. Knowledge Base System (KBS)

The concept of KBS is derived from the field of AI. AI is a machine learning (ML) that intends the understanding of human intelligence and building of computer programs that are capable of simulating or acting one or more of intelligent behaviors (Azeb, 2009). Intelligence is the capability of observing, learning, remembering and reasoning. According to Russell and Norvig (1995), **intelligent system** can use vast amount of knowledge to learn from experience and adapt to changing environment in interacting with human using natural language. Computer programs that try to solve problems in a human expert-like fashion by using knowledge about the application domain and problem solving techniques are known as KBS (Azeb, 2009).

As a result, AI requires extensive knowledge of the subject (domain) at hand and AI program should have knowledge base which is referred to as case base in this research. Knowledge representation is one of the most important and most active areas in AI. AI program learns and updates its knowledge.

According to Russell and Norvig (1995), some of the definitions of AI along two dimensions; human vs. ideal and thought vs. action are:

1. Systems that think like humans: **Thinking humanly (The cognitive modeling approach)**.

The interdisciplinary field of **cognitive science** brings together computer models from AI and experimental techniques from psychology to try to construct precise and testable theories of the workings of the human mind.

2. Systems that act like humans: **Acting humanly (The Turing Test approach)**.

From this perspective of definition, the computer would need to have the following capabilities:

- **natural language processing** to enable it to communicate successfully in English (or some other human language);
- **knowledge representation** to store information provided before or during the interrogation;
- **automated reasoning** to use the stored information to answer questions and to draw new conclusions; and
- **machine learning** to adapt to new circumstances and to detect and extrapolate patterns.

Turing's test deliberately avoided direct physical interaction between the interrogator and the computer, because physical simulation of a person is unnecessary for intelligence.

3. Systems that think rationally: Thinking rationally (The laws of thought approach).

The so-called **logicist** tradition within AI hopes to build on such programs to create intelligent systems.

4. Systems that act rationally: Acting rationally (The rational agent approach)

Acting rationally means acting to meet one's goals, given one's beliefs. An **agent** is just something that perceives and acts. (This may be an unusual use of the word, but you will get used to it.) In this approach, AI is viewed as the study and construction of rational agents.

Another dimension is whether we intend our intelligent computers to be conscious or not. Philosophers have had a lot to say about this issue, and although most AI researchers are happy to leave the questions to the philosophers, there has been heated debate. The claim that machines can be conscious is called the **strong AI** claim; the **weak AI** position makes no such claim.

Russell and Norvig (1995) have defined AI as:

- collection of algorithms that are computationally tractable, adequate approximations of intractably specified problems;
- the enterprise of constructing a physical symbol system that can reliably pass the Turing Test;

  "Turing Test, designed by Alan Turing to provide a satisfactory operational definition of intelligence, is an indistinguishability behavioral intelligence test mechanism between intelligent entity-human beings and machine (program or computer) for answering some written questions without direct physical interactions but a conversation via online typed messages and cannot tell whether the written responses come from a person or not. There is an interrogator who has to guess if this conversation is with a program or a person; the program passes the test if it takes-in the interrogator 30% of the time."
- the field of computer science that studies how machines can be made to act intelligently;
- a field of study that encompasses computational techniques for performing tasks that apparently require intelligence when performed by humans;
- a very general investigation of the nature of intelligence and the principles and mechanisms required for understanding or replicating it; and
- the getting of computers to do things that seem to be intelligent.

From the AI perspective, KBS are systems based on the methods and techniques of AI. Like ways, the definition of KBS is based on human centered and organizational centered perspectives. The most common definitions of KBS is human-centered. This highlights the fact that KBS have their roots in the field of AI and they are attempts to understand and imitate human knowledge in computer systems (Wiig, 1994). However, the intelligence of AI systems is quite different from human intelligence (Dreyfus and Dreyfus, 1986, Winograd, 1990) as they lack creative powers of reproduction and their learning capabilities are relatively primitive.

A second set of definition looks for characteristics in the architecture of a KBS (Lucas and Van Der Gaag, 1991). The main components of KBS are distinguished: a knowledge base, an inference engine, a knowledge engineering tool, and a specific user interface (often natural language based) and finally learning behavior (Lucas and Van Der Gaag, 1991). The knowledge base and the inference engine define the core of the KBS. The former is an active DB with some form of 'formal knowledge' about how the data may be used in practice. The inference engine defines the ways in which the knowledge base may be put to use. Typical supplements for these two components are a knowledge engineering tool, offering instruments for filling the knowledge base, and a dedicated user interface. Characteristically, a KBS user interface should allow 'why-' and 'how-' questions, having the system explain its behavior when dealing with a given problem (Dhaliwal and Benbasat, 1996). Definitions of KBS centering on architectural peculiarities are hardly more satisfactory than the human-centered ones.

A third set of definitions uses the term KBS to show all those organizational information technology (IT) applications that may prove helpful for managing the knowledge assets of the organization (Laudon and Laudon, 1997). Expert systems and groupware, data warehouses, or even intranets would be included in this definition. These definitions, however, are rejected here since they lead to erosion of the meaning of the term. KBS, in the researcher points of view, are not just any IT (information) systems used for dealing with knowledge but the expert systems to advice customers like human experts.

2.2. Case Based Reasoning (CBR)

The origins of CBR can be found in both cognitive science and AI as that it can be considered as a model for human problem solving seen in the above. CBR systems are KBS that solve problems by remembering similar past situation and reusing its solution and lesson learned from it. According to Aamodt & Plaza, (1994), CBR is an AI technique to support the capability of reasoning and learning in advanced decision support systems. Specifically, it is a reasoning paradigm that exploits the specific knowledge collected on previously encountered and solved situations, which are known as cases. CBR is a recent approach to problem solving and learning that has got a lot of attention over the last few years. It is used to solve a new problem by

remembering a previous similar situation and by reusing information and knowledge of that situation (Aamodt and Plaza, 1994).

CBR like AI has different techniques such as case representation, indexing, storage, retrieval and adaptation are the commonly used techniques in any CBR research paradigm (Aamodt & Plaza, 1994).

CBR is a problem solving paradigm that in many respects is fundamentally different from other major AI approaches (Aamodt and Plaza, 1994).

- 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.
- CBR is also an approach to incremental, sustained learning, since a new experience is retained each time a problem has been solved, making it immediately available for future problems.

In problem-solving with experimental method, the four "Re" of CBR cycle (see section 2.5): Retrieve, Reuse, Revise, and Retain should be considered for cyclic learning (Aamodt and Plaza,1994) as CBR solutions are proposed by retrieving previous experience and adapting it to solve new problems.

According to Aamodt and Plaza (1994), the CBR paradigm covers a range of different tasks such as: representing, retrieving, utilizing and revising the knowledge retained in past cases. Cases may be stored in a case base and distributed within the knowledge structure using the selected tool. Cases may be indexed by a prefixed or open vocabulary, and within a flat or hierarchical index structure. The solution from a previous case may be directly applied to the present problem, or modified according to differences between the two cases. The matching of cases, adaptation of solutions, and learning from an experience can be guided and supported by a model of general domain knowledge, or be based on an apparent similarity.

2.3. Case Based Reasoning (CBR) vs. Rule Based Reasoning (RBR)

As seen earlier, KBS have been developed by using different approaches. As Prentzas and Hatzilygeroudis (2007) stated, RBR and CBR are two popular approaches used for problem solving in intelligent systems and they are natural alternatives in knowledge representation.

Rules usually represent general knowledge, while cases encompass knowledge accumulated from specific (specialized) situations.

As Sasikumar, and et al. (2007) stated, symbolic rules constitute a popular knowledge representation scheme used in the development of expert systems. Rules represent general knowledge of the domain. RBR is therefore a particular type of reasoning which represents knowledge in the basic form of "**if-then-else**" rule statements with the forward and backward chains (Sasikumar et al., 2007). Rules are simply patterns and an inference engine searches for patterns in the rules that match patterns in the data. The "if" means "when the condition is true," "then" means "take action A" and "else" means "when the condition is not true take action B."

However, symbolic rules or RBR have their own drawbacks such as: lack of method (i.e., systematic procedure), the rules do not interact among themselves, lack of structure, and representing procedural tasks (Sasikumar et al., 2007).

CBR offers some advantages compared to symbolic rules and other knowledge representation formalisms (Prentzas and Hatzilygeroudis, 2007; Pal and Shiu, 2004). Cases represent specific knowledge of the domain. Cases are natural and usually easy to get. New cases can be inserted into a knowledge base without making changes to the pre-existing knowledge. Incremental learning comes natural to CBR. The more cases are available the better the domain knowledge will be represented.

In general, as compared with Rule-Based, sometime without complexity of rules systems, new cases can be added easily. So, CBR Systems are easier to maintain as compared to Rule-Based.

CBR has many advantages as compare to other sub fields of AI such as RBR.

As stated by Pal and Shiu (2004) some of the advantages of CBR include the following.

- Reducing the knowledge acquisition task

The knowledge acquisition task of CBR consists of collection of relative past cases, and their representation and storage. In the rule based systems, knowledge acquisition is necessary by extracting a set of rules.

- Avoiding repeating mistakes made in the past

In CBR system failure and success record, as well as reason to failure record kept. This information about cause of the failure is used in the future for reducing the failure outputs.

- Providing flexibility in knowledge modeling

CBR systems used past experience as domain knowledge and can provide reasonable solution, appropriate solution through adaptation. But, modeled base system cannot solve problem due to fixed modeling and formulation that is on the boundary of their knowledge.

- Reasoning in domains that have not been fully understood, defined or modeled

CBR system can still be developed by only adding small set of cases from the domain, in situation where too insufficient knowledge exists to build a model. Addition of new cases is caused of expanded knowledge of CBR System. These are used in this direction that is determined by the cases encountered in its problem solving.

- Making prediction of the probable success of proffered solution

The past solution information is stored in a case base with a level of success. Case based reasoner may be able to predict success of solution for current problem. The level of success of these solution and differentiate between pervious and current cause of applying these solution, the procedure is done by referring stored solution.

- Learning over time

CBR systems are normally worked in learning situation as they express more problems and get solutions. The level of success tested and determined in the real world, these solutions can be added to case base to solve the future problems. By adding procedure of more cases the CBR systems become more and more efficient.

- Reasoning with incomplete or imprecise data of concepts

The retrieved cases not may be very similar solution to the current case. When they are within some defined potential of similarity to the present case any drought and incompleteness can be deal by Case-Based reasoned or domain expert.

- Avoiding repeating all the steps that need to be taken to arrive at a solution

Reusing a pervious solution also allow the actual steps to taken to reach that solution to be reused for solving other problems. Because in problem domain that requires a well defined process to create solution of new problem, but alternative approach of modifying an earlier solution can reduce this process.

- Providing a means of explanation

CBR systems can explain solution to user by explaining how pervious case was successful in that situation, by using similarities between the cases. It is possible because, CBR system can provide information about pervious case and its successful solution to help a user.

- Extending to many different purpose

The CBR system can be implemented in many different ways. It can be used in many forms like decision support systems, creating plan, arguing a point of view and making a diagnosis.

- Extending to a broad range of domains

CBR can be applied on various numbers of application domains. This is due to the appearing of unlimited number of ways representing, indexing, and managing cases.

- Reflecting human reasoning

Human can understand a CBR system reasoning and explanation and are able to convince of the validity of the solution they receive from the systems.

2.4. CBR Cycles

Based on the CBR cycle, the whole LCMS decision-making functionality can be described in chain functionality. A case is a previously experienced problem situation. When it has been captured and learned in such way that it can be reused in the solving of future problems, it is referred to as a past case. Correspondingly, a new case or unsolved case is the description of a new problem to be solved. CBR is in effect a cyclic and integrated process of solving a problem, learning from this experience, solving a new problem, etc (Aamodt & Plaza, 1994). Therefore, as depicted in Fig. 2.1 the CBR cyclic processes are stated by Aamodt & Plaza in the four "Re's" applications of CBR. These are:

- retrieve the most similar case(s), with respect to the current input situation, contained in the case repository, which is known as the case base;
- reuse them, or more precisely their solutions, in order to solve the new problem;
- revise the proposed new solution (if it is considered necessary); and
- retain the current case for possible future problem solving.

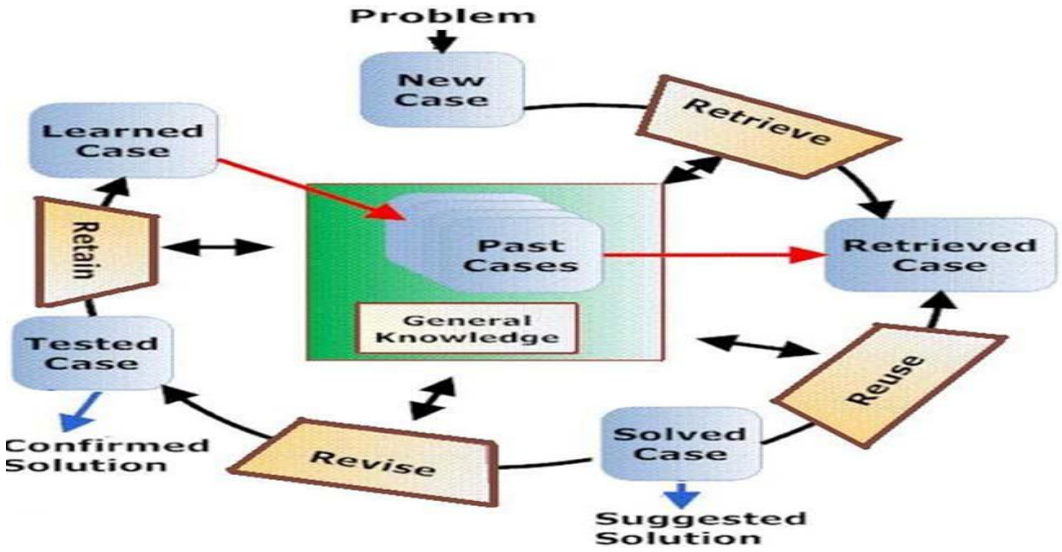


Fig. 2.1: The Case-based Reasoning Cycle (adopted from Aamodt and Plaza 1994).

As depicted in Fig. 2.1., each CBR process involves several more specific steps (see Fig. 2.3). The top level task is problem solving and learning from experience. Other four major CBR tasks: retrieve, reuse, revise and retain are partitioned to sub-tasks. It is important to describe each of the 4 "Re's" of the CBR cycles as follows.

2.4.1. Retrieval in CBR

An important step in managing the CBR cycle (Fig. 2.1) is the retrieval of previous cases that can be used to solve the target problem. In case retrieving, similarity based retrieve is the major considerable issue. A basic assumption in similarity-based retrieval is that a given case is more acceptable than another if it is more similar to the user's query. The solution is retrieved from the CBR on the similarity between a new case and cases already stored in case base.

Therefore**, retrieval** refers to the process of returning one or more cases from the case base through the comparison of a new case (the input problem) with the ones in the case base (candidate cases). The result of this comparison is the selection of one case (or a combination of) that suggests the solution to the input case (Weber, 1998).

Many CBR applications rely on domain knowledge encoded in the similarity measures used by the system to guide the retrieval of relevant cases. In some applications of CBR, it may be adequate to assess the similarity of the stored cases in terms of their surface features. The surface

features of a case are those that are provided as part of its description and are typically represented using attribute-value pairs. In other applications, it may be necessary to use derived features obtained from a case's description by inference based on domain knowledge.
In yet other applications, cases are represented by complex structures (such as graphs or first order terms) and retrieval requires an assessment of their structural similarity. As might be expected, the computation of derived features or use of structural similarity is computationally expensive. However, the advantage is that more relevant cases may be retrieved.

A CBR system can guarantee that it retrieves the k cases that are maximally similar to the target problem by computing the similarity of the target problem to every case in memory.

2.4.2. Reuse in CBR

After the retrieve phase, it is possible to establish a solution for a new case which is called then a solved case (see Fig. 2.1).

Reuse refers to the reutilization of the content of the adopted retrieved case to solve the new case (Weber, 1998). The reuse process in the CBR cycle is responsible for proposing a solution for a new problem from the solutions in the retrieved cases.

Reusing a retrieved case can be as easy as returning the retrieved solution, unchanged, as the proposed solution for the new problem. This is often appropriate for classification tasks, where each solution (or class) is likely to be represented frequently in the case base, and therefore the most similar retrieved case, if sufficiently similar, is likely to contain an appropriate solution. In particular, adaptation knowledge is of major importance: it is used during the retrieval step to retrieve a good source case, to evaluate the cases and build the solution to the current problems or cases faced to the employee (s) accordingly.

In reuse stage, a complete design where case-based and slot-based adaptation can be hooked is provided. At reuse stage several methods for adaptation are available (like direct proportion).

2.4.3. Revision in CBR

In the **revise** step, the solution proposed is evaluated. But reuse becomes more difficult if there are significant differences between the new problem and the retrieved case's problem. In these

circumstances the retrieved solution may need to be adapted to account for these important differences. Law decision making is one domain in which adaptation is commonly required.

Revise is a task of CBR application that evaluates the solutions adapted by the reuse method and accordingly repair is followed. At revise stage methods for revision of cases are realized, as well methods for new indices generation and methods for decision making.

2.4.4. Retention in CBR

Retain or adaptation refers to the addition of the new experience or of the initial ones to the case memory that works as a learning device (Weber, 1998). During retain, useful experience is retained for future reuse, and the case base is updated by a new learned case, or by modification of some existing cases. Adaptation in CBR is normally improved by the application of domain specific rules to the retrieved cases. Thus the retrieved solution is simply an initial solution and any differences between the new problem and the retrieved case's problem are likely to alter the retrieved solution.

In the classic review paper by Aamodt & Plaza (1994), retention is presented as the last step in the CBR cycle, in which the product of the most recent problem-solving episode is incorporated into the system's knowledge. To a great extent this has traditionally translated into a variety of approaches for recording the product of problem solving as a new case that can be added to the case base. Of course, there are various issues concerning how best to learn a new case and different systems record different types of information in their cases.

Many systems store only the solution, but others record a much deeper representation of the problem solving process that brought about the particular solution. These rich knowledge structures describe precisely how a given solution was derived, providing a trace of the decision-making processes that led to a particular solution.

In general, the modern view of retention accommodates a broader perspective of what it means for a CBR system to learn from its problem solving experience, a view that is largely a response to certain critical issues that have arisen during the practical application of CBR systems in complex problem solving scenarios. This task can be embraced with maintenance in other more task applications: review and restore of CBR as seen in Fig. 2.2. The process of updating the

case base is totally based on implementation of the case-base. Retain is used for storing the solved new cases for future use. At retain stage there are methods for query retaining to the case base for future use.

2.4.5. Case-Based Maintenance

Maintenance issues arise when designing and building CBR systems and support tools that monitor system state and effectiveness to decide whether, when, and how to update CBR system knowledge to better serve specific performance goals. Understanding the issues that underlay the maintenance problem and using that understanding to develop good practical maintenance strategies is crucial to sustaining and improving the efficiency and solution quality of CBR systems as their case base grow and as their tasks or environments change over long term use (Roth-Berghofer, 2003; Iglezakis et al., 2004). And today there is a general recognition of the value of maintenance to the success of practical CBR systems.

The available maintenance operations may target different knowledge containers (e.g., indices, the cases themselves, or adaptation knowledge) and may be applied at different times or to varying portions of the case base. Of course, the success of maintenance depends not only on the maintenance knowledge containers or policies themselves, but also on how maintenance is integrated with the overall CBR process.

Reinartz et al. (2001) propose to extend the classic 4-stage CBR cycle shown in Fig. 2.1 to include two new steps, a review step to monitor the quality of system knowledge, and a restore step which selects and applies maintenance operations. Their revised model, shown in Fig. 2.2, emphasizes the important role of maintenance in modern CBR and indeed proposes that the concept of maintenance encompass retain, review and restore steps (Iglezakis et al., 2004). As shown in Fig. 2.2, the steps **Review** and **Restore** are not part of the original 4 “Re’s”cycle. These steps support the monitoring and maintenance of single CBR systems (Roth-Berghofer, 2003).

The addition and/or deletion of cases in a CBR system is just one aspect of maintenance, and maintenance policies can be applied to a variety of other knowledge sources beyond the case base.

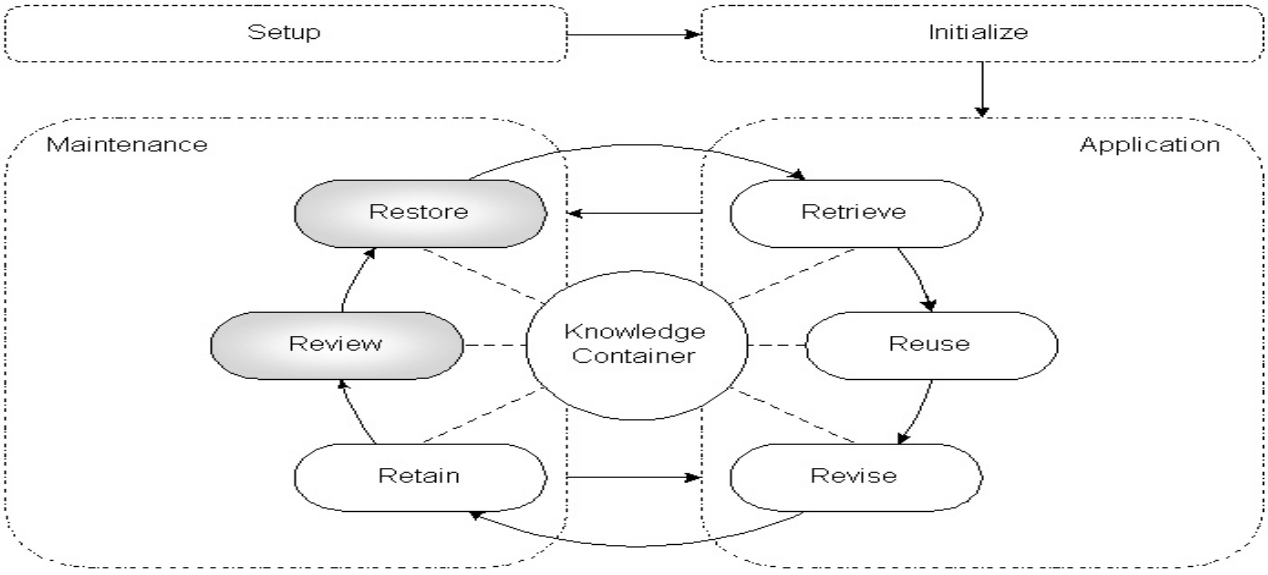


Fig. 2.2: An extension of the 4 Res, of the CBR methodology for maintaining CBR systems. (Adopted from Iglezakis et al., 2004).

There are methods to solve tasks either by decomposing a task into other subtasks or by solving it directly. Each of the four processes involves a number of more specific steps. For example, retrieve involves identify features of collecting descriptors (attributes) of interpreting problem, search index structure and case base, initially match of similarity and select for elaborating explanations; reuse involves copy of solution and solution methods and adopt to modify solution and solution methods; revise includes evaluate the solutions in the real world and in the mode and repair faults by user and retain includes integrate to return problem and update the case base, index to determine and generalize index structure and extract solution method, justifications and relevant attributes (Aamodt & Plaza, 1994).

However, they all assume that the case base is ready for the first process, case retrieval, although they discuss the representation of cases and believe that a case-based reasoner is heavily dependent on the structure and content of its collection of cases. Therefore, a new problem is solved by retrieving one or more previously experienced cases, reusing the case in one way or another, revising the solution based on reusing a previous case, and retaining the new experience by incorporating it into the existing case-base.

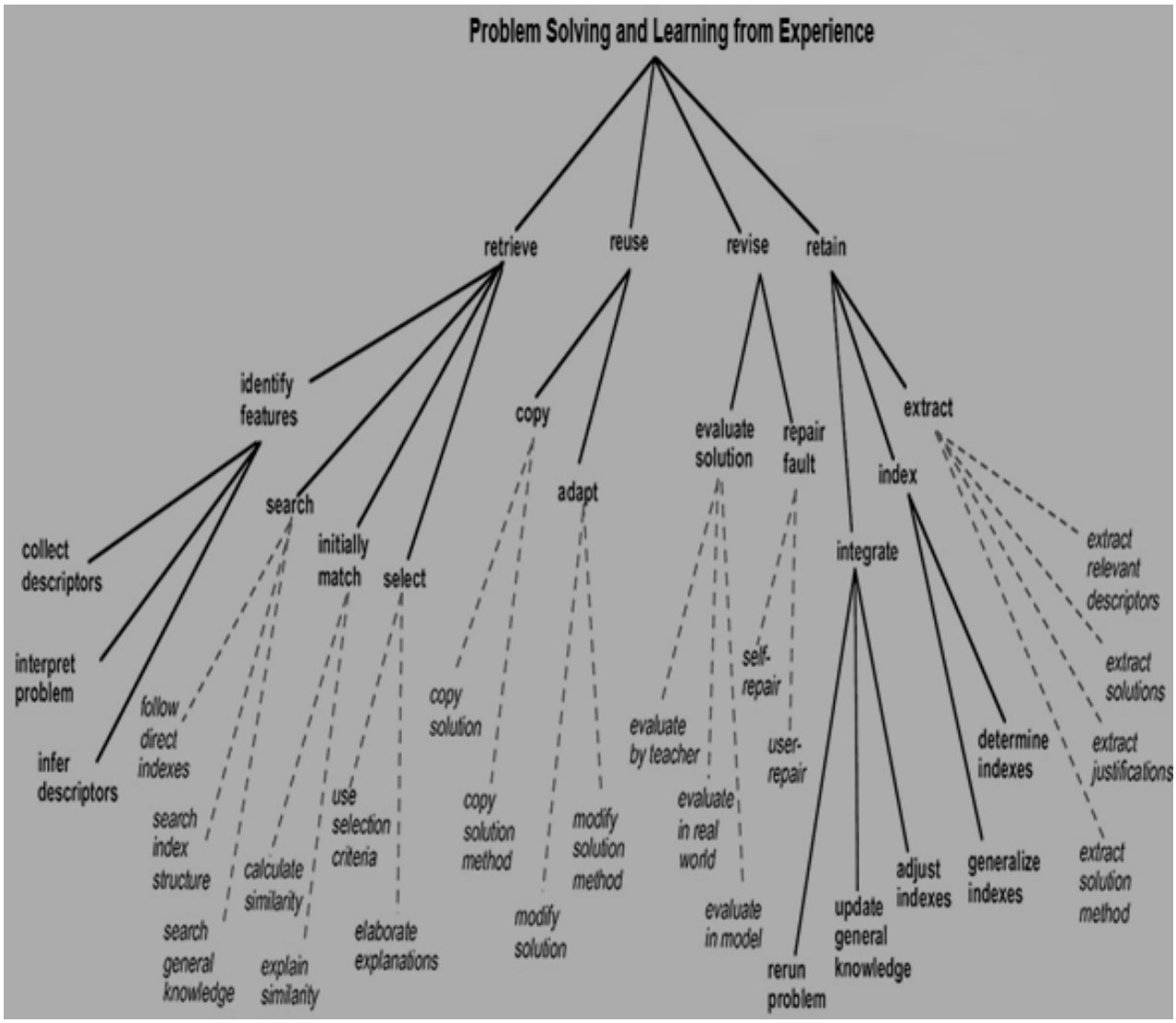


Fig. 2.3: A task-method decomposition of CBR application (adopted from Aamodt and Plaza, 1994)

2.5. Case-Based Reasoning (CBR) Developmental Tools

CBR tool is software that can be used to develop several applications that need CBR method. CBR shells are kind of application generators with graphical user interface. There are a number of developmental tools that can be used for developing a CBR system. Some of these are accessible freely as open source or non-commercially. Some of these are stated as flow.

myCBR: myCBR is one of the most popular CBR software platforms and developed by the German Research Center for Artificial Intelligence (Stahl & Roth-Berghofe, 2008). It can be easily modified by the users depending on the purpose. The purpose of myCBR is to minimize the efforts to create CBR applications. The framework myCBR supports description of cases with various attributes: numeric, character, string, logical and class type. The templates of the cases are generated as classes or subclasses with a number of attributes, called slots.
In myCBR, the case and their attributes created manually or automatically. The automatic generation of attributes (slots) is done during the import procedure of the Comma Separated Value (CSV) file. The tool supports only Retrieve and Retain among four "Res".

ReMind: It is produced by Cognitive Systems Inc. It is basically developed for Macintosh, but after some time, it is also developed for Windows and UNIX. ReMind offer template, nearest neighbor, inductive and knowledge-based retrieval. Its limitation is retrieving speed. Nearest neighbor is very slow, on the other hand inductive retrieval is very fast. When it creates inductive index, then it becomes slow (Watson & Marir, 1994). It will able to access data in ODBC-compliant databases and very influential tool.

ART*Enterprise: It is the product of Inference Corporations (Watson & Marir, 1994). In 1980, ART was advertised as an AI-based tool. Inference dropped the label of AI and now ART*Enterprise marker as an integrated and object-oriented development tool. ART* Enterprise has many features (Watson & Marir, 1994). It offers cross-platform support for all operating system. It provide excellent environment to integrate CBR with others application. Since the whole package is very powerful, ART*Enterprise is ideal tool for embedded CBR functionality within a corporate wide information system. Its package include graphic user interface (GUI) builder.

Induce (it is later renamed to **CasePower):** was developed by Inductive Solutions Inc. and it builds its cases in matrix environment provided by Microsoft Excel. Rows and columns of spread sheet are used to define cases and their attributes. It uses nearest neighbor retrieval and reduces the search time by calculating the index in advance. If new case is retained, then entire set of case indices must be recalculated (Watson & Marir, 1994).

CBR-Express: This CBR tool primarily designed for help desk applications by Inference Corporation. It provides a comfortable user interface and fast retrieval speed. It has simple case structure and nearest neighbor retrieval algorithm of cases. The key features of CBR-Express are to handle free-form text and enable customers to describe their problem in own words. The use of trigrams means that CBR-Express is reluctant of spelling mistakes and typing errors such as letter transpositions. It stores cases in relational database. CBR-Express is a network ready and cases can be shared between organizations.

KATE (it is later renamed to **CaseWork):** This tool is developed by AcknoSoft (Watson & Marir, 1994) that can run on MS Windows, Mac, or SUN. Kate is made up of Kate-induction, Kate-CBR, Kate-Editor and Kate-Runtime, this tool support both kind of Nearest Neighbor and Inductive retrieval algorithm. Kate-Induction is an ID3-based induction system that supports object-oriented representation of cases. Cases can be imported from many databases and spread sheets. Induction algorithm can make use of background knowledge. In induction algorithm, retrieval using trees is extremely fast. KATE-CBR uses nearest-neighbor approach. It supports same case objects hierarchies as Kate-Induction. Two techniques can be combined in a single application. Users can customize similarity assessments.

CaseAdvisor: It is marketed by Sentenia Software at Frazier University in Canada (Watson & Marir, 1994). It is also developed by Inference‟s CBR product. This software has three parts (Watson & Marir, 1994). These are CaseAdvisor Authoring, CaseAdvisor Resolution and CaseAdvisor WebServer.

Eclipse – The Easy Reasoner: It is a product of Haley Enterprises. Eclipse is implemented in C by NASA. In late 1980, the former chief scientist of Inference developed a new language like Eclipse. Eclipse is available for Dos, Windows and UNIX platforms. It supports nearest neighbor and inductive retrieval (Watson & Marir, 1994).

Esteem: It is from Esteem Software (Watson & Marir, 1994). It is developed in Intellicorp's Kappa-PC. Esteem uses kappa's inference engine for developers to create adaptation rules. It supports application which has multiple case-bases and nested cases. This means that one can reference another case-base through an attribute slot in a case.

CBR-Works: it is a shell for Case-Based application building. Besides the retrieval of cases, it supports modeling the cases' structure and maintaining the case base. Its consultation mechanism also covers the whole CBR-Cycle from retrieving to revising (Schulz, 1999). CBR-Works comes from the German company TECINNO, running on MS Windows, Mac, OS/2, and various UNIX platforms. Written in SMALLTALK, it supports an object-oriented model and flexible retrieval methods. It also supports the definition of concept and type hierarchies to help define similarity of symbolic concepts. CBR-Works includes an attribute editor, a rule editor, similarity criteria editor, distributed processing support and is easily integrated to existent applications. CBR-Works can import case-bases from Microsoft Excel and in the CASUEL case format.

According to Schulz (1999), a case in CBR-Works has four possible states: unconfirmed, confirmed, protected, and obsolete. Usually, new cases become unconfirmed being unrevised or uncompleted cases not valid for retrieval. Revised cases become either confirmed which allows for retrieval or protected which additionally protects the case from changes. Old cases, no longer valid for retrieval but probably useful for further statistics, becomes obsolete. The tool was used by Ethiopia Tadesse to develop the Amharic legal precedent retrieval system.

jcolibri: The jcolibri is a system for building CBR applications that is an evolution of previous work on knowledge intensive CBR. jcolibri is a technological evolution of COLIBRI and it is an object-oriented framework in Java which is designed for building CBR systems. It is a java-based and uses JavaBeans technology for case representation and automatically generation of user interface. This framework is developed by the Group of Artificial Intelligence Applications (GAIA) in Complutense University in Madrid.

The framework is built in two hierarchical levels upper and lowers (see Fig. 2.4). The lower level consists of library of classes (Software modules) for full four "Re's" CBR cycle, also for definition of cases, attributes and connectors for access to outer databases. The upper level is "black box" – graphical interface, which allows non-complicated user CBR application generation based on lower levels modules. The jcolibri tool has been used for this research so it is better to describe it as follow in more detail.

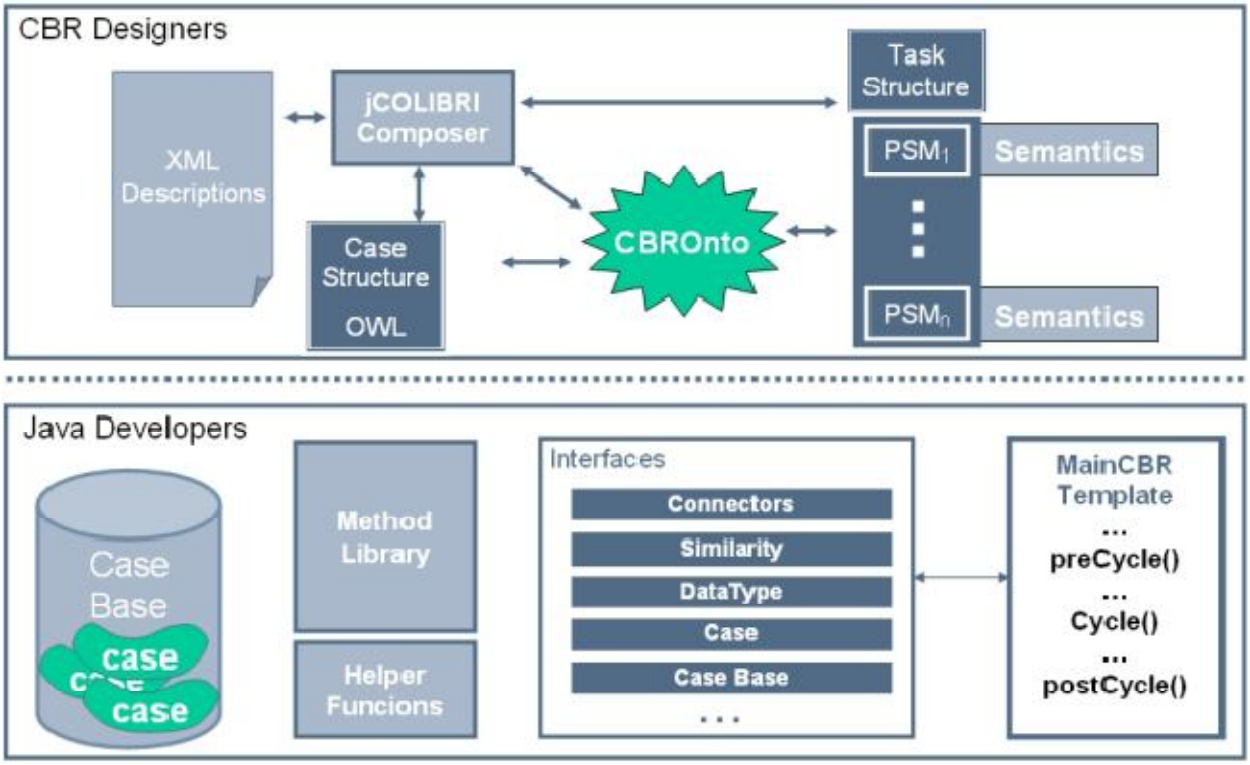


Fig. 2.4: Jcolibri framework architecture

2.5.1. Jcolibri in CBR Applications

Jcolibri is used for developing a new CBR application and instantiation framework classes directly. Learning of framework and how to use it for creating a new CBR application are hard processes.

The application framework of jcolibri (Bello-Tomas et al., 2004) comprises a hierarchy of JAVA classes plus a number of XML files. Jcolibri allows retrieval from clustered (organized) and indexed case bases and submits program interfaces (connectors) to access text and XML files, as well standard and description logics (DL) databases. These interfaces can be used for retrieval systems DB access. There are lots of CBR applications, developed on jcolibri based: additional shells (abstract levels) for distributed CBR systems, statistical CBR systems, multi-agent supervisor systems for text file classification, and lots of CBR recommender systems. The framework is organized around four main elements: tasks and methods, case base, cases, and problem solving methods (PSM).

Tasks and Methods: XML files explain tasks supported by the framework and the methods to solve these tasks. Tasks are the key elements that represent the method goal and can identify it by name and description in an XML file. Users can add task to the framework at any time when necessary.

Case Base: jcolibri has a memory organization interface that assumes that whole case-base can be read into memory for the CBR to work with it. Jcolibri implemented a new interface who allows retrieving cases enough to satisfy a structured query language (SQL) query. A second layer of case base is a data structure which will organize cases after they loads into memory. The two layer approach is efficient enough to allow different strategies for retrieving cases.

Cases: jcolibri represent cases in a very simple way. A case is individual which has number of relationships with other individuals attributes. Framework is supported by different data types which define any simple case.

Problem Solving Methods: jcolibri deals with the CBR methodology to support full CBR cycle. This developmental tool is used because of its flexibility and it supports to the four “Re’s”CBR tasks: **retrieve** the most similar case(s), **reuse** its/their knowledge to solve the problem, **revise** the proposed solution and **retain** the experience.

The core of jcolibri is called CBRCore, and taken from jcolibri (see Fig. 4.5). It is the most important component of the framework. The core is in charge of the application, and must always be present for an application to run. It handles the configuration, and executes the application. To do all this, the core is divided into three main components which will be described in turn: state, context and packages.

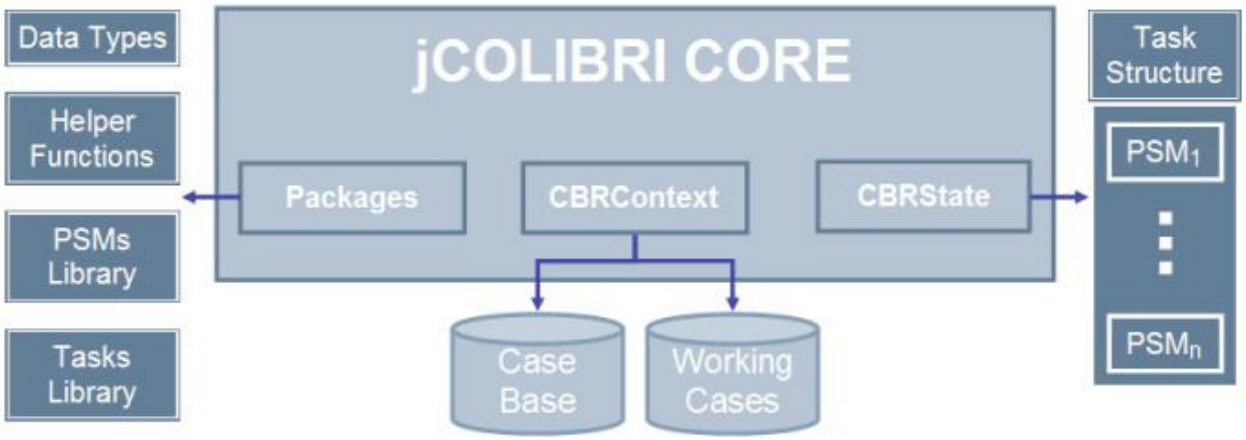


Fig. 2.5: The jcolibri Core

The state, which is called CBRState, handles the configuration of tasks and methods. It will always have the current configuration status of the CBR application. The context, called CBRContext, acts as a communication blackboard where methods can share data during the execution of an application. The context will have the case base and the working cases. The last application configuration is sure to satisfy each component's conditions because of the context checker. A PSM, which relies on knowledge to accomplish a task, may be given preconditions and post conditions before and after its execution.

The remaining components (such as data types, similarity functions, case structures) of the system are located in packages. Each package may contain a set of one or more components. Jcolibri comes with a few rather stable packages at this time: core, textual, DL and web. The core package should always be enabled for an application, and the GUI does this by default.

Jcolibri can be started and visible by clicking on exe file **jCOLIBRIGUI.bat** and then the GUI of jcolibri could be appeared. GUI of the jcolibri helps to create new CBR application with predefined tasks and methods. These predefined tasks and methods are in the files of task.xml and methods.xml.

2.5.2. Theoretical Comparison of jcolibri with other related Tools

According to Iqbal and Ashraf (2006), the theoretical comparison of jcolibri with other CBR-based tools is highlighted below.

- With CBR-Express

CBR-Express support simple case structure but jcolibri uses simple to complex case structure. Cases are stored in relation DB by CBR-Express and on the other hand, jcolibri stores cases in text and database. Both of them have the facility to share data between networks. CBR-Express handles free-form text which is not supported by jcolibri.

- With ReMind

ReMind and jcolibri are very flexible which supports many case retrieval methods. Jcolibri is domain independent. This feature is not supported in ReMind. It represents cases as an attribute but jcolibri supports many different ways to represent cases. ReMind dominating feature is to access data in ODBC compliant databases.

- With ReCall

It is coded in c++ and Jcolibri is built in java. Both of them are open architecture which allows the user to add functionality of its own will. Recall supports two type of adaptation mechanism but Jcolibri supports many type of mechanism. These two can easily integrate with other applications. jcolibri does not have feature to diagnose and do maintenance. This feature is supported by ReCall. Only information retrieval is supported by ReCall. jcolibri supports many different techniques of retrieval.

- With Eclipse

jcolibri is very suitable for non-experienced programmers as Eclipse. There are built-in methods. You have to just click on method and method is ready to perform your task. Eclipse is suitable for only experience programmers. Usual range of variable types is supported by these tools. jcolibri use index to identify the case and stems are uses by Eclipse for this purpose. Most grooming quality of Eclipse is optimizing forward chaining using the Rete Algorithm.

- With Esteem

It has feature to support multiple case-bases and nested cases like jcolibri. It also gives weight to features. These two tools are suitable for non-programmers. User can integrate his own similarity. In jcolibri, user can add similarity, variable type, task, method and way to doing this

task. One good feature of Esteem is it can be used as a server. It is very slow tool. jcolibri is very fast tool. Bad feature of Esteem is inflexible interface which make this tool inappropriate for serious application.

- With ART*Enterprise

ART*Enterprise is platform independent like jcolibri. It has more controlled CBR functionality as compared to jcolibri. These two tools are excellent environment to integrate with others application and also support complex case adaptation. The feature which differentiate ART*Enterprise with other tools is sophisticated version control facilities.

- With CasePower

It represents cases in matrix environment provided by Microsoft Excel as compared to jcolibri which represent cases in plain text and database. It is not efficient as jcolibri. It provides basic functionality of CBR for numeric applications only. It is not flexible like jcolibri which adopts new methods easily. Case adaptation in CasePower is performed by using Excel formulas and macros.

- With CBR-Works

Its consultation mechanism covers the whole CBR-Cycle from retrieving to revising but limited to retain task. However, jcolibri implements the 4 “Re’s” application of CBR tasks.
Though CBR-Works is designed as a complete environment, it may also act as a CBR-Server for several clients by the use of CQL (Case Query Language). Last but not least, CBR-Works offers an open interface to build a Case-Based application from existing data stored in common DB systems.

In CBR-Works, concepts define the structure of the cases while in jcolibri manage case structure indexes the structure of the cases. They are defined in hierarchy similar to a class model hierarchy including inheritance. Each concept consists of attributes which can be either atomic (defined by a type) or complex (has-part relation-ship to another concept).

A case in CBR-Works has four possible states: unconfirmed, confirmed, protected, and obsolete. Usually, new cases become unconfirmed being unrevised or uncompleted cases not valid for retrieval. Revised cases become either confirmed which allows for retrieval or protected which additionally protects the case from changes. Old cases are no longer valid for retrieval but

probably useful for further statistics, become obsolete. But in jcolibri new case is possibly to store after evaluating by the user and the old cases become used for retrieval.

2.5.3. Advantages of jcolibri

Jcolibri is good tool to use and represent cases in very simple way. It has graphical user interface to help users. It is easy to use its core methods and tasks. In knowledge level jcolibri has three advantages: it supports frame-work instance, explains possible extensions and guides framework design.

According to Iqbal and Ashraf (2006), there are also a number of advantages of jcolibri. Some of these are: jcolibri is an object-oriented framework built in java that supports software reuses; all the configured data are stored in XML form, which is interchangeable language between computers to communicate one another; its frame-work is very powerful which supports different types of CBR systems from simple nearest neighbor to more complex knowledge approaches; It's very easy to develop CBR systems when compared to other CBR developmental tools; it facilitates CBR system design by providing graphical tool; it manages big case bases in a very efficient way; the key aspect of jcolibri is to use available ontologies as a cases index, hence it can retrieve efficiently; jcolibri has library of PSMs to solve the CBR tasks; it provides graphical user interface that allow the user to choose methods by his/her own desired tasks; it is an extendable framework in which new elements or cases can easily integrate or add with old ones; it facilitates developers or designers to work with different case representations; it is very suitable for indexing and retrieval over the web; own data type can be created in jcolibri by using mange data type option; and CBR applications, which are developed by jcolibri, are working as platform independent because jcolibri is totally based on JAVA object oriented framework.

Based on these and such advantages of jcolibri framework, it is selected and used to develop this prototype CBR system for legal case management for labor law cases.

2.6. Application of CBR in Law

Today's CBR-based applications are used very commonly in many fields (such as in law, medicine (health), planning). Application areas of CBR include help-desk and customer service, recommender systems in electronic commerce, knowledge and experience management, medical

applications and applications in image processing, applications in law, technical diagnosis, design, planning, as well as applications in the computer games and music domain.

AI and law in the modern era have worked hand in hand to manipulate case. AI literature describes many efforts toward modeling legal reasoning. However, the application of AI in the domain of Law might be faced from a broader perspective if the final goal is to develop intelligent systems that support legal activities successfully (Weber, 1998). Intelligent applications in the domain of Law started with expert Systems (ES) which embody intelligent KBS that can perform an expert task (Durkin, 1994); they can be classified into rule-based systems, frame-based systems, and inductive systems.

CBR systems have demonstrated that CBR is an appropriate technique to deal with the idiosyncrasies and peculiarities of designing KBS in the legal domain. Applications of CBR in the legal domain enlarged the horizon of tasks with the use of CBR technology. The peculiarities of the knowledge in the legal domain are delaying practical and successful results (Weber, 1998). Some of the applications of CBR systems in the law are listed in Table 2.1.
Ashley (1987, 1991) stated HYPO as a CBR system that evaluates problems by comparing and contrasting them with cases from its Case Knowledge Base. **HYPO** is a computer program that models reasoning with cases and hypotheticals in the legal domain. It generates legal arguments citing the past cases as justifications for legal conclusions about who should win in problem disputes involving trade secret law. HYPO's arguments present competing adversarial views of the problem and it poses hypotheticals to alter the balance of the evaluation. HYPO uses dimensions as a generalization scheme for accessing and evaluating cases.

Name	Highlights with references
JUDGE	Adapts past similar cases to sentences juvenile criminals and retrieves another similar episode to check the consistency of the sentence. (Bain, 1984, 1986, 1989)
MEDIATOR	Uses CBR technology implementing a planning task where parts involved in a dispute have goals and sub-goals to achieve. (Simpson 1985, Kolodner & Simpson, 1988)
PERSUADER	Mediates disputes in the domain of labour negotiations, adapting plans considering goals and constraints to create arguments. (Sycara, 1987a)
HYPO	Creates legal arguments from a case base on the domain of trade secret law. (Ashley & Rissland, 1988a, 1988b, Ashley, 1990)
GREBE	A hybrid system integrating cases with rules in legal explanations; uses explanations to support classification to perform legal reasoning. (Branting, 1991c)
BANKXX	Generates arguments by performing a case-based heuristic search of a highly interconnected network of legal knowledge in the area of personal bankruptcy. (Rissland et al., 1993)
CABARET	Integrates reasoning with rules and reasoning with previous cases to perform interpretation in the domain of income tax law. (Rissland & Skalak, 1991)
PRUDENTIA	Researches for past legal decisions that can be useful in teaching lessons to a new situation performing intelligent jurisprudence research. (Weber, 1997; Weber et al., 1997a; Weber et al., 1997b; Weber et al.,1998)

Table 2.1: CBR applications in the legal domain (adopted from Weber, 1998).

As Ashley (2003) described, there are reasons to believe that at least some European legal systems are converging in their use of precedence: Given the "Europeanization of Europe" courts are beginning "to rely upon decisions not only of the European Court of Justice, but also of other Member State courts." In short, judicial decisions are becoming more amenable to distinguishing and to … use of the fact-based result of the decision in addition to the announced rationale and the discernible principles" (Lundmark, 1998). Another important reason is that, "the proliferation

of computers puts past decisions at the fingertips of judges and lawyers." In addition, it is predicted that, as the "density of regulation" increases and as norms change more rapidly, "the same set of facts raises more legal issues than before. Consulting previous decisions (precedents) helps to chart one's way through the legal thicket of, for example, the burgeoning European private law." Also, a "self-imposed adherence to precedent" will help judges "to reduce political disapproval, and to forestall legislative measures to restrict their ability to stray from precedent." (Lundmark, 1998).

2.7. Related Works

Globally, a number of scholars have investigated CBR systems in the domain of law. As Ashley (2003) stated, researchers (scholars) in AI and Law have developed a number of computational models of case-based legal reasoning. Some of these scholars are cited in Table 2.1 that have developed CBR systems in the domain area of law. Ashley (2003) has cited on his article, "Case-Based Models of Legal Reasoning in a Civil Law Context" , different CBR systems has been developed by different scholars such as: Hypo (Ashley, 1987; 1990, 1991), CABARET (Rissland & Skalak, 1991), GREBE (Branting, 1991; 1999), CATO (Aleven, 1997; 2003), BankXX (Rissland, Skalak, et al. 1996), and Split-Up (Zeleznikow, Stranieri, et al., 1995-1996).

According to Ashley (1987, 1991), the approach taken in HYPO differs from that of other AI research in analogical reasoning in a number of respects. HYPO:

- Performs a **highly** constrained kind of matching.
- Does not use a static hierarchy among features.
- Does not assume a strong model of the domain.
- Combines information from the most analogous cases.
- Symbolically compares the most analogous cases and counter-examples.

As Ashley and Rissland (2001) stated, CABARET's three-tiered model of statutory interpretation focused on relevant similarities and differences by taking into account the point-of-view of the arguer, the apparent status of the rule on the problem case, and its status in past cases. The top tier contains four argument stances: (1) broaden a rule that on its face does not hold but that one wants to hold in the problem case; (2) discredit a rule that does seem to apply

and that one doesn't want to; (3) confirm that a rule establishes a desired conclusion, (4) confirm that a rule fails to establish an undesired conclusion.

Ashley (2002) described CATO as a tutoring system which has several mechanisms to help teach law students the skills of making and responding to arguments for deciding problems, formulating analogy warranting rules and supplying rationales for them. Specifically, CATO employs and presents explicit templates for argument moves like Analogizing/Conflict-Resolution, which specifies the form of the analogy-warranting rule, and Distinguishing, which shows how to attack the analogy-warranting rule. It also presents and implements algorithms and examples inviting students to generate hypotheses like the analogy-warranting rules and to test them against the case base. CATO controls the dialogue, the judge.

CATO analyses cases in terms of factors, where a factor is a prototypical fact situation that predisposes the decision in favor of one party or the other in the case; for trade secret law, the domain CATO is designed for, the factors concern trade secret misappropriation.

The models have originated in common law jurisdictions, often by lawyers/computer scientists influenced by common law legal practice and traditions. More recently, some of the case-based models have been developed by researchers in countries with more or less civil law traditions (Prakken & Sartor, 1997; Bench-Capon & Sartor, 2001).

In our country, Ethiopia (2002) has investigated the application of CBR system in labor law domain. Ethiopia (2002) developed a prototype CBR system, Amharic legal precedent retrieval using CBR-Works tool. Ethiopia used 39 precedent cases to build and test the prototype. Using recall and precision performance measurements, she achieved 97.5% average recall and 47.8% average precision. She has tried to show retrieving similar cases for the current problems or queries. However, among of the four major applications of CBR: Retrieving, Reusing, Revising and Retaining, she was focusing on the Retrieving application mainly and due to the nature of the tool as seen earlier some ideas for Reusing and Revising. With regard to this drawback of the tool and application of CBR, the researcher has planned to carry out the four "Re's" applications of CBR with the help of the developmental tool jcolibri frame-work in this study.

Ethiopia recommended that adding high number of cases in the case base will increase the performance of the system and a continuation of the work by adopting the retrieved cases is important. Therefore, the researcher adds 11 more cases and 2 attributes to see this effect on the prototype system performance with in the 4 “Re’s” application of CBR.

2.8. Amharic Language Description

In Ethiopia, there are over seventy five native languages spoken all over the different regions of the country. One of these languages, Amharic, is the national working language of the country. Amharic is a Semitic language and the national language of Ethiopia.

The majority of the so speakers of Amharic can be found in Ethiopia, but there are also speakers in a number of other countries, particularly Eritrea, Canada, the USA and Sweden. As the evidence is found from site, http://en.wikipedia.org/wiki/Amharic_language, Amharic is the second-most spoken Semitic language in the world, after Arabic, and the official working language of the Federal Democratic Republic of Ethiopia. Thus, it has official status and is used nationwide. It is written using Amharic Fidel, “ፊደል”, which grew out of the Ge'ez scripts fidel ("alphabet", "letter", or "character").

As Baye (1992) described, the foundation for any languages are sounds. Sounds make phonemes. The phonemes in turn make up words, the words then sentences and so on. Sentences fully describe a thought. These language expressions are important to retrieve the legal cases in the CBR system to this research because Amharic legal Case management System in the domain of law has not been developed except only by Ethiopia (2002) considering only retrieval CBR application by using the CBR developmental tool CBR-Works. As a result of this, the purpose of this research is to accomplish the four applications of CBR methods: retrieving, reusing, revising and retaining. As expression of thoughts, a language is a collection of sentences in this case a collection of 50 cases.

To express thoughts through writing, symbols that represent sounds, words or phonemes are needed. These symbols have to be, of course, understood by the users of the language. Languages that have these symbols are called written languages whereas those that do not have them are known simply as spoken languages. This art of expressing thoughts through symbols is called writing and the nature of writing as writing system. Script is the term that is used to

describe the symbols as a whole. The Amharic writing system of a sound/alphabet consists seven orders or vowels as seen in Table 2.2 and Appendix II. Very few of the Amharic sounds their corresponding transliterations are described in Table 2.2 as examples.

1st order	2nd order	3rd order	4th order	5th order	6th order	7th order
ሀ	ሁ	ሂ	ሃ	ሄ	ህ	ሆ
ha	hu	Hi	ha	he	H	Ho
ለ	ሉ	ሊ	ላ	ሌ	ል	ሎ
le	lu	li	la	le	l	Lo
መ	ሙ	ሚ	ማ	ሜ	ም	ሞ
me	mu	mi	ma	me	m	Mo
...	...	...	...	...	...	...

Table 2.2: Sample Amharic alphabets with their transliterated symbol

There are 33 consonants consisting of seven orders and 44 liable alphabets in Amharic. So the total character will be 33 × 7 + 44 = 275 characters in Amharic language. But since presenting these Amharic characters in plain text file is so difficult using jcolibri, the transliterated English characters have been used in this study. However, there are different problems to retrieve Amharic texts from the system. Some of these major problems are listed below.

A. Redundancy and ambiguity of some characters

Particularly in the case of Amharic alphabet, there are different characters or syllabuses which have the same sound used simultaneously. But in Ge'ez these characters have different meaning. In order to retrieve texts in Amharic, these characters make difficult the retrieval tasks in the system. Thus, 4 distinct sets of 7 orders can represent the sound /h/ + vowel/order: ሀ ሐ ኀ ኸ

2 sets represent /a/ + vowel: አ ዐ

2 sets represent /s/ + vowel: ሰ ሠ and

2 sets represent /ts'/ + vowel: ጸ ፀ are the redundant sounds to request the query for retrieval purpos.

More over, the same sounds of their first and fourth order vowel (like ሀ and ሃ; ሐ and ሓ; ኀ and ኃ; አ and ኣ; ዐ and ዓ) and the same sound of its second and six order vowel ዉ /'wu' and ው /'we'

have also aonther ambogiuty to retrieve the best match texts from the system to the request queries. The 44 labialized consonant symbols (/k_wa/ - ኳ and ኩዋ, etc.) are also arguably redundant, wholly or partially, and there is considerable variation in the spelling of many words that may involve them.

For instance retrieving the query ከሥራ ማስወጣት may be represented in the system as ከስራ ማስወጣት፣ከሥራ ማሥወጣት፣ ከሥራ ማሥወጣት፣ ከሥራ ማስዎጣት፣ ከስራ ማስዎጣት፣ከሥራ ማሥዎጣት፣ ከስራ ማሥዎጣት and may not retrieved the reqired query.

Even the morphological analysis is important to avoid redendancies and ambiguities of such amharic words, the transliterarted words are simply used in this study because different similar sounds make uniform.

B. Compound word writing system

In amharic writing system, there are two words written together wich have different meanings to form a single word and to represent a single meaning. But the new formed single word can be written as a single word or a separate word conbination. Forinstance the word መኝታ ቤት - "megnita bet" which means "bed room" can also be written as: መኝታቤት and መኝታ-ቤት which has its own problem to be retrieved.

C. Different irregular spelling for a single word

For instance, the word "semtoal" which means 'He has heard' may be spelled as ሰምቷል፣ ሰምቶኣል ሰምቶአል፣ ሰምቱዋል or ሰምትዋል and also the frequently used term.

The word "Ethiopia" can also be spelled as ኢትዮጵያ፣ ኢትዮጲያ፣ ኢትዮፕያ፣ ኢትዮጲያ፣ ኢጦጵያ፣ ኢጦዮጲያ፣ or ጦቢያ. Thus, this is another Amharic writing script problem that a number of words in Amharic can be written with different spelling.

The other issues regarding to irregular spelling is transliteration of foreign words (like English sounds) into Amharic writing system. For instance, as stated by Ethiopia (2002) one can find the word 'television' translated into different forms, which include: ቴሌቭሽፕን, or ቴሌቭዢን.

However, the transliterated word has bees read as a uniform sound for different spellings in Amharic.

CHAPTER THREE

KNOWLEDGE (CASE) ACQUISITION AND CONCEPTUAL MODELING

Knowledge engineering (KE) is all about build, maintain and development of KBS in the field of AI. It has phases such as elicitation, representation, design (modeling), and implementation (Ferruccio, Nicola, and Paolo, 2010). These processes of activities are pillar to develop the KBS in general and CBR systems in particular within the big field of AI.

3.1. Knowledge Acquisition

Knowledge acquisition (KA) is an important part of developing a KBS using the appropriate methods that should be used for acquiring the knowledge needed for creating and testing the CBR system. Therefore, KA in this context is the process of gathering information from the field of labor law i.e. lawyers' decisions and using it. Typically it involves retrieving information from documentation, representing it in some way and translating it into a way that machines can understand.

KA, also called knowledge elicitation, is a technique that acquires knowledge or cases from the previous solved cases through analyzing recorded reviews or documents, and preprocessing the acquired knowledge with appropriate representation (see Fig. 3.1).

To develop a CBR application, first there is a need of acquiring knowledge (cases) from the FSC by reviewing the recorded documents which are previously decided cases by domain experts. Thus, 50 Amharic legal text cases were selected from these manual papers and they need to be preprocessed. That means the selected text cases were organized in the columns-row wise form with Microsoft excel 2007 application software then saved as CSV (comma delimited) (*.csv) form from .xlsx file format to perform cases with the selected jcolibri tool. Since cases are in Amharic font, they are not compatible with the comma delimited .csv file extension. This was a very difficult task to the researcher. So the researcher has been obligated to transliterate each Amharic text legal case into English. The collected Amharic text cases from the FSCE were the verdicts of both the FSCE and other courts if the dispute was previously established from other

lower court and it has got decision. However, due to time constraints, only verdicts solved in FSC is used for the selected attribute “Wusane”.

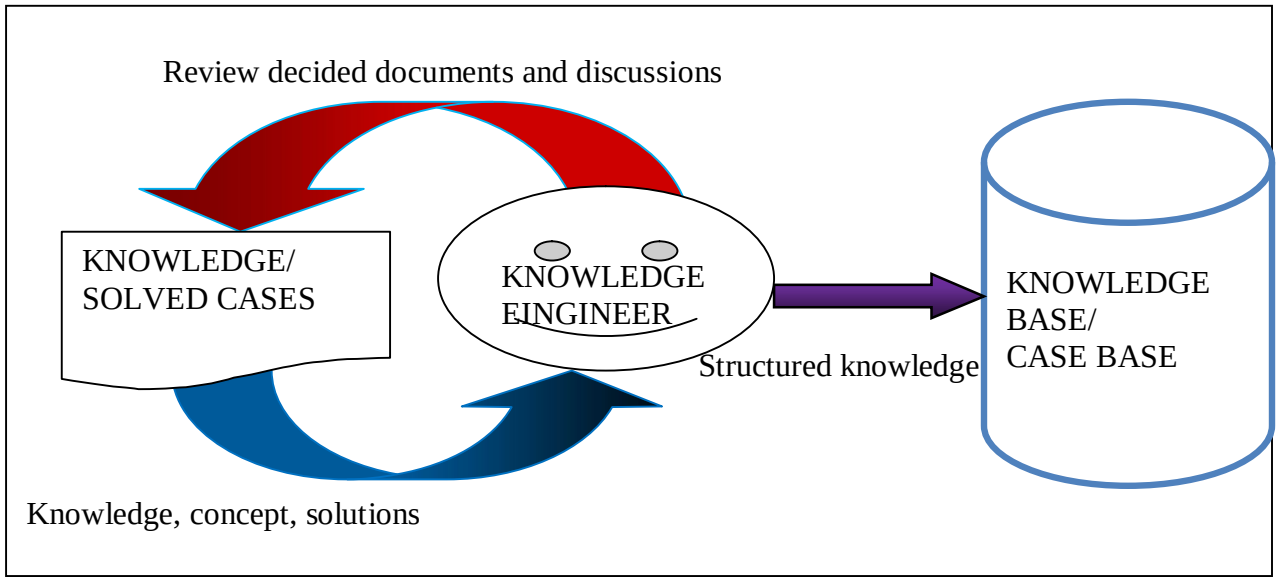


Fig. 3.1: Simple knowledge acquisition processes

3.2. Legal Case Representation from Text

Case representation is a CBR problem area that refers to selecting proper descriptors (dimension-value pairs) to describe and index cases for configuring CBR tasks with the help of Microsoft excel application software and jcolibri framework. Case representation is enabling the storage of cases in the case base according to a predefined structure. A case refers to specific experience or knowledge tied to labor law cases that is difficult remembering manually for future use.

Even though there are more than 10,000 solved legal cases in the FSCE, 50 cases are taken to develop and evaluate the performance of the prototype CBR system. The legal cases of the Court are the description of formal requests that realize parts of lawsuits. All these texts are written by judges who work in the Court and they have almost very similar backgrounds. These texts, which are recorded in paper, are collected and then organized in the appropriate way of representation i.e. organizing these raw texts in the selected attributes and their values in the

form of A1, A2, ... An with values of cases C1,C2,…Cm. Here n = 9 that mean the number of selected attributes are 9 and m = 50 that means the number of case selected are also 50.

There are several different ways to represent cases in a CBR system. Jcolibri supports three types of representations: plain text cases, cases stores in a DB and cases stored into ontologies. The researcher chooses a plain text file representational technique as the collected cases from the court are preprocessed into plain texts file to the jcolibri compatibility.

A case could be raised by either the employers or employees and solved by more than 5 lawyers in different times. These cases are received from the plaintiffs (applicants) in the hard written application form (see Fig. 3.2).

ከሣሽ/አመልካች/ይግባኝ ባይ፡ _____
ተከሣሽ/ተጠሪ/መልስ ሰጭ፡_____
የክሱ ዓይነት፡ _____
ፋይሉ የተሰጠበት ቀን፡_____
ፋይሉ የተመራለት ችሎት፡ _____
የተወሰነበት ቀን፡ _____

በዳኞች መካከል ዉሳኔ መስጠት፤

ፌደራል ከፍተኛ ፍ/ቤት መዝገብ ቁጥር _____
ፌደራል መጀመሪያ ደረጃ ፍ/ቤት መዝገብ ቁጥር _____
የ _____ክልል _____ፍ/ቤት መዝገብ ቁጥር _____
ሌሎች፤ _____
የጉዳዩ ደረጃ፡
መጀመሪያ ደረጃ ☐ ይግባኝ ☐ ሰበር ☐
የጉዳዩ ዓይነት፡
ፍትሐብሄር ☐ ወንጀል ☐ ስራ ክርክር ☐
ክርክሩ ገንዘብ መጠን _____ ለዳኝነት የተከፈለ ገንዘብ _____

የገፅ ብዛት (ፋይሉ ሲከፈት) _____
ቀጠሮ መመዝገቢያ:
_____
_____
_____

የሞላዉ ኃላፊ ስም _____ፊርማ _____ቀን _____

Fig. 3.2: Case receiving application form from offenders or plaintiffs.

After this form is filled by the plaintiff (ከሣሽ), the case is opened and the defendant (ተከሣሽ) should be aware of his/her case by the court. The defendant then prepares his/her opponent ideas in a clear description. Among the three types of disputes (የጉዳዩ ዓይነት): labor dispute (ስራ ክርክር), criminal dispute (ወንጀል) and civil dispute (ፍትሐብሄር) provided in the case application form seen in Fig. 3.2, the plaintiff should select labor dispute (ስራ ክርክር) when opening the case. Because of labor law is a focused area to this study with different case types (የክሱ ዓይነት) that are clearly defined in the proclamation No. 377/2003.

The purpose of the labor law is to make sure the fair and rapid settlement of labor disputes and to determine labor dispute settlement bodies and the procedures for settlement of labor disputes. As it is stated in the proclamation No. of 377/2003, labor dispute means any controversy arising between a worker and employer in respect to the application of law, collective agreement, work rules, employment contract or customary rules and also any disagreement arising during collective bargaining or in a connection with collective agreement.

The purpose of this research is to manage cases in terms of retrieving the old cases, reusing the retrieved cases, revising these cases and retaining the cases in the automatic machine learning approach with the help of indexing in vertical representation way. Therefore, these cases could be represented after collecting from the source with the attribute-values wise features to access them easily when similar cases found in the case base.

The case-base is a DB of previous solved legal case of constraints. Each case represents an individual problem solving episode. Cases store the characteristics of the dispute which are used by the domain experts to determine the action that will be used to address them. They also store information about the **repairs** (maintenance) that were used to solve new cases/problems which are used in the future to generate solutions to new problems.

Cases are represented through dimension (attribute)-case value pairs called descriptors. The attribute represents the column wise and the values represent the row wise to develop a case base. Selecting the proper descriptors characterizes case representation. The case representation helps to judges to retrieve the existed case from the case base and compare the similarities with the new one (query) and make a decision.

The cases might be represented by using the CBR tool called Jcolibri framework version 1.1 which supports the four “Re’s” CBR applications.

3.3. Attribute Selection

Attributes are dimensions or features that are used as a representative for selected case values written in column wise as seen in the earlier section. There are different possible attributes that might be used in representing the legal cases in the domain of labor law. In this study nine appropriate attributes are selected. These attributes are selected based on the DB features, discussions with 2 lawyers and the previous case analysis. The attributes values are used to retrieve the solution for a problem from the case-base.

The descriptive features (columns) are critical parts of the case and they represent the key feature of a case. There are nine attributes of legal cases selected to represent the case values in case representation for this research. These are: Judges/Lawyers “Danyoch”, Case type “Yeksu Aynet”, Plaintiff “Kesash” Defendant “Tet'eri/Tekesash”, Proclamation No. “Awaj Qut'r”, Article “Anqets' Qut'r”, Case opened date “Ksu Yetekefetebet Qen”, Case closed date “Ksu Yetezegabet Qen”, and Verdict “Wusane”. Each of the attributes is described as follows.

Case opened date: a time that used to indicate when the case has been opened for the first time.
Case closed date: a time that indicates when a specific case has been got a solution and closed.
Plaintiff: a person or an undertaking that could open the case to get justice when dispute has been aroused between him/her/them and other opponent (s).
Defendant: a person or an undertaking that could accused with specific case type or a person or employer that assumed to violet the right of someone.
Judges: are lawyer who provide the right justice to both plaintiff and defendant by examining the case and set decisions. They are expected to provide fair decisions to both labors.
Case type: is a type of labor dispute that were stored in the case base and/or might be raised by the offender or plaintiff to get the right justice. There are a number of case types such as: dismissal from work, over time payment, termination of payment contract, punishment of discipline, different payments that protect social security right, promotion, return to work, work position, services, illegal job transfer, mental health related cases, claim for damage against workers, etc.
Proclamation No.: it is a booklet that has been established by the house of federation to enhance the endorsement of the labor law or established by the concerned body for guiding rules and regulations about the labor. There are different established proclamation numbers by

concerned bodies. The main proclamation numbers used for this feature are: 42/85, 377/96, 209/55, Addis Ababa municipality number 12/9, “yeftabher snesrat qut'r 91”, 494/98, 25/88, 345/95, and underscore “_” for unknown numbers which could be analyzed by the lawyer’s general knowledge.

Article: it is a specific number that specifies a specific code of conduct within an established proclamation number. It also indicates the right and obligations of labors (both the employers and employees) what “have to do” and “don’t have to do” in their working life.

Verdicts: are legal court decisions or solutions in Amharic that have been decided by the judges who examine the cases raised by the labor(s).

These case attributes are selected by analyzing the prior solved cases, observing the given DB and consulting two lawyers who work in the court and a PhD in law student at AAU for their validity to represent the case values.

The CaseID, which is not regarded as attribute, is a number that uniquely identifies the case number. All cells in the plain text file have a unique name/number, for instance “10” to mean CaseID = 10.

3.4. Designing LCMS with CBR Applications

After the explicit knowledge is acquired from the source by the knowledge engineer (the researcher in this case), the next step is structuring the textual cases in the attribute-value forms as seen earlier. And now it is important to design the architecture of the organized cases in accordance with CBR applications for legal labor cases (see Fig. 3.3).

As seen in chapter 2 of this paper, CBR is an AI methodology which uses solutions to previous problems to solve similar new problems. A DB of past experience, called a case-base, is constructed and then used to solve new problems through processes of retrieval, adaptation, reviewing, and memorizing/learning. Each experience is stored within a case base, which usually represents a description of the problem, the solution, and an assessment of the outcome of using the case. Cases are knowledge container and units that describe an experience with dimension-value pairs (descriptors). CBR operates on the basis that ‘similar problems will require similar solutions’. Indeed, the concept of similarity, of both problems and solutions, is a key to the development of successful case-based systems.

Cases are used to represent domain knowledge of a case based system. Thus, cases in the case base represent collection of specific experienced, captured and learned situations of the application domain, (Aamodt and Plaza, 1994). It can be classified as past case and new case. 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.

Problems are represented by sets of feature indices/attributes which store case-value about the problem relevant to the problem solving it could be used for. As seen in chapter two, the 4 "Re's" of CBR applications have been used to design the system architecture in the modeling view of knowledge engineering. They are discussed below.

Case retrieval: When a new problem is encountered LCMS could first search for the most similar case (or cases) in the case-base. This process of **retrieval** may use a similarity measure to rank the cases in the case-base. In jcolibri framework, different local similarity measures are available to each data type. For instance, Equal and MaxString for string and file data type, Equals StringIgnoreCase and Threshold for Boolean data type, and Interval for integer data type are adjusted for use. But only Equal and MaxString local similarities in string data type are used for this study. The retrieved case (s) based on these similarity measures contains a solution that was used to solve the problem that it represents. This solution is used to solve the new problem.

Case reuse (Adaptation): it is rarely possible to directly apply the old solution to the new problem, and therefore solutions must be adapted to the context of the new problem. Solutions in cases may represent on the spot of the domain variables, or the methods (and method parameters) that were previously used.

Case revise (repair): The suggested solutions are adapted to the context of the current problem in revise method. After case are adapted as a solution to a new problem. In this situation cases might be repaired or maintained for storing purpose.
Most often maintenance in CBR refers to maintenance of the case-base in terms of the cases it contains. It can, however, also refer to the revision of the case indices or of the retrieval and adaptation phases (see Fig. 4.3).

Retain (learning): In the **retain** CBR task, any useful information or experience that can be gained from the current problem solving instance is retained for future retrieval reasoning. In this step, a given problem is mapped to the case structure of the CBR system and the most similar case(s) is retrieved from the case base. In this last step, retain, the retrieved, adapted and revised case can be added to the case base. In this way, the CBR system can learn (Aamodt and Plaza, 1994) and the learned case is stored.

A fundamental property of CBR systems is that of **learning** capability. Learning is achieved in CBR systems by memorizing new cases as they are encountered. Learning can be done with both human (judges) and machine (system). When supervised by human experts case-based systems can be trained both initially and continuously throughout their lifetime. Under automated operation, that is machine learning approach, case memorization could be undertaken with great care, to ensure that the cases stored are relevant, useful, and consistent with the existing knowledge.

"Remembering a case" to use in "later problem solving" is a necessary learning process. According to Kolodner (1995), features of the CBR approach:

- Learn from experiences,
- Learning to be integrated/based on reasoning,
- Learn from mistakes and do not repeat them,
- Allows a reasoner to solve problems with a minimum of effort,
- Provides a way of dealing with an uncertain world,
- In most cases, what was true yesterday is likely true today.

Judge in this manner has the operations of interpretation, retrieval, different analysis, application and modification and generalization the case using the case-base to decide on the new case/ problem. So management of cases becomes more familiar to lawyers.

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. As it is seen in the earlier sections, CBR is also paradigm for combining problem-solving and learning that has became one of the most

successful applied subfield of AI of recent years. Here 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 (Aamodt and Plaza, 1994).

Thus, learning in CBR systems is driven by both successes and failures. After a solution has been generated, the final step is to apply the solution, to repair it if necessary, and to learn from the experience. When the CBR process (learning) is successful; the resulting solution is stored in case base for future reuse. And when CBR process is failure; CBR is committed to the value of learning from failures as well as successes that is failures reveal that learning is needed and failures help focus decisions about what to learn: the needed learning must help avoid future failures (Leake, 1996). When a failed solution is repaired, the new solution is stored; this is simply learning from a new successful solution.

As shown in Fig. 3.3, the important method is to convert court decisions (texts) collected from the court into plain text cases to build the CBR system's case base from where the cases can be retrieved. There are two distinct phases or views for modelling the conceptual structure of the system, the transfer method and its implementation/modeling method.

As it is described by Cabrerizo, P´erez, and Herrera-Viedma (2009), there are two main views to knowledge engineering or modeling:

- the **transfer view** also called the **traditional view**. In this view, the assumption is to apply conventional KE techniques to transfer the acquired human knowledge into AI systems.
- the **modeling view** also called the **alternative view**. In this view, the knowledge engineer attempts to model the knowledge and problem solving techniques of the domain experts into the AI system. A KA process with transfer view can be supported by jcolibri tool. This tool provides an environment in which knowledge engineer or experts can search knowledge through an interactive process in CBR tasks. This modeling view looks like semiautomatic knowledge modeling method.

Both of these views were be used and applied to develop the prototype CBR system, LCMS, in this research (see Fig. 3.3).

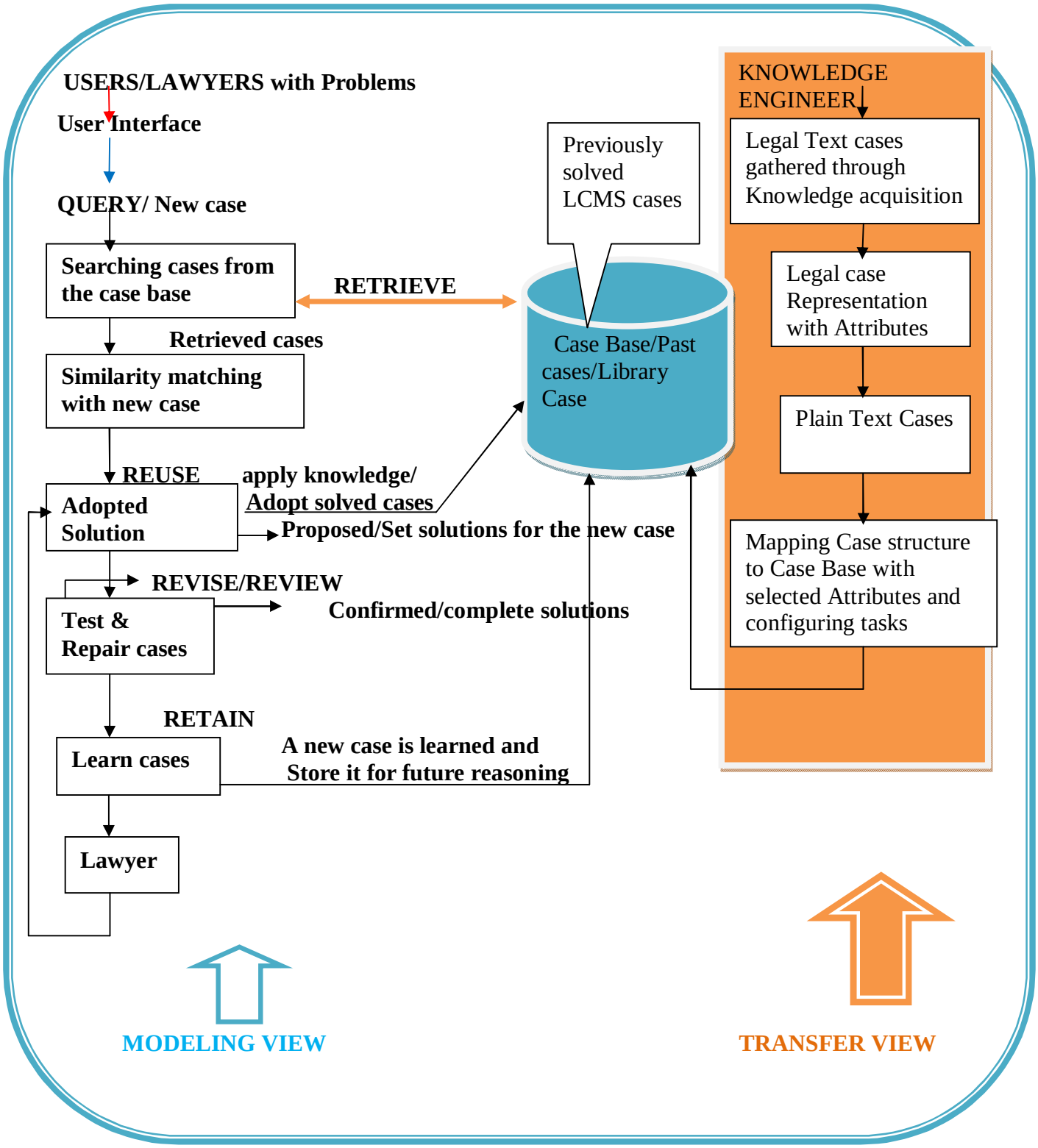


Fig. 3.3: Conceptual architecture of LCMS application in labor legal cases.

CHAPTER FOUR

PROTOTYPE CBR IMPLEMENTATION, TESTING AND EVALUATING AND DISCUSSIONS

The conceptual models depicted in Fig. 3.3 are configured and saved in the case base to the system functionality. The basic functionality is hand crafted (transfer view), but more functionality is added automatically (modeling view) when the system is configured and used. This functionality is taken place between a cyclic CBR application tasks.

4.1. Building the Case Base

The most important thing in any CBR system is the cases, and hence it is also important how they are represented and built with the given developmental tool. As seen earlier, a case is a knowledge container that contains the previously court decided cases in the preprocessed plain text file form. With jcolibri framework, a case base was created/built using this plain text file by indexing with attributes connected with cases in hierarchical structure. This case structure is important when loading cases into the system from a case persistency and when obtaining a query from the user before the retrieve CBR step. Therefore, with the vertical structure of cases in their representative of attributes and with connector the case base is built for managing legal cases. See the sample plain text file with attribute-value representation in Appendix I.

4.2. Configuring the CBR Tasks in Jcolibri

As see in the earlier chapters, CBR includes the four "Re's" tasks: Retrieve, Reuse, Revise and Retain. These tasks can be performed with different methods of CBR using jcolibri tool. To use the jcolibri framework for task manipulations, first it is important to launch the main window by clicking on **JColibriGUI.bat** file on the extracted file location and it becomes ready to use. By clicking CBR on the menu at the top, select **start new CBR systems** and give its **CBR application name** as "LCMS" shown in Fig. 4.1 for accessing the CBR application tasks in later use by clicking "OK" button.

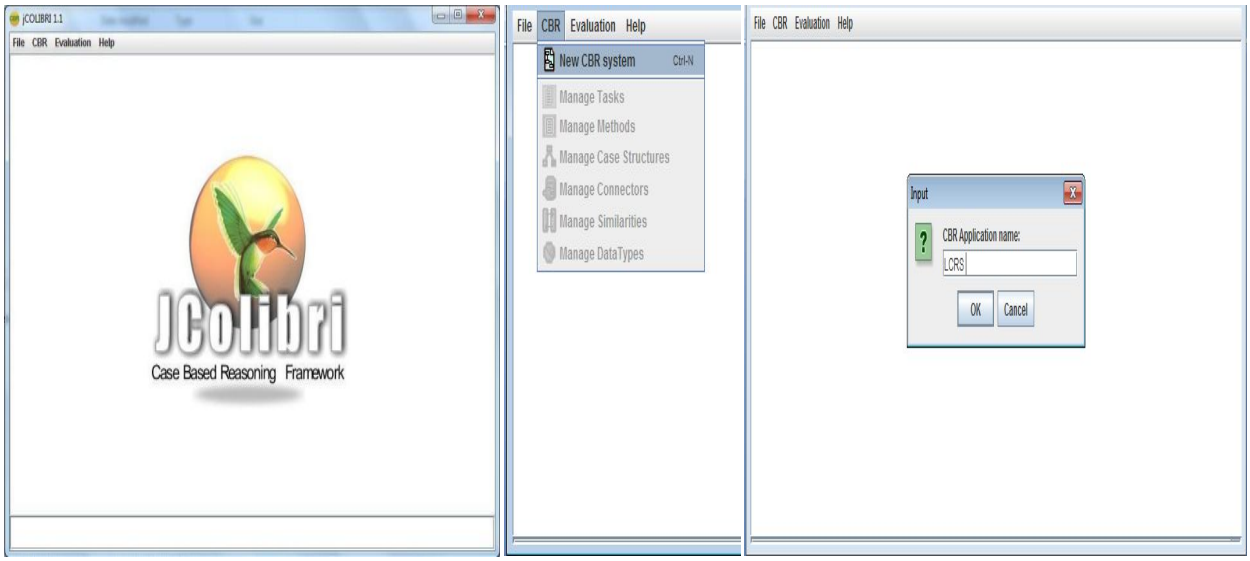


Fig. 4.1: The main and starting window of the jcolibri framework

Therefore, the development of a prototype CBR system, LCMS, application in the domain of labor law with jcolibri was done in consecutive four processes. These processes are dependent to each other, like connecter depends on case structure of application. Each task or process could be classified and configured properly to functioning of the prototype LCMS model. These classified tasks in labor law legal case management application are:

1. Managing/Defining the Case structure
2. Managing Connectors
3. Managing Task/Method

4.2.1. Managing (Defining) the Case Structures for the Prototype LCMS

When structuring cases in a vertical indexing manner, case attributes are indexed with their vocabulary concepts. The vocabulary in this way, defines the information entities (properties) and structures (e.g. attributes' Name, Data types, Weight, and Local similarity) that can be used to represent the assigned values of properties. This means that case structure would cover the different features or attributes that a lawyer would take into consideration when deciding to do an operation for decision purpose. A case component or compound attribute is an attribute which contains many simple attributes.

There are several different ways to represent cases in a CBR system in jcolibri such as: textual cases (plain text), cases stores in a DB and cases stored into ontologies representations. The researcher chose to plain text case representation in this research. Therefore, the collected cases from FSCE are saved in text file format from the .csv (comma separate value) file format after pre-processing.

As shown in Fig. 4.2, a visual CBR developmental tool jcolibri can be used to define case structures easily. The window is divided in two regions. The left panel displays the structure of the case as a tree (vertical), and the right panel shows the property values of the selected attributes. A case is composed by three components: **description** (describes the problem), **solution** (represents a possible solution approach) and **result** (reveals if the proposed solution is able to solve the problem). Result case part is not used to configure attributes of cases manually like description and solution but it is important to display the proposed solutions automatically when retrieving with a given query in the system. Description and solution are collections of Attributes that are two types: simple or compound attributes. These attributes have significant impact on LCMS functioning. The researcher used these attributes that let to build a hierarchical (vertical or tree like) case structure.

Therefore, by using jcolibri GUI the case structure is indexed with defining simple attributes that describe the cases together with their characteristics of Name (Danyoch), type (String), weights (1.0) and local similarity function (Equal). Compound attributes collect other simple or compound attributes allowing complex case structures. The properties of the compound attributes are the name and the global similarity function.

When the case structure is saved, it generates an xml file with the structure information (see Fig. 4.2). It could be used later to configure a connector for mapping the cases to the chosen persistence media. This mapping (connector) is also saved to an xml file (see Fig. 4.4).
All of the case attributes have string type and equal and MaxString local similarity (see Table 4.1) and the global similarity of these attributes is the average of all simple case attributes (see section 4.3).

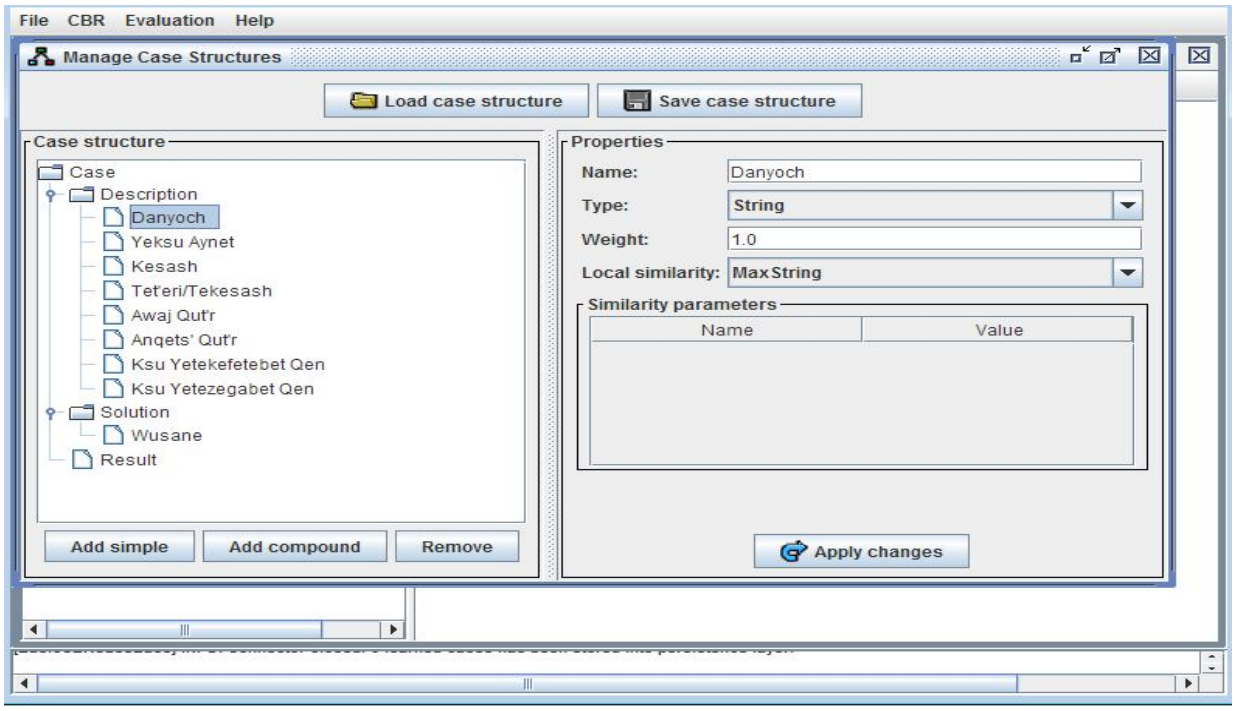


Fig. 4.2: Jcolibri window for defining Attributes and Case Structures

4.2.1.1. Description of Case Attributes

Defining case structure in jcolibri are done by using simple manage case structure window. It is very easy to define with jcolibri. Because just have to add attributes in description and solution of case structure and set properties of attributes in the opened jcolibri window. It creates codes automatically and saved in xml file. Most significant attributes that are set by declaring weight have higher weight as compare to other attributes.

A case has 8 description attributes and 1 solution attribute. Solution attribute is used for storing solution that is set manually after finding best selected case.

Description Attributes			
Attribute Name	**Data Type**	**Weight**	**Local Similarity**
Danyoch	String	1.0	Equal
Yeksu Aynet	String	1.0	MaxString
Kesash	String	1.0	MaxString
Tet'eri/Tekesash	String	1.0	MaxString
Awaj Qut'r	String	0.95	Equal
Anqets' Qut'r	String	0.95	Equal
Ksu Yetekefetebet Qen	String	0.9	Equal
Ksu Yetezegabet Qen	String	0.9	Equal
Solution Attribute			
Wusane	String	1.0	MaxString

Table 4.1: Description of case attributes in jcolibri

4.2.2. Managing Connectors

Connectors (with Plain text file) are persistence of cases which are built around the concept of connector in jcolibri. After the case structure is configured, the cases persistence concept is then built around those connectors representing objects (attributes) that know how to access and retrieve cases from the storage media (case base) and return those cases to the CBR system in a uniform way. Therefore, connectors provide an abstraction mechanism that allows the developer to load cases from case base sources in a transparent way.

The labor legal cases are stored in plain text file. By using jcolibri tool, it is very easy to connect text case bases with CBR applications or tasks. It has three choices of type for connectors to connect case bases with applications as shown in Fig. 4.3. Plain text file, SQL DB and Other (like XML files, DLs, ontology) connectors are already available for use in the window. Other connectors can be included depending on the specific application requirements based up on the developer's need. However, Plain text file connector is chosen by the researcher to connect the cases with the CBR tasks and to manage this connector (see Fig. 4.3).

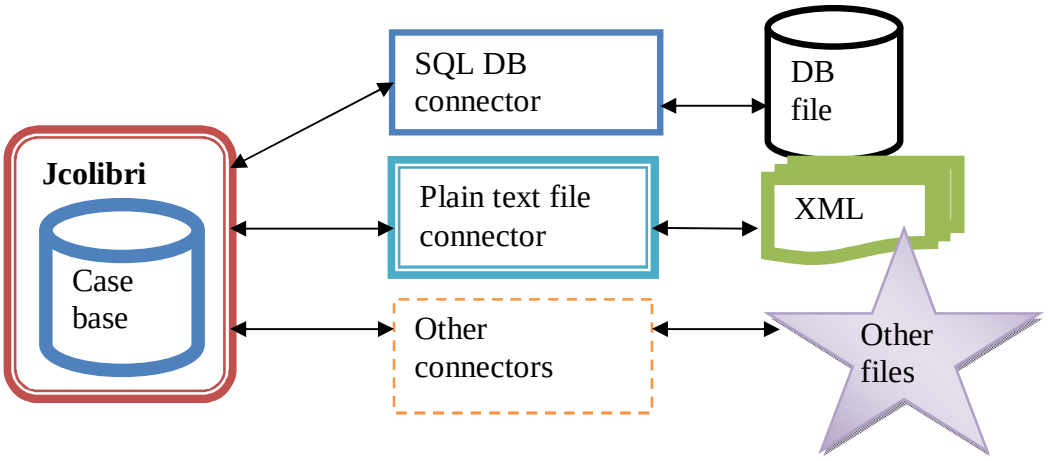


Fig. 4.3: Jcolibri case base connector types

Jcolibri offers a graphical tool that is used to easily configuring connectors to load existing case bases in xml file formats that is the case structure (see Fig. 4.2). In the plain text file connector, it must be specified with the path of case structure, the path of text file, and the delimiter of comma punctuation mark "," (see Fig. 4.4). All the attributes of a case should be mapped as shown in Fig. 4.4. This is connector's responsibility to retrieve data from case base and return it back to GUI or application.

Fig. 4.4 shows how the case structure is mapped with plain text file (using the connector with the same case structure contained). Now the connector like case structure is also saved in xml form for later process to manipulate the task and method configuring in jcolibri.

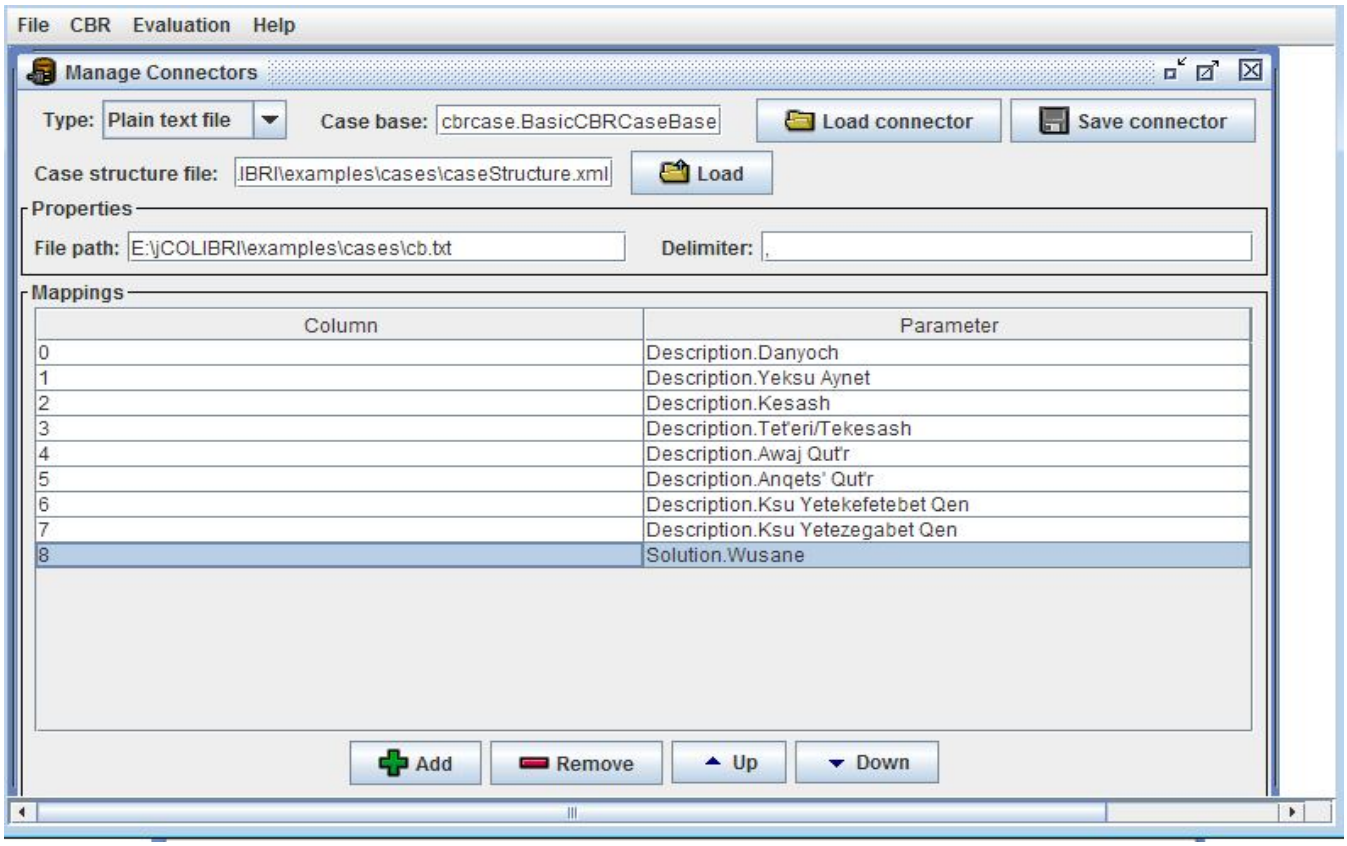


Fig. 4.4: Managing Connector for attribute mapping with the plain text file connector.

4.2.3. Managing Tasks/Methods: Configuring the CBR Applications

This is an important step for creating a new application where the CBR application is configured. In jcolibri, CBR applications are composed by a PreCycle that load cases, the typical 4 “Re’s” Cycle and a PostCycle that stores cases into the persistence layer.

The CBR model developmental tool, jcolibri framework, has two types of tasks packages: Core and User defined packages. These packages can perform and execute tasks and methods in a sequential process. A core package task in this research was used. This package includes the three core persistence: PreCycle, main CBR cycle and PostCycle which incorporate different major tasks and other subtasks (decomposition PSMs).

The core of jcolibri package is called CBRCore class and it is the most important component of the framework. The core is in charge of the application, and must always be present for an application to run.

When building a CBR system using the jcolibri framework, it is important to implement the interface. This interface or method of CBR application is composed by the following mentioned four core packages.

- a **configuration** method to set up the application.
- a **preCycle** that loads cases and prepares the application to run.
- the **cycle** method that runs a CBR step using the given query.
- a **postCycle** in charge of finishing the application.

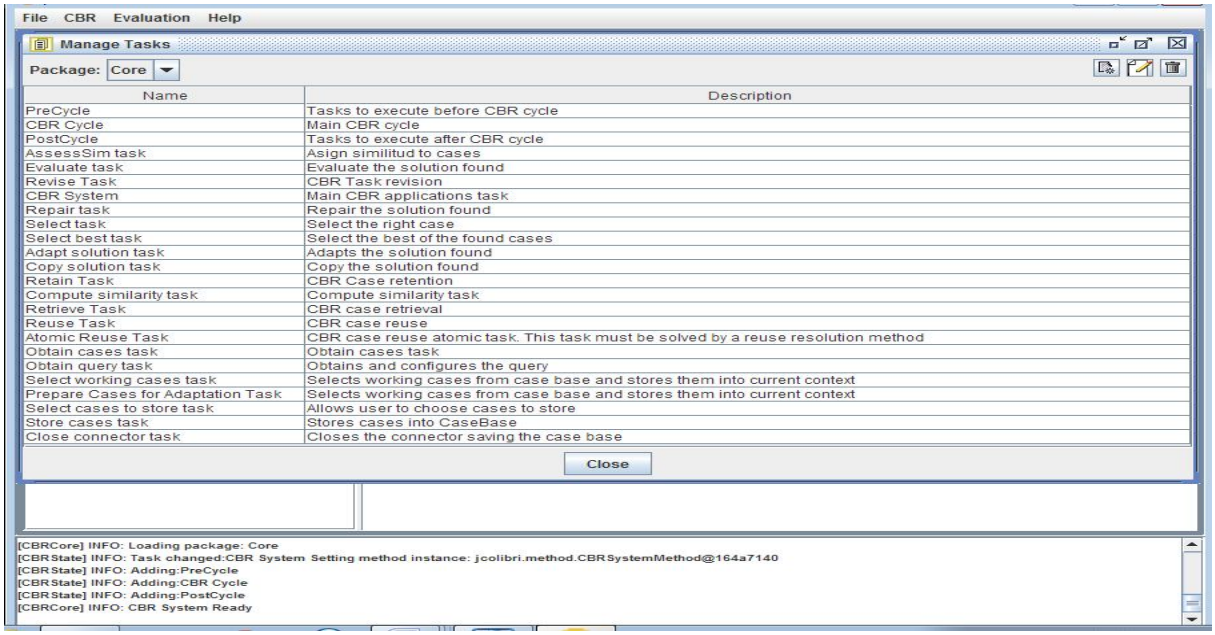


Fig. 4.5: Managing the core packages with tasks

Normally, there are two types of PSMs:

- Decomposition: It is used for dividing main task into subtask for instance, retrieve, reuse and retain.
- Resolution: Its solve tasks directly for instance, reuse.

Configure: The configure method sets up the application with regards to creating the objects of a case base, structure, connector and so on.

It organizes the cases into a normal LinealCaseBase, where cases can be stored without closing the application. The method also initializes the similarity, result, revision and retains dialogs. The query dialog is initialized in the main method.

Precycle: After the case structure and connector are configured the next tasks is deploying the cases. To do so, the first task is configuring the precycle task to obtain cases task. The first thing that happens in the precycle method is that all the cases are being loaded through the connector into the case base. Each case is printed out with its solution attached to it, although this is just for debugging purposes. The precycle reads the cases from the case base into an in-memory case base structure.

The precycle has its sub task named obtain cases task. Thus, defining the path of the connector in its subtask is required then it could retrieve the case from case base. This task can execute cases before CBR cycle.

Main CBR cycle: It is main task of CBR application and includes obtain a query task and the four "Re's" CBR cyclic tasks. It also has sub tasks.

Obtain query task: it is the first task of the CBR system and it obtains the query that contains the description of problem and is going to be used to retrieve the most similar cases, applying retrieval task. The framework supplies a useful and general method, named Configure Query Method, to solve this task. This method reads the case structure from the XML file that stores the case structure, and dynamically generates a visual form that can be used to introduce the query data. This form takes advantage of the best graphical components available according to the case structure and the types of the attributes.

Retrieve Tasks: it retrieves CBR case. It consists of following subtasks:

- Select working cases task: It selects working cases from case base and stores them into current context.
- Compute similarity task: it computes similarity task.
- Select best case: It shows the best matched of cases. It means that best matched case is set.

Reuse Task: it reuses CBR case. It has three subtasks. These subtasks are the following:

- Prepare Cases for Adaptation Task: It selects cases from case base and stores them into context with the specified path of case structure in this method.

- Atomic Reuse Task: It should be resolved by reuse resolution method.
- Reuse Task:

Revise task: is used to correct the cases by the user. In this task users are expected to modify the cases based on their knowledge and decisions for the next use to solve a certain problem.
Retain Task: Its main aim is CBR case retention. Subtasks of this task are following:

- Select cases to store task: It gives authentication to user for storing case.
- Store cases task: it stores cases into case base.

In order to create GUI for query tasks, it is important to give the path of case structure in it by clicking the instance button. Then it is associated with the case attributes that already indexed features. For this reason, it can be called as obtain query task. The cycle method executes the CBR cycle with the query set in the query dialog interface and cases could be retrieved. The retrieved cases are printed out (debugging) and shown in the result dialog window. Here the user chooses which case he/she wants to use from the retrieved ones, where the first case shown is the most similar (higher similarity value).

When the solution from this case has been applied to the current case, it is shown a revise dialog window. The revise dialog is meant to be used by the user to correct the application with regards to the actual solution/outcome. If the solution was correct the user may simply press next in the dialog. If however, the solution was not correct, the user can act as the domain expert or lawyer and revise the solution for this current case. Regardless of the manipulations, the user is met with a new dialog window called retain. In this dialog the user can either choose to store the query case with the applied solution, or simply ignore it.

The CBR cycle is now complete and the user can start over with a new query or end the application. If the user opts to end the application, postcycle is called to close the appropriate objects used.
Post cycle: It will execute after a CBR cycle. Its task is to close a connection between case base and GUI that is run to shut down the applications, typically closing the connector and DB server.

To conclude, the first task has been started with clicking on PreCycle at the left on the jcolibri window, and at right side window for task configuration is displayed as stated in Fig. 4.6. Each

task can be solved by several methods and can choose the most appropriate and instance it. When Click on PreCycle on it, cases task is obtained for creating instance. After clicking on instance button, new window could be appeared where path of connectors can be given. The same procedure could be performed with the CBR cycle and give path of case structure file for add instance of obtain query task. It is automatically created by the "obtain query task" method using the case structure xml file. Likewise, add instance with same procedure for four "Re's" tasks and each task is tested during configuring of the application using a little red  button called "**Solve to …**" located on upper left corner.

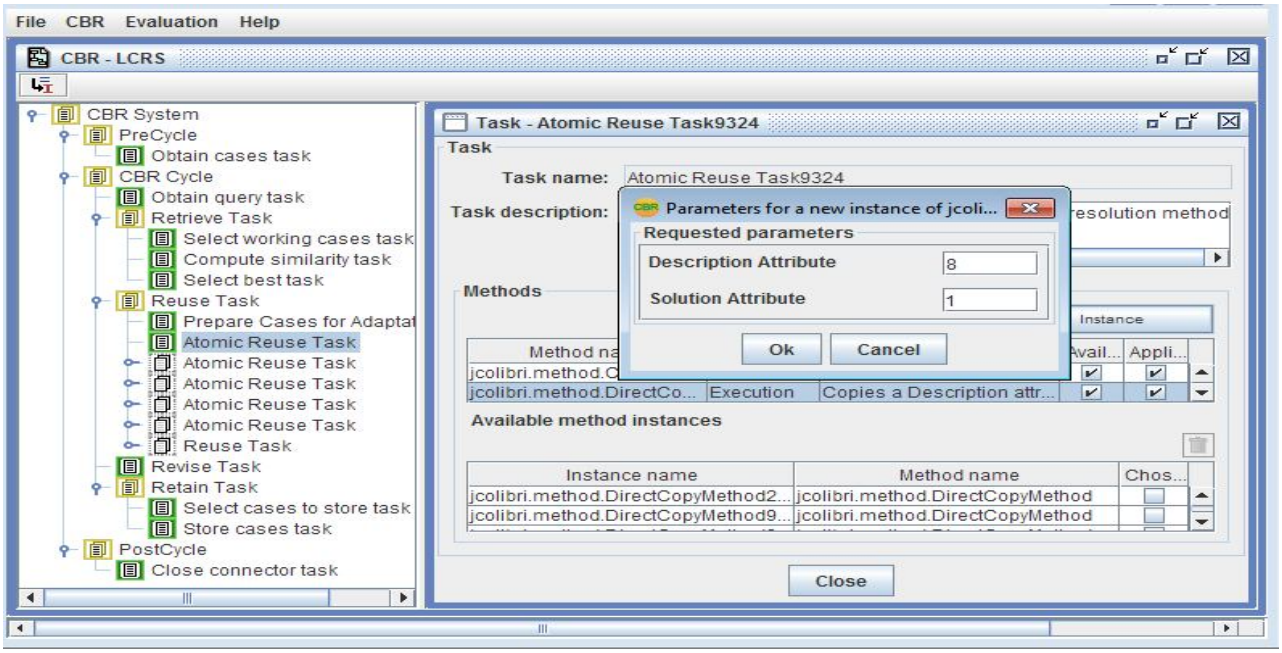


Fig. 4.6: Configuring Tasks and Methods

Each task seen in the above including the subtasks are deployed by using different CBR methods (see Fig. 4.7). So CBR core methods were used to solve tasks of the LCMS application. Some of the core methods of jcolibri are described below.

1. LoadCaseBase Method: loads the case base from the persistence layer using a connector configured in the xml file and it uses this connector to retrieve case base. It has the three types of connectors in jcolibri as seen earlier.

2. ConfigureQuery Method: It automatically picks the case structure and shows a window where the user has to give the input or query to retrieve similar cases to this query.
3. SelectBaseCase Method: User will get best match case from case base. Responsibility of this method is to provide best case after matching each attribute of case base with input values.
4. SelectSomeCase Method: In this method, user can get desired number of cases which are best match with hi/her input.
5. NumericProportion Method: it is a very important method, which calculates numeric proportion between description attribute and solution attribute. This method is sub method of reuse method. It resolves reuse task.
6. RetainChooser Method: This method will allow the user to choose the method that will store case base. User can choose that he/she want this method to store in case base.
7. ManualRevision Method: it allows the users to modify cases. User can change case according to his/her will especially in the revise task.

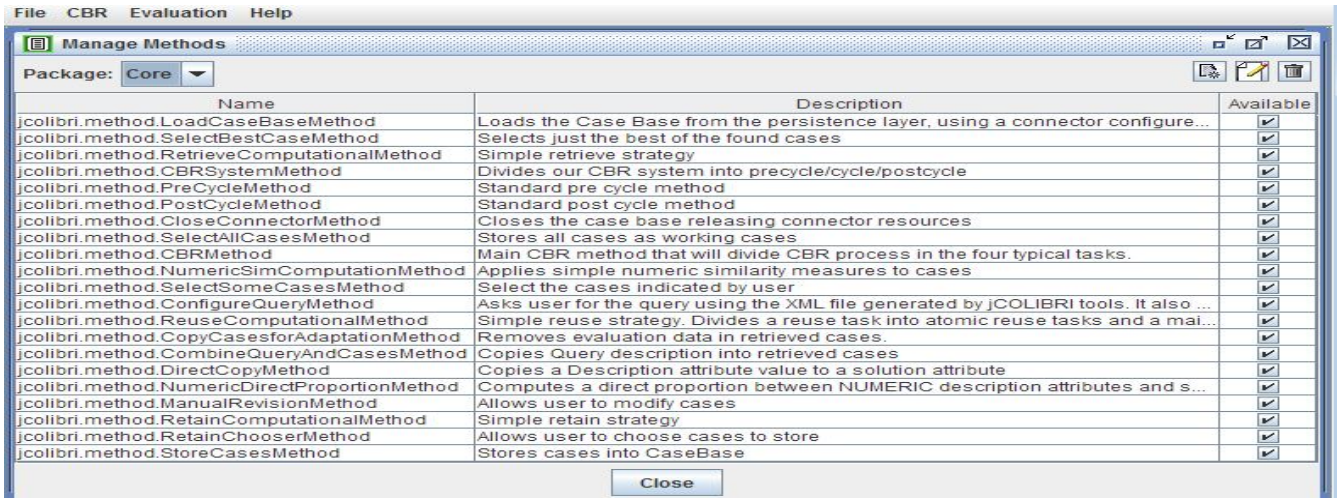


Fig. 4.7: Managing Methods to configure tasks.

4.3. Implementing the Prototype CBR Application

Now configuring the CBR application is completed. The designed prototype CBR system, LCMS, can be executed from the task/method configuration interface (see Fig. 4.8). jcolibri stores all the configured cases using different xml configuration files. Xml intends to be a

standard interchange language of data between computers, not to be managed directly by humans. When the application is executed or run, the framework core reads these files to knowhow to configure the prototype CBR system LCMS. This configuration files were written or modified by hand although it can be a very tedious task.

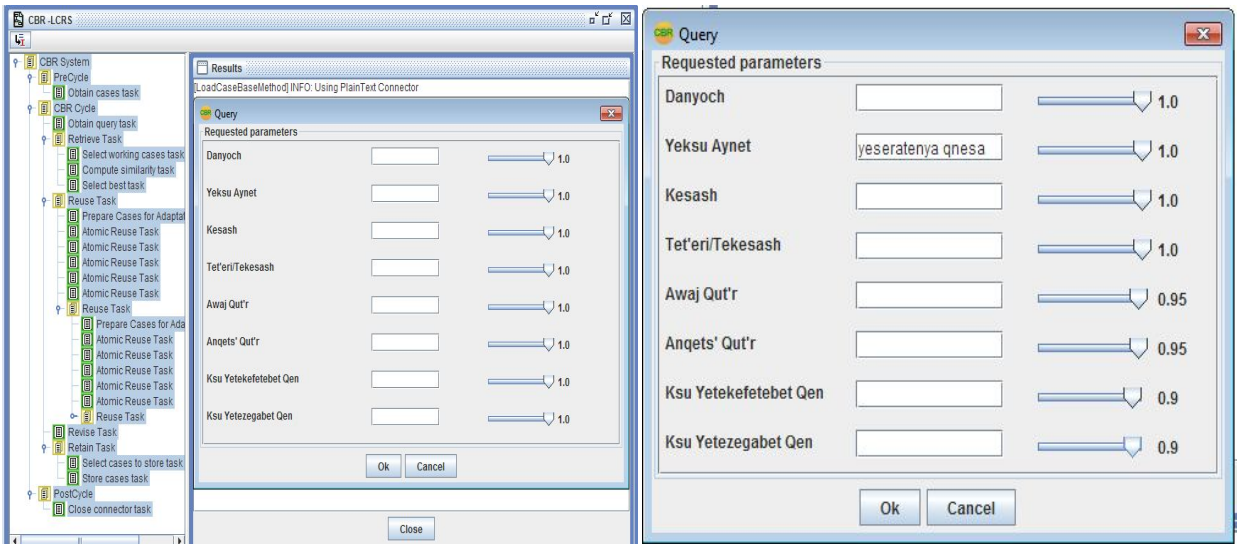
Configuration and the total CBR task could be saved for future use and modifying. On the main window (see Fig. 4.6), each one of the four CBR tasks involves a number of more specific subtasks (see section 4.2.3 and Fig. 4.6). The configuration of CBR application is finished when all the tasks have been configured.

After the task and method configuring is completed, it is time to deploy the result and the application could be tested the system from inside the graphical interface by selecting the tasks that are going to be executed in the tasks tree inside the graphical interface and pushing the  "**Solve to …**" red button on the top left corner of the jcolibri window to retrieve similar old case (see Fig. 4.8a). The effects of the execution are shown in the results window (see Fig. 4.8a). Then the GUI window for interring the query or new case by the user is displayed as seen in Fig. 4.8a. Here the required queries should input into the provided interface. But a case type, which is "yeksu aynet", is mandatory while others are the optional to be interred into the GUI but it is required to fill the features in the revision step for retaining purpose as the lawyers' desire. The possible query concepts or case types for this attribute are: *"yeseratenya qnesa, yalagbab kesra mesenabet, Yedemwoz ch'imari, t'qmat'qm, yesra wul mequaret', yeamet ereft kfya, yalagbab yesra wul mequaret'ena lyu lyu kfyawoch, yet'ureta abel, yedereja edget, yewuzf demewez kfya, yalagbab yesra wul mequaret', yesra snbt kfya, yedemewez kfya, discipline, yetrf gize kfyana t'qmat'qm, demewez, yewuklna slt'an, yesra bota zwuwur, yet'ureta abel, yalagbab yesra snbtna wuzf demewez and yesra wul mequaret'"*

The users are now required to fill the interface and adjusting attribute weights as seen in Table 4.1 and Fig. 4.8 b. Then by clicking on **Ok** button, the consecutive step reuse (adaptation) is ready for application. This technique is emerged naturally (automatically) during the implementation without much planning (visible application) in advance.
In the reuse method, the solutions in the retrieved cases are then used to solve the new problem.

The reuse step adapts the solution of the retrieved cases to the given problem. These adapted solutions are checked by an expert or a user of the CBR system in the revise step. The result of the check can be stored in the specific case.

After a while, a third CBR step revise is substitute followed by the adaptation cycle. At this task, the result is displayed automatically for revision purpose with the solution attributes and values that are important to make legal decisions (see Fig. 4.9a). Now the revision step of the CBR cycle became implementing with manually filling and its automated generated queries filled in the GUI retrieval cycle. At the moment layers have a chance to modify the attribute values and /or proposed solutions and add more cases in the case base as a confirmed (completed) solution.

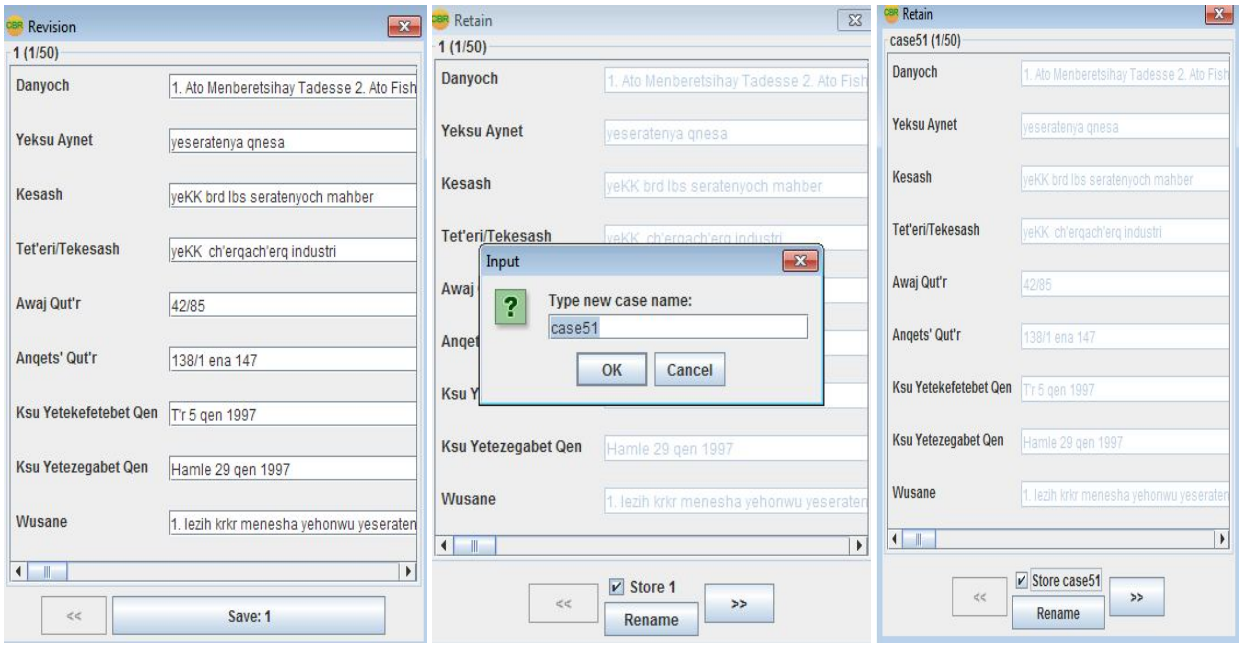


a) Provided interface with selecting tasks b) Query filled interface with attribute weights

Fig. 4.8: GUI Window for Case Entry into the Case Base

In reality would probably even a fully automated step contain the possibility for manual pinch, but rarely needed. A completely manual process has the advantage that the user has to evaluate and control the case to expert decisions. A partial automated revise step could be made and thus the revise step became manual and work on this process could be postponed to retain (store) cyclic application at the end of the implementation after saving changes of the case with caseID for latter consecutive task use (see Fig. 4.9a). When saved this application, retain window becomes available to retain or store a case (see Fig. 4.9b).

The last important CBR cyclic step is a retain application. As seen in the revise application in the above manual-human learning approach is more considerable as lawyer are required to evaluate the cases for decision making purpose to the created problem at the movement. However, in the retain application is considered as the machine learning approach as is used to store the case for future use and the system learned this stored case. In order to store the case in the case base for future retrieval, the "store caseCaseID" should be activated as "✓" and adjusted the new case name if needed (see Fig. 4.9b) and then the saved case name is stored and displayed as its name for instance case51 in this purpose at the left top corner of the retain window (see Fig. 4.9c).



a) Revise b) Retain with name input c) Retain after case store

Fig. 4.9: The revise (a) and retain windows (b and c)

At retain task application, the user can't modify the attribute values and/or the proposed solutions since it is completed at the revise task of CBR applications but the user can read all the cases stored by clicking the ">>" next button in a ranking order based on the similarity values. However, it is also possible to retrieve null results. If the requested query interred within the displayed GUI does not completely match with the old cases stored in the case base during retrieving, then the copy of attributes are displayed with 0 case similarities on the GUI of the

revise task to mean null instead of their values as shown in Fig. 4.10 because there might not been represent similar cases to this query. In this situation, the domain experts are required to revise the decisions manually in the revision process to insert values and store this new null case with valid solutions within the case base for future use.

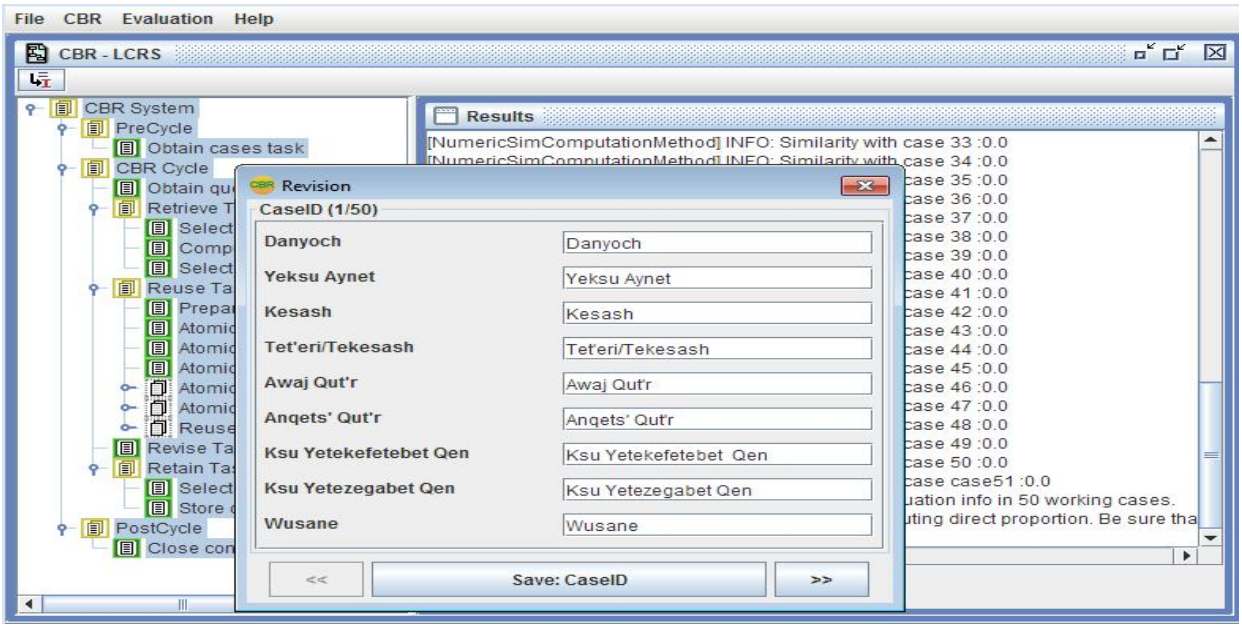


Fig. 4.10: Null results for unknown query in the case base

4.4. Similarity Measure of Cases

The notion of similarity plays an important role in CBR since cases are selected based on their similarity to the current problem in the ranked order. The established Case Description and Case Solution could represent the way to find the similarity between two cases and their attributes. Similarity is usually produces a similarity assessment as a real value from [0; 1].

Retrieving the best similar cases to the query from case base using the developmental tool jcolibri is based on the nearest neighbor algorithm. The similarity function manipulates computing the similarity between the stored cases in the case base and the new cases (query), and selects nearest similar cases to the query. Nearest match can be represented in the following equation.

$$Similarity(T,S) = \sum_{n=1}^{n} f(Tj,Si) \times Wi$$

Where,

T= target case (query)

S= source case (case base)

n= number of attributes in each case (i.e. n = 10 in this case)

I= individual attribute from 1 to 10

f= similarity function for attributes I in cases T and S

w= importance weighting of attribute I

The average score (global similarity) of each attribute between the existing case and the query are computed and the result is assigned to the object (the similarity between the stored case and the query). And then the computed values of similarity among the retrieved cases are displayed as shown in Fig. 4.11. Similarity in jcolibri works in two ways, global (for compound attributes) and local (for simple attributes).

A. **Local Similarity:** It is used to compare simple attribute values. The two types of local similarity used in this research are:

Equal: when equal local similarity is selected for each attribute, the input and value of case base must be match. If the interred value matches with the stored case exactly then it will get result otherwise match failure.

Equal returns either 1 or 0 depending on whether case one is identical to case 2 (query) or between 0 and 1 if partial match. The class of compute method uses the equal method implemented in the jcolibri managing methods.

MaxString: the attribute values in the system matches with the requested query by using the maximum string length. This can enable the lawyer user to prove their decisions at maximum string level as they want on the provided GUI especially in the revision task.

B. **Global Similarity:** It is linked with compound attributes and used to get similarity of collected attributes in unique similarity value. A common Global similarity function is the average over simple collected attributes. In the instance of case Description the similarity

between a case and query is calculated by averaging(global) over the similarity between each simple attribute(local), where each attribute is assigned their own local similarity function.

Modeling similarity means decomposing the similarity function according to the vocabulary. This decomposition is done in such a way that local similarity functions for each attribute mapping preferences. Local similarities are then aggregated into the global similarity by an appropriate combination of the local similarity values. This aggregation also takes the different weights of attributes into account.

When two cases are compared, the local similarity functions are used to compare simple attribute values. Local similarity function such as Equal and MaxString work on simple attributes. Then the similarity value of two cases is computed as the similarity of their description concepts. The available similarity measures are listed in a configuration file, and can be managed using the GUI. These functions compute the similarity between the query and the case, and are used to choose the most similar case to the query. For instance if the user requests for the query "yalagbab kesra mesenabet", then the following similarity measure results could displayed.

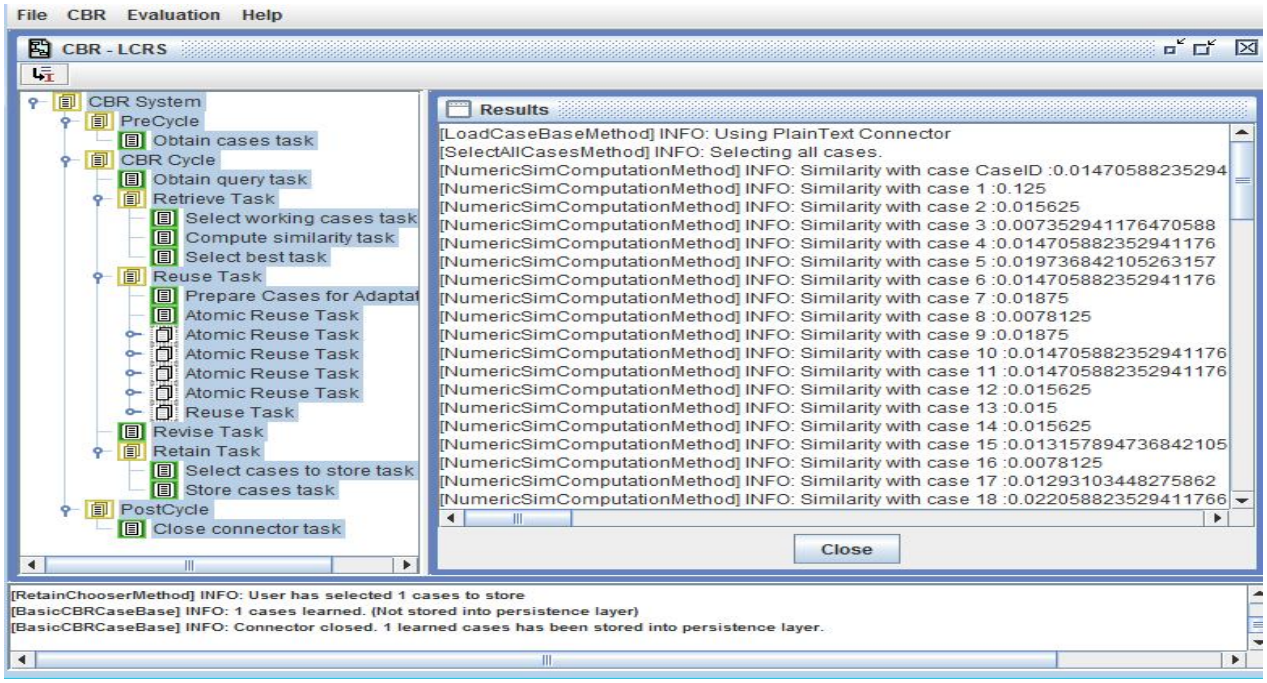


Fig. 4.11: Similarity computing results [0-1] between old cases and the query

4.5. Evaluation of LCMS Performance

After implementing the prototype system, the final step is measuring the performance of the system to achieve the set up objectives. One of the objectives of this research is to test and evaluate the performances of the designed prototype LCMS. The developed system, LCMS, is tested and evaluated to check whether the objectives of the research are achieved or not. The testing task is including the CBR application tasks during development, the user acceptance and statistical analysis in terms of precision and recall to evaluate the performance of the prototype system.

4.5.1. Testing the Developmental Functionality of LCMS

It is important to test the system regularly during the development phase, to be able to find design faults related to new implementations. LCMS has been tested regularly during implementation phase. As it is seen in chapter 3, the 50 previously solved cases were collected to develop the prototype CBR system. The case base contained 50 plain text cases as the basic case base, and more cases could be added automatically during the execution of the tests.

To test each tasks of the CBR application of the prototype, retrieval of previously stored cases to solve new problems is the first step in any CBR application. Retrieval of similar cases to the new case from previously solved cases is followed by the reuse of similar solutions. Thus, all the selected 50 cases were retrieved with their similarity values during the process. The similarity retrieval of cases in this research is performed using the nearest neighbor retrieval algorithm in jcolibri framework. Nearest-neighbor retrieval technique is used to measure similarity between source case and case which could be searching. If case is not matched exactly with CBR library, then CBR system will return nearest match.

The reuse application in this phenomenon has been performed automatically. After retrieving the cases, it is possible to see all the adapted 50 case in the system in ranked order of the case similarity by executing the reuse task individually. In the same manner, revision task displays the GUI with the attribute and their values with save button. The final retain tasks have been tested after a case is saved in revise task. In this situation a saved case is learned and stored within the system. So, the machine learning approach is performed from this final CBR task while the human learning is manipulated in the revise task. For more detail, see Fig. 4.9.

4.5.2. Statistical Analysis Evaluation

The statistical evaluation the same as developmental phase testing uses 50 legal cases that have been collected from FSCE. In addition to the user acceptance, the effectiveness of the retrieval process of the LCMS was measured by using recall and precision which are the most commonly used measures for evaluating the system in measures of performance of the retrieval process in information retrieval and CBR applications for this study. The harmonic mean of precision and recall which is called F-measure is also used to measure the system performance in statistical analysis. It allows weighting a user's preferences between precision and recall, and expresses performance in a single statistic value. However, the resulting number cannot be easily interpreted, as it is used mainly of academic interest. Thus, the F-measure is not used and appropriate for evaluating LCMS in this study.

Recall (sometimes called coverage) is the ability of the retrieval system to retrieve all relevant cases to a given new problem (query) from the case base while precision (sometimes called accuracy) is the ability to retrieve top ranked cases that are mostly relevant to a given query in a threshold or count-based technique i.e. [1, 0.8) threshold similarity boundary. This implies that the number of cases and the ability of the case base to provide an adequate solution from a maximum similarity threshold of 100% to a minimum similarity threshold of 80% i.e. between 1 and 0.8 case similarity values. The computational equation is:

$$\text{Precision} = \frac{\text{number of relevant cases retrieved}}{\text{total number of retrieved cases}}$$

$$\text{Recall} = \frac{\text{number of relevant cases retrieved}}{\text{number of relevant cases in the case base}}$$

The statistical analysis testing technique is performed by identifying the relevant and retrieved cases with the given test query that were provided to the three of the six evaluators of the system performance again. The totals of 6 test cases are provided to the three domain experts for evaluation. These sample test cases are not trained cases but they were selected from the FSCE

for testing the performance of a prototype. The reason for selecting only 6 test cases among a large is due to the time constraint.

Using confusion matrix makes easy to evaluate the performance with recall and precision. A confusion matrix is a visualization tool typically used to present the retrieved case results attained by a machine learner. As shown in Table 4.2, the columns of retrieved and none retrieved in the matrix represent the instances of cases in a predicted class, while relevant and non-relevant rows represent the instances of cases in an actual class.

As seen earlier, recall is the ability of a retrieval system to obtain all or most of the relevant cases in the case base. It represents the proportion of the relevant retrieved cases to all the relevant cases in the case base. It represents the ability to retrieve relevant cases, the more relevant cases retrieved the better recall will be. Accordingly, the average recall performance of the prototype system is 71% as shown in Table 4.2.

Precision on the other hand, is the fraction of a retrieved result that is relevant for a particular query. It represents the proportion of the relevant retrieved cases to all the retrieved cases, the more relevant cases retrieved and the less irrelevant cases retrieved the better precision will be. Its calculation requires knowledge of the relevant and non-relevant cases in the six evaluated set of cases. Thus, it is possible to calculate the precision of each test case at [1, 0.8) threshold boundary in the retrieved cases ranked as top case **n** to bottom case **m** ranked order of the prototype system. Therefore, the average precision (accuracy) of the system performance is 86% as shown in Table 4.2.

This testing is mainly focused on the retrieval task. The reuse task is the adaptation of the retrieved cases for later revise task. The revise task is performed by the user to retain for later use. And modifying the retain task by the user is impossible. Therefore, testing these last 3 consecutive tasks of CBR is not possible to test in terms of precision and recall like the retrieval CBR task.

Test query		Retrieved case	Total No. of relevant cases	Recall	Precision
1: ch'imari demwoz	Relevant cases	3	5	0.6	0.75
	Total no. of retrieved cases	4			
2: yesra wul mequaret'	Relevant cases	2	2	1.0	1.0
	Total no. of retrieved cases	2			
3: yalagbab mesenabet	Relevant cases	8	9	0.89	0.89
	Total no. of retrieved cases	9			
4: wuzf demwoz	Relevant cases	3	4	0.75	0.75
	Total no. of retrieved cases	4			
5: t'qmat'qm lagny	Relevant cases	1	2	0.5	1.0
	Total no. of retrieved cases	1			
6: lyu lyu kfyawoch	Relevant cases	7	12	0.58	0.86
	Total no. of retrieved cases	8			
Total	Relevant cases	24	34	0.71	0.86
	Total no. of retrieved cases	28			
Average Performance				0.71	0.86

Table 4.2: Confusion matrix, Recall and Precision results to the query

4.5.3. User Acceptance Testing

Testing and evaluating the performance of LCMS acceptance by user judgments is valuable in addition to the developmental phase testing. Evaluation of the system is the process of judging the goodness of a proposed solution to achieve the set up objectives.

User acceptance testing is performed in a real situation at Addis Ababa University (AAU) for the system validity. Since lawyer at the FSCE are so busy to give time for evaluating the system, the researcher has selected 6 law experts from AAU Law department students (1 PhD, 3 LLM and 2 LLB) to test the applicability of the prototype with 6 sample cases (queries).

In this test, experts are requested to rank for the set up evaluation parameters. The researcher assigned values in numbers for each scale as excellent=5, very good=4, good=3, fair=2, and poor =1. Based on this given scale, system evaluators (selected experts) provide a value for each parameter with the selected queries. This testing method helps the researcher to examine the user satisfaction to the prototype system based on the response interpretations.

The user acceptance evaluation method is measured with manual computation using the following formula for average (performance).

$$Av = \sum_{i=1}^{S=5} \text{Svi} * \frac{nri}{Nr}$$

$$\text{AvP} = \text{Av} \times \frac{100}{\text{S}} = \text{Av} \times 20$$

Where, i is individual scale i.e. from 1 to 5, S is the total number of scale i.e. 5, Svi is an individual scale value, Nr total number of respondent, nri is number of respondent participated in each scale value, and AvP is average performance. The result is summarized in Table 4.3.

Evaluation parameters	**Parameter values with no. of respondents and Average Performance**						
	1= poor	2 = fair	3 = good	4 = very good	5 = excellent	Av	AvP (in %)
Attribute are relevant to represent legal case values.			1	3	2	4.2	84
The system is easy to use and understand when retrieving cases.			1	4	1	3.9	78
The system is applicable to legal case management in law domain.				2	4	4.7	94
The system is efficient in speed and time saving while executing.					6	5	100
The performance of the prototype system is used for more situations in law domain			1	4	1	3.9	78
Total	0	0	3	13	14	**4.3**	**86**
Average of parameter values (in %)	**0**	**0**	**10.0**	**43.3**	**46.7**	**86**	

Table 4.3: User Acceptance Testing

As indicated in Table 4.3, for the evaluation parameters “the relevance of attributes of the system”, 17%, 50% and 33% of the respondents rated the rank as good, very good and excellent respectively. The overall average performance for this parameter is 84%.

In the same way, for the parameters “easily use and understandability of the system to the user”, 17%, 66%, and 17% of the respondents rated the rank as good, very good and excellent respectively. The overall average performance for this parameter is 78%.

Similarly, the “system applicability to legal case management in law domain” 33% and 67% of the respondents rated the rank as very good and excellent respectively. The overall average performance for this parameter is 94%.

Likewise, for the criteria of the “system is efficient in speed and time” 100% of the respondents rated the rank as excellent. The overall average performance for this parameter is 100%.

Finally, for the criteria of ‘performance of the prototype system is used for more situations in law domain’ 17%, 66%, and 17% of the respondents rated the rank as good, very good and excellent respectively. The overall average performance for this parameter is 78%.

In general, based on the six system evaluator’ responses and feedback, the average performance obtained out of 5 number of scales is 4.3 which is the result obtained from the scale values assigned for each evaluation parameters. In addition, out of 30 possibilities of choosing each ranked order of the given evaluation parameter for each scale value; 3 times is as good (i.e. 10%), 13 times is as very good (i.e. 43.3%), and 14 times is as excellent (i.e. 46.7%). This result also indicates that 86% of users are satisfied by the performance of the prototype CBR system. This means the proposed CBR system is valid with 86% of user acceptance in the law domain.

4.6. Discussions of Results

The testing results discussed based on user acceptance and statistical analysis evaluation described in the above influence on the performance of the prototype system. Based on the evaluator feedback, there are no results about the system performance ranked as poor and fair but 8.3 % of good, 38.9% of very good, and 52.8 % of excellent judgments have been gained while checking the system performance by six evaluators as shown in Table 4.3. Moreover, the domain experts in the domain area were accepting the system validity with about total average performance result of 86%. This result shows that the prototype system is more applicable in terms of time saving and speed with quality services, case analysis, and importance of attribute selection, easy system usage, understandably, attractiveness and its applicability in the domain area of law.

To sum up, the performance of the prototype CBR system in this situation has been tested in statistical analysis and user acceptance evaluation techniques. A statistical analysis evaluation is performed in terms of recall and precision statistical measurements. As shown in Table 4.2, the average system performance results evaluating using recall and precision is 71% and 86% respectively. In addition to this, user acceptance testing was followed. So, 86% of the users or evaluators have accepted the prototype system in the domain of labor law. Therefore, the CBR system could help lawyers to provide better decision and quality services in case management activities which is a major problem in the current situations.

By considering the above performance results of LCMS, it is important to compare with previous CBR System done by Ethiopia (2002) in the same area as indicated in the Table 4.4.

Title and researcher	Used tool	Scope of CBR tasks	Performance measurements and results (in %)		
			Recall	Precision	User acceptance
Legal Precedence Retrieval, Ethiopia (2002)	CBR-Works	Only Retrieval	97.5	47.8	Not specified
Legal Case Management	Jcolibri	4 “Res”	71	86	86

Table 4.4: Comparison of LCMS with the previous CBR System work

As shown in Table 4.4 above, Ethiopia (2002) achieved a higher interesting performance in recall (97.5%) and lower precision performance (47.8%) in comparison with this study. In terms of precision, this work has a better improvement than a previous CBR system done by Ethiopia. The higher the threshold interval results the higher precision and the lower the recall and vice versa. Because of the number of retrieved cases becomes fewer when the threshold similarity increases and the precision becomes higher while computing.

The result difference could be due to the increment of cases by 11 and attributes by 2 as the number of cases increase the system performance also increase. In addition to these, the difference might be due to threshold interval value i.e. when a higher threshold interval similarity is used then retrieving most similar cases from the case base is possible. This could make a cause for lower average recall and higher average precision of the prototype system performance. Although the performance of this research is good, there is a need of improvements for gaining a better performance.

CHAPTER FIVE

CONCLUSIONS AND RECOMMENDATIONS

In this chapter, concluding remarks based on findings of this research and recommendations are provided in the following sections.

5.1. Conclusions

CBR has been a hot major research focused area within the AI and KBS in any specific domain area applications. Since the past few years, the application of AI in the sub field of CBR for case or precedent analysis in the domain of law has attracted many researchers. Thus, the main focus of this research is exploring the potential applications of CBR in legal case management with Ethiopian labor law cases using jcolibri framework.

The labor legal cases were represented in an attribute-value vector to build a plain text file for jcolibri case structure and connector in CBR. The represented plain text file was indexed or mapped with the appropriate selected attributes using the jcolibri framework. The four CBR tasks were then configured using the mapped case structure and connector to build a case base for managing the cases using four "Re's of CBR tasks in general. Therefore, the study was conducted for developing a prototype CBR system for legal case management that could help the domain experts (lawyers) in managing legal cases in terms of: retrieving similar cases to the query from the case base, adapting the retrieved case for use, revising the adapted solutions and retaining the modified case for future use to solve the current problem.

The reason for investigating the application of CBR in legal case management is since searching relevant cases manually is very tiresome and time consuming for providing quality services to the customers. This is a major problem to the FSCE in the present time. As a result, this prototype system could show the possibility to manage previously solved cases using CBR tasks to solve a problem. So, 50 relevant labor cases were selected with their case types to manage them in the appropriate way to provide quality services at the FSCE.

The four "Re's" of CBR applications of tasks might be accessed by a user or lawyer in LCMS. By utilizing these four "Re's" of CBR applications (Retrieval, Reuse, Revise and Retain), a

prototype CBR system might solve a problem in CBR with jcolibri framework. This application is done by using the first CBR task retrieval of cases with entering a new problem description (case) by using the GUI query window to retrieve most similar cases in the case base. After the retrieval of these similar cases, reusing the previously solved cases from the case base is performed automatically; case revision is followed by manual revision of cases to solve the problem at hand and save it by the domain experts-lawyers; and the fourth final task is retaining (learning) that used to store and learn the revised case in the case base for future use. This means that any case can be added by the lawyers in the revise task from the existing case base

The relevance of CBR systems of legal case management in the context of a labor law jurisdiction depends on many considerations. Judges or lawyers in labor law jurisdictions could make decisions with the retrieved legal cases using the case base. Lawyers or judges increasingly could need LCMS to compare current problems to past decided cases for purposes of drawing legal decisions. In this way, the CBR system of legal case management offers techniques for improving upon the ability of full-case LCMS to process retrieved cases in an intelligent manner that reflects their significance in legal arguments in incremental learning process.

While testing the performance of the LCMS by the user acceptance testing, 86% of the evaluators have accepted the prototype system to apply it in the domain of labor law. And in statistical analysis, 71% of average recall and 86% of average precision has been registered. The user acceptance and precision evaluation results were better improved while recall was decreasing when compared to the previous work. This is the reason that more relevant and less irrelevant number of cases was retrieved during the retrieval process. Even this is a prominent result of the prototype system performance; improvement of this prototype system is also important for further investigations.

The major contribution of this prototype study is to come up the CBR applications into considerations in the field of legal labor law cases management for the possible development of a prototype to new real legal expert system that could provide fast decisions making and enhance the **service delivery improvements** in law domain. The service improvements could be achieved in terms of simplicity in using LCMS, quality services, improvement of traditional

technology (searching cases manually) and ease of reducing layers' work load at the FSCE and also could be distributed to other local courts all over the nation.

In addition to this, the result of this study could contribute to the design of intelligent legal case management systems and contributing to Cognitive Science models of the use in various related fields (civil and criminal law) of case comparison methods to transform ill-structured problems of cases into better structured of case manipulations or management.
Moreover, the result based on the user feedback, this study could applicable in the domain area to provide quality services and enhance efficient tasks in terms of time and speed.

However, the following major critical **challenges** that need for further investigations were faced while doing this study. The collected Amharic legal texts were not represented with the plain text representation technique that is required to be loaded by the jcolibri connector. This was a major challenge to the researcher in this study.

The general knowledge explanation facility to advise the user when similarity previous solved cases are not found in the case base is not achieved in this study. This is because of elicitations of tacit knowledge from domain experts and representing it as general knowledge discussing with domain experts was another challenge.

There are three major law types of cases in the court. But, due to the time constraints and cases accessibility, only labor law is selected as a specific study domain area which is the major hot issue in the current situations. This is also the other challenge in addition to the above stated challenges.

5.2. Recommendations for Future Work

The objectives of this research have been achieved with constraints and challenges. So the researcher has forwarded the following recommendations for future investigations.

- AI and law are tied hand in hand in the knowledge society to develop legal case management expert systems. AI is used to develop the intelligence systems with almost all natural language processing applications. However, cases in the domain of law in our country are documented in Amharic language writing systems. To use these cases as they are the CBR developmental tool jcolibri was not compatible with the language. Here the researcher was obligated to transliterate the cases in to the machine readable format for compatibility purpose. This could reduce the quality of the prototype CBR application. Therefore, it is important to use the local Amharic language **Computational Linguistics** (or natural language processing-**NLP**) application. This Computational Linguistics could include grammar checkers and natural language interface to convert legal texts into legal cases automatically to solve a problem with CBR applications for a better performance of a prototype system.

- Rule-based and case-based reasoning are two complementary popular approaches used in intelligent systems in the field of AI and/or KBS. Rules usually represent general knowledge explanation facility, whereas cases encompass knowledge accumulated from specific (specialized) situations to incremental learning and specific knowledge acquisition. So, LCMS has no general knowledge explanation facility in this research as it focused on the specific domain expert (knowledge). Therefore, integrating these interesting RBR system and CBR system of KBS methods to produce effective hybrid applications is important for developing a prototype to real CBR system to improve the performance of LCMS over the individual methods.

- In the enormous development of industrial organizations in this era, different labor disputes (ስራ ክርክር), are arising. Thus, the labor cases are increasing from time to time in the FSCE and difficult to manage them manually. This is a reason that the researcher has

identified it as a major problem area in this study. However, there are other disputes such as: criminal dispute (ወንጀል) and civil disputes (ፍትሐብሄር) established in the FSCE. Because CBR is important in legal practice of all these kinds of disputes in law domain, not just for legal case management applications in Labor Law jurisdictions. Therefore, developing a prototype to a real CBR system for these three major types of disputes in an "**all in one**" application is better by representing and structuring each case within a case base to help lawyers in managing all cases for a better way.

- 50 legal cases have been used in this study. But, if the number of cases increased in a case base, then performance of the system will also increase. Therefore, it is recommended that extension of this work by adding more cases in the case base to increase the performance of the prototype system using **ontology** based CBR application other than plain text file. Because CBR ontology could support case-based comparisons to find relevant cases, compare them with the problem, draw inferences based on the comparisons, and make arguments how to decide the problem.

References

- Aamodt, A., & Plaza, E. (1994). Case-Based Reasoning: Foundational Issues, Methodological Variations, and System Approaches. AICom-Artificial Intelligence Communications. 7(1), 39-59.
- Ashley, K. (1987). Modeling Legal Argument: Reasoning with Cases and Hypotheticals. A Ph.D. Dissertation, Massachusetts University, Department of Computer and Information Science, Massachusetts-Amherst.
- Ashley, K. (1991). Reasoning with cases and Hypotheticals in HYPO. International Journal of Man-Machine Studies, 34(6), 753-796.
- Ashley, K. (2002). An AI Model of Case-Based Legal Argument from a Jurisprudential Viewpoint, Artificial Intelligence and Law, 10 (1-3), 163-218.
- Ashley, K. (2003). Case-Based Models of Legal Reasoning in a Civil Law Context. In Invited paper. International Congress of Comparative Cultures and Legal Systems of the Instituto de Investigaciones Jurídicas, Universidad Nacional Autonoma de México, Mexico City.
- Ashley, K., & Rissland, E. (2003). Law, Learning and Representation. Artificial Intelligence, 150(1), 17-58. Retrieved at: www.elsevier.com/locate/artint/Retrieved date on February 10, 2014.
- Azeb B. (2009). Integrated Case Based and Rule Based Reasoning for Decision Support. A PhD dissertation at Norwegian University of Science and Technology Department of Computer and Information Science, Norway.
- Baye, Y. **(**1992). Ethiopian Writing System. Department of Linguistics, Addis Ababa University, Addis Ababa, Ethiopia, website: http://www.ethiopians.com/bayeyima.html /retrieved date on March 05, 2014.
- Bello, J., Gonz´alez-Calero, P., & D´iaz-Agudo, B. (2004). Jcolibri: An Object-Oriented Framework for Building CBR Systems. In ECCBR, 32–46.
- Cabrerizo, F., P´erez, I, & Herrera-Viedma, E. (2009). Managing the Consensus in Group Decision Making in an Unbalanced Fuzzy Linguistic Context with Incomplete Information. Journal of Knowledge-Based Systems. 23(5): 169–181.

- Dhaliwal, J., & Benbasat, I. (1996). The Use and Effects of Knowledge Based System Explanations: Theoretical Foundations and a Framework for Empirical Evaluation. Information Systems Research 7(3), pp. 342-362.
- Dreyfus, H., & Dreyfus, S. (1986). Mind over Machine: the Power of Human Intuition and Expertise in the Era of the Computer, Free Press, New York.
- Durkin, J. (1994). Expert Systems: Design and Development. Prentice-Hall Inc., New York, 502.
- Ethiopia, T. (2002). Application of Case-based Reasoning for Amharic Legal Precedent Retrieval: A Case Study with the Ethiopian Labor Law, MSc Thesis, School of Information Science of Addis Ababa University, Addis Ababa, Ethiopia**.**
- Federal Negarit Gazeta of the Federal democratic Republic of Ethiopia, Proclamation No. 377/2003. Addis Ababa 36th February 2004, 10th year No. 12.
- Ferruccio M., Nicola P., & Paolo C., (2010). Tools and Methods Based on Knowledge Elicitation to Support Engineering Design, Retrieved from the open archive university website: http://openarchive.univpm.it/jspui/bitstream/123456789/341/1/Tesi.Cicconi.pdf/ /retrieved date March 10, 2014.
- Iglezakis, I., Reinartz, T., & Roth-Berghofer, T. (2004). Maintenance memories: beyond concepts and techniques for case base maintenance. In Proceedings of the Seventh European Conference on Case-Based Reasoning, 227-241. Springer Berlin Heidelberg.
- Iqbal, N., & Ashraf, M. (2006). Evaluation of jCOLIBRI: Case Based Reasoning Framework. Jurnal Teknik ITS, 1(1), 351-356.
- Kolodner, J. (1992). An Introduction to Case-Based Reasoning. Artificial Intelligence Review, 6(1), 3-34.
- Kolodner, J. (1995). Case-Based Reasoning. 1993. Morgan Kaofmann Publishers, San Mateo, CA.
- Laudon, K., & Laudon, J. (1997). Management Information Systems, Organization and Technology, (5th edt.). Prentice Hall, Upper Saddle River, New Jersey.
- Lucas, P., & Van Der Gaag, L. (1991). Principles of Expert Systems. Wokingham: Addison-Wesley, Netherlands.
- Pal, S., & Shiu, S. (2004). Foundation of Soft Cased-Based Reasoning. Wilely Series on Intelligent Systems, John Wiley & Sons, Inc., Hoboken, New Jersey.

- Prentzas, J., & Hatzilygeroudis, I. (2007). Categorizing approaches combining rule-based and case-based reasoning. Expert Systems, 24(2), 97-122.
- Recio-García, J., Díaz-Agudo, B., & González-Calero, P. (2008). jCOLIBRI2 Tutorial. Group for Artificial Intelligence Applications, Department of Software Engineering and Artificial Intelligence, University Complutense of Madrid, Madrid.
- Reinartz, T., Iglezakis, I. & Roth-Berghofer, T. (2001). Review and Restore for Case-Based Maintenance. Computational Intelligence, 17(2), 214-234.
- Roth-Berghofer, T. (2003). Knowledge Maintenance of Case-Based Reasoning Systems: the SIAM Methodology, 262, IOS Press, Amsterdam.
- Russell, S., & Norvig, P. (1995). Artificial Intelligence: A Modern Approach, Prentice-Hall, Inc, Upper Saddle River, New Jersey.
- Russell, S., & Norvig, P. (2003). Artificial Intelligence: A Modern Approach (2^{nd} ed.), Prenice-Hall, Upper Saddle River, New Jersey.
- Sasikumar, M., Ramani, S., Raman, S., Anjaneyulu, K., & Chandrasekar, R. (2007). A Practical Introduction to Rule Based Expert Systems. Narosa Publishing House, New Delhi.
- Schulz, S. (1999). CBR-Works- a State-of-the-art Shell for Case-Based Application Building. In Proceedings of the 7th German Workshop on Case-Based Reasoning, GWCBR, 99, pp. 166-175.
- Stahl, A., & Roth-Berghofer, T. (2008). Rapid Prototyping of CBR Applications with the Open Source Tool myCBR. In Advances in Case-Based Reasoning, 615-629.
- Watson, I. & Marir, F. (1994). Case-Based Reasoning: A Review. The Knowledge Engineering Review, 9(4), 327-354.
- Weber, R. (1998). Intelligent Jurisprudence Research, PhD dissertation, Federal University of Santa, Catharina.
- Wiig, K. (1994). Knowledge Management. The Central Management Focus for Intelligent-Acting Organizations, Schema Press Ltd., Arlington.
- Winograd, T. (1990). Thinking Machines: Can There be? Are we?, in: D. Partridge, Y. Wilks (Eds.), The Foundations of Artificial Intelligence, Cambridge University Press, Cambridge, 167-189.

Appendices

Appendix I: Sample Plain Text Cases

CaseID,Danyoch,Yeksu Aynet,Kesash ,Tet'eri/Tekesash,Awaj Qut'r,Anqets' Qut'r ,Ksu Yetekefetebet Qen,Ksu Yetezegabet Qen,Wusane

1,1. Ato Menberetsihay Tadesse 2. Ato FishaWorkneh 3. Ato Abdulkadir Mohamod 4. W/o Sindu Alemu 5. Ato Mesfin Ekubeyonas,yeseratenya qnesa,yeKK brd lbs seratenyoch mahber,yeKK ch'erqach'erq industri ,42/85,138/1 ena 147,T'r 5 qen 1997, Hamle 29 qen 1997,1. lezih krkr menesha yehonwu yeseratenya qnesa yewol sra krkr bemehonu gudayun yemedanyet sltan yebord enj yefrd bet lihon aygebam bemalt wsnual 2. ygbanyu yeqerebelet yefederal keftenya frd betm yhnnu tegenzbo wede krkru bemegbat ena bebordu yetesetew frd kehg akuaya shtet yalebet mehon alemehonun bememermer gudayu kememeria gar wede bord limeles yemigebawu mehon alemehonu reged wusane lisetbet ygebal bemalet tewesnual 3. yefederal keftenya frd bet krkrun mayet endiqetil yaschlew zendm gudayu beFtabher SneSrat qutir 343/1/ meseret temelsoletal

2,1. Ato Menberetsihay Tadesse 2. Ato Fisiha Workineh 3. W/o Sindu Alemu 4. Ato Asegd Gashaw 5.W/o Desta Gebru,yalagbab kesra mesenabet, yegimbi ketema astedader tsihfet bet,W/ro Merertu Fekadu,42/85, 2/1 ena 3/2,T'r 25 qen 1996,T'qmt 1 qen 1998,yegimbi wereda frd bet yemrab welega keftenya frd bet yeaoromiya t'qlay frd bet ena bemeleya qut'r 1235 yekllu t'qlay frd bet seber semi chilot gudayun beawaj qut'r 42/85/85 meseret aytew yeset'ut wusane beftabher snesrat hg qut'r 348/1/ meseret teshrual.

3,1. Ato Menberetsihay Tadesse 2. Ato Fisiha Workineh 3. W/o Sindu Alemu 4. Ato Asegd Gashaw 5.W/o Desta Gebru,Yedemwoz ch'imari,Ato Teshome Jifar,yeityopia telekomunikeshn korpoeshn,42/85,147, Miaziya 13 qen 1996,T'qmt 1 qen 1998,yeaseri ena seratenya guday wosany bord bemeleya qutr 14/01/93 yefederal keftenya frd bet degmo bemelya qutr 21501 yesetut wusane tsentual

4,1. Ato Menberetsehay Tadesse 2. Ato Fisha Werkneh 3. W/ro Sindu Alemu 4. W/ro Desta Gebru 5. Ato Asegd Gashaw,t'qmat'qm,yeityopia telecomunikeshn corporeshn, Ato Genta Gema,377/96,9;10;138 ena 142, Hamle 2 qen 1996, T'qmt 22 qen 1998,"1. yeaserina seratenya guday wesany bord miyazya 21/96 amete mhret bequt'r 303/02/94 yeset'ewu wusanena yefederal keftenya frd bet bemeleya qut'r 30835 sene16/96 amete mhret yeset'ewu wusane beftabher snesrat qut'r 348/1/ meseretbedmts' blch'a teshrual. 2. wusanewochachew yetesharu mehonachewun endiawqut yefrduna yewusanewu qj ytelaleflachew. 3. wechna kisara gra qenyu beyerasachew ychachalu mezgebu tezegtual wede mezgeb bet mezgeb bet."

5,1. Ato Menberetsehay Tadesse 2. Ato Fisha Werkneh 3. W/roSindu Alemu 4. W/ro Hosaena Negash 5. Ato Asegd Gashaw,yesra wul mequaret',yeityopia telecomunikeshn ,W/rt Tigst Werku,42/85,9;10, Gnbot 25 qen 1995,Hdar 15 qen 1998,"1. beamelkachena betet'ri mekakelyetederegew yesra wulyetequaret'ebet mknyat hgawi enj hfe wet' balemehonu amelkach wed sra ltmeles aygebam bemalet tewesnual. 2. yh chilot bemeleya qut'r 17189 legudayu agbabnet lalachew yeawaj qut'r 42/85/85 dngagewoch trguame bemest't hgu sra lalteserabet gize wuzf demewez endikefel yemifeqd aydelem yemil medemdemia lay bemedresu tet'eri lalserachbet gize demewez likefelat aygebam bemalet tewesnual. 3. bemehonum tet'eri yeamet fekad endiset'at yetewesenewun yesr frd betoch wusane saych'emr qeriwochu befederal yemejemeria frd bet bemeleya qut'r 224/90?befederal keftenya frd bet degmo bemeleya qut'r857/93 yeteset'ut yewusane kfloch beftabher hg snesrat qut'r348/1/ meseret teshrewal."

Appendix II: Amharic Alphabet

	ā/ä [a]	u [u]	ī/i [i]	a [a]	ē/e [e/ɛ]	(i)/(ə) [ə]	o [o/ɔ]
h [h]	ሀ ha	ሁ hu	ሂ hi	ሃ ha	ሄ he	ህ h(ə)	ሆ ho
l [l]	ለ le	ሉ lu	ሊ li	ላ la	ሌ le	ል l(ə)	ሎ lo
ḥ/ḥ [h]	ሐ ha	ሑ hu	ሒ hi	ሓ ha	ሔ he	ሕ h(ə)	ሖ ho
m [m]	መ me	ሙ mu	ሚ mi	ማ ma	ሜ me	ም m(ə)	ሞ mo
s/ś [s]	ሠ se	ሡ su	ሢ si	ሣ sa	ሤ se	ሥ s(ə)	ሦ so
r [r]	ረ re	ሩ ru	ሪ ri	ራ ra	ሬ re	ር r(ə)	ሮ ro
s [s]	ሰ se	ሱ su	ሲ si	ሳ sa	ሴ se	ስ s(ə)	ሶ so
sh/š [ʃ]	ሸ ʃe	ሹ ʃu	ሺ ʃi	ሻ ʃa	ሼ ʃe	ሽ ʃ(ə)	ሾ ʃo
k'/q [k']	ቀ k'e	ቁ k'u	ቂ k'i	ቃ k'a	ቄ k'e	ቅ k'(ə)	ቆ k'o
qh [ʁ']	ቐ ʁ'e	ቑ ʁ'u	ቒ ʁ'i	ቓ ʁ'a	ቔ ʁ'e	ቕ ʁ'(ə)	ቖ ʁ'o
b [b]	በ be	ቡ bu	ቢ bi	ባ ba	ቤ be	ብ b(ə)	ቦ bo
t [t]	ተ te	ቱ tu	ቲ ti	ታ ta	ቴ te	ት t(ə)	ቶ to
ch/č [ʧ]	ቸ ʧe	ቹ ʧu	ቺ ʧi	ቻ ʧa	ቼ ʧe	ች ʧ(ə)	ቾ ʧo
h/ḫ [h]	ኀ ha	ኁ hu	ኂ hi	ኃ ha	ኄ he	ኅ h(ə)	ኆ ho
n [n]	ነ ne	ኑ nu	ኒ ni	ና na	ኔ ne	ን n(ə)	ኖ no
ny/ñ [ɲ]	ኘ ɲe	ኙ ɲu	ኚ ɲi	ኛ ɲa	ኜ ɲe	ኝ ɲ(ə)	ኞ ɲo
ʾ/ʾ [ʔ]	አ (ʔ)a	ኡ (ʔ)u	ኢ (ʔ)i	ኣ (ʔ)a	ኤ (ʔ)e	እ (ʔ)(ə)	ኦ (ʔ)o
k [k]	ከ ke	ኩ ku	ኪ ki	ካ ka	ኬ ke	ክ k(ə)	ኮ ko

	ā/ä [a]	u [u]	ī/i [i]	a [a]	ē/e [e/ɛ]	(i)/(ə) [ə]	o [o/ɔ]
h/ḵ [h]	ኸ he	ኹ hu	ኺ hi	ኻ ha	ኼ he	ኽ h(ə)	ኾ ho
w [w]	ወ we	ዉ wu	ዊ wi	ዋ wa	ዌ we	ው w(ə)	ዎ wo
ʿ/ʿ [ʔ]	ዐ ʔa	ዑ ʔu	ዒ ʔi	ዓ ʔa	ዔ ʔe	ዕ ʔ(ə)	ዖ ʔo
z [z]	ዘ ze	ዙ zu	ዚ zi	ዛ za	ዜ ze	ዝ z(ə)	ዞ zo
zh/ž [ʒ]	ዠ ʒe	ዡ ʒu	ዢ ʒi	ዣ ʒa	ዤ ʒe	ዥ ʒ(ə)	ዦ ʒo
y [j]	የ je	ዩ ju	ዪ ji	ያ ja	ዬ je	ይ j(ə)	ዮ jo
d [d]	ደ de	ዱ du	ዲ di	ዳ da	ዴ de	ድ d(ə)	ዶ do
j/ǧ [ʤ]	ጀ ʤe	ጁ ʤu	ጂ ʤi	ጃ ʤa	ጄ ʤe	ጅ ʤ(ə)	ጆ ʤo
g [g]	ገ ge	ጉ gu	ጊ gi	ጋ ga	ጌ ge	ግ g(ə)	ጎ go
t'/ṭ [t']	ጠ t'e	ጡ t'u	ጢ t'i	ጣ t'a	ጤ t'e	ጥ t'(ə)	ጦ t'o
ch'/č̣ [ʧ']	ጨ ʧ'e	ጩ ʧ'u	ጪ ʧ'i	ጫ ʧ'a	ጬ ʧ'e	ጭ ʧ'(ə)	ጮ ʧ'o
p'/ṗ [p']	ጰ p'e	ጱ p'u	ጲ p'i	ጳ p'a	ጴ p'e	ጵ p'(ə)	ጶ p'o
ts'/ṣ [ts]	ጸ ts'e	ጹ ts'u	ጺ ts'i	ጻ ts'a	ጼ ts'e	ጽ ts'(ə)	ጾ ts'o
ts'/ṣ́ [ts]	ፀ ts'e	ፁ ts'u	ፂ ts'i	ፃ ts'a	ፄ ts'e	ፅ ts'(ə)	ፆ ts'o
f [f]	ፈ fe	ፉ fu	ፊ fi	ፋ fa	ፌ fe	ፍ f(ə)	ፎ fo
p [p]	ፐ pe	ፑ pu	ፒ pi	ፓ pa	ፔ pe	ፕ p(ə)	ፖ po
v [v]	ቨ ve	ቩ vu	ቪ vi	ቫ va	ቬ ve	ቭ v(ə)	ቮ vo

Appendix III: Glossary

Application: An automated system or part of an automated system, which provides the end user with access to and processing of stored data. An application may be a manual process, which has been automated.

AI (Artificial Intelligence): AI is the area of science which focuses on creating systems that can engage on behaviors that humans consider intelligent. Additionally, intelligence is the computational part of the ability to achieve goals in the world.

Case- base: denotes a knowledge base that stores the previously solved knowledge or cases like DB or corpus. Knowledge base contains all necessary knowledge about the domain that is required to handle problems.

Case: it is a knowledge that usually denotes and refers to a specific experience or knowledge tied to specific problem situation that is worth remembering for future use.

CBR (Case-based reasoning): is in effect a cyclic and integrated process of solving a problem, learning from experience, solving a new problem, etc to use previous similar cases to solve, evaluate or interpret a current new problem..

jcolibri: "Java Cases and Ontology Libraries Integration for Building Reasoning Infrastructures" is an object-oriented framework in Java for building Case-Based Reasoning (CBR) systems. It includes mechanisms to Retrieve, Reuse, Revise and Retain cases and is designed to be easily extended with new components.

KBS (Knowledge Based Systems): A system that uses stored knowledge to solve problems in a specific domain. KBS is a program for extending and/or querying a knowledge base. The Computer User High-Tech Dictionary defines a knowledge-based system as a computer system that is programmed to imitate human problem-solving by means of artificial intelligence and reference to a DB of knowledge on a particular subject. KBS are systems based on the methods and techniques of AI. Their core components are the knowledge base and the inference mechanisms.

KNN (k-nearest neighbors): is a algorithm and method for machine learning approach a for retrieving cases based on closest similarity to the old cases stored in the case base in the feature space. It is a type of instance-based learning, or lazy learning where the function is only approximated locally and all computation is deferred.

Learning: is a very important feature of case-based reasoning so called as case-based learning that can be both machine (the system) and human (the lawyers) learning.

Ontology: In computer science and information science, ontology is a formal representation of a set of concepts within a domain and the relationships between those concepts. It is used to reason about the properties of that domain, and may be used to define the domain.

RBR (Rule Based Reasoning): A particular type of reasoning which uses "if-then-else" rule statements for general knowledge explanations.

XML (Extensible Markup Language): is a method for putting structured data (such as that in a worksheet) in a text file that follows standard guidelines and can be read by a variety of applications.

Column: A vertical field in an attribute Table or a vertical group of cells in a grid or pixels in a plain text.

Row: A record, instance, or occurrence of attributes in a Table or a horizontal group of cells in a gird or pixels in a plain text.

Declaration

I declare that this thesis is my work and has not been attempted to copy any other sources without citations and duly acknowledged.

_____

Abebaw Alem

June, 2014

This thesis has been submitted for examination with my approval as a university advisor.

_____

Martha Yifru (PhD)

June, 2014