KNOWLEDGE ACQUISITION

Knowledge acquisition is the process of acquiring, organizing, and studying knowledge. Two main types of sources of knowledge:

- Documented (which can take many forms) and
- Undocumented (usually in the expert's mind).

There are two types of knowledge: shallow knowledge and deep knowledge.

Shallow knowledge: An expert has a base understanding of the subject, some of which could be described as general. For example a pension's expert will know that a pension is a source of income after retirement. Now, obviously a great many non-experts know this. It is, however, central to the understanding of the pension's domain. Other general knowledge could be the general strategy that the expert employs when solving a problem, the general class of problem that the expert is tackling and some typical problem solutions. This is sometimes referred to as meta-knowledge.
Example: surface level information might be represented as: If the weather is bad then stay in bed.

Deep Knowledge: This is the knowledge that has been acquired by years of experience and study and is the detailed `core` of the knowledge base. This knowledge covers such items as:

- Relationships between data
- Rules/concepts
- How important each item is
- What order he/she performs tasks
- The detailed strategies for arriving at solutions

Example: if we take the weather example again, we may ask: for instance what do we classify as bad weather, why does bad weather cause us not to want to go out, what's so good about staying in bed etc. Frames and semantic networks enable us to represent deeper knowledge.

The Knowledge Acquisition Process

i). **Identification: -** This is a stage in the problem is identified, its characteristics are examined and it is broken down into smaller sub-problems. The participants are identified (experts, users, knowledge engineers). The resources are identified or provided. The situation is studied.
ii). **Conceptualization: -**The best method of representing knowledge is identified and selected.
iii). **Formalization**: - The knowledge is extracted from the sources. Software and hardware issues are resolved.
iv). **Implementation: -** The program is written or a prototype is developed.
v). **Testing: -** Test cases are used. The experts are consulted on acceptability of results.

Knowledge Elicitation: - It implies that knowledge acquisition is accomplished from a human expert.

Knowledge Acquisition Methods

This constitute tools used for the process of modeling knowledge. There are two basic strategies in knowledge engineering. Either, *starting from general and overall concepts, gradually leading the expert to elicit details of a topic* or *starting from the details of specific cases and helping the expert establish and derive general concepts from the specific examples.* Therefore, the methods of eliciting knowledge can be categorized into two broad groups:

i). Top-down (or deductive) methods
ii). Bottom-up (or inductive) methods

i). **Top-Down Methods**

The knowledge engineer organizes the acquisition sessions for discovering general concepts, rules, and objects, and then gradually goes into the details of each concept, rule, or object.
This can be grouped into four categories: questioning methods, object-oriented methods, quantitative methods and inventive methods

1) **Questioning Methods:**
The knowledge engineer interviews the expert in a series of meetings, or asks the expert to fill out a questionnaire. There are three strategies used in questioning the expert

i). Structured interviews
- Designed to be systematic and goal-oriented.
- It forces a directed discussion on a topic; therefore it can reduce interpretation issues and may also reduce the expert's subjectivity.

ii). Unstructured interviews
- Informal: Good to use when beginning the KA process.
- They help the KE understand the domain and can help identify discussion points for structured interviews.

iii). Questionnaires
- Used in cases where access to the expert is limited,
- A questionnaire is prepared for the expert to answer in writing.
- This method is useful for clarifying already developed topics in the advanced stages of knowledge engineering.

2) **Object - oriented Methods.**
The KE focuses the interview sessions on discovering the objects within the domain. Usually done by asking the expert to group the actual objects in the field in order to form a class of objects that has a common set of attributes.

3) **Quantitative methods.**
The methods were developed in *cognitive science and decision analysis for eliciting the degree of a decision maker's preferences and utilities*, and in grouping various objects and attributes
It is used to measure and determine:
- The extent of relationships among objects (or concepts)
- The degree of uncertainty about the domain knowledge

4) **Inventive methods.**
Expert is allowed a more active part in the process in one of the following roles:

Expert as a teacher:
The expert is responsible for teaching and transferring expertise to the knowledge engineer (KE) and is given the responsibility for the preparation and organization of the elicitation sessions. It is efficient at the early stages of knowledge acquisition

Expert as a partner in systematic innovation:
This is an abstract concept and requires the expert and the KE to identify pieces of knowledge that are in *contradiction, and to discover solution methods for removing the contradiction*
E.g. the applicant's request for a loan and a less than favorable credit rating form a contradiction. The expert must provide a solution for this contradiction, which leads to the elicitation of new pieces of knowledge. The expert and knowledge engineer form a partnership in discovering contradictions and creating solutions; this taps the expert's deep understanding of the domain.

Expert as the knowledge engineer
The expert may have both technical interest in the system and the needed training in knowledge engineering i.e. the expert plays the role of a knowledge engineer.

ii). **Bottom-Up Methods**
The knowledge engineer focuses the expert's attention on specific cases. This helps the expert abstract the decision for resolving a specific case to a more generalized rule or concept.
The methods that can be grouped in this approach are: example-based methods, protocol analysis and observation.

1) **Example-based methods**

The example-based approach *constitutes the foundation of case-based learning and learning by analogy*. In this method, the knowledge engineer and the expert work on a number of representative cases or examples in one of the following ways:

a) **Grouping examples: -** The expert groups the examples based on their similarities and differences. This process helps to determine categories of examples and the development of general rules for each category.

b) **Walk-through method: -** The knowledge engineer selects a number of cases previously decided by the expert, and asks the expert to walk through the decision process.

c) **Quantitative analysis of examples: -** The quantitative techniques are tools for helping the expert discover the relations among various attributes of the decision cases. It is categorized into two groups:

i). Statistical methods
- The examples must be a random sample of the cases decided by the expert(s).
- The data on the examples are fed into a statistical technique, such as regression analysis, in order to discover the expert's decision criteria.

ii). Inductive methods
- The example set contains a representative set of all possible cases the expert has encountered.
- The examples are fed into the inductive method, which produces a decision tree or a set of decision rules.

2) **Protocol analysis**

The expert is asked to *think aloud and verbalize his or her thought process while solving a set of actual (or simulated) problems and making decisions*. The KE records the process, and later analyzes the large volume of information produced from this method to discover the general rules the expert uses in solving problems.

3) **Observation**

This involves observation of the expert while solving a problem. It is useful when the solution of the problem is procedural and takes place in a sequence of steps through time.
The absence of the biases and intrusion inherent in the KE's questions makes this approach useful. The KE must make sure that the expert is making decisions in the most realistic environment.

Computer Aided Knowledge Acquisition

Most Knowledge Acquisition (KA) techniques are VERY time consuming. The experts are often inarticulate. The knowledge engineers are inexperienced and/or not sufficiently conversant with the domain. It is suggested that computers could be used effectively in this process and this should:
- Increase productivity
- De-skill the KA process
- Eliminate the need for an expert

The use of computers for KA is rapidly expanding and this section considers a technique called rule induction and mentions some of the tools available.

Automated Rule Induction

This technique takes a set of examples to try and generate general rules.

Advantages of rule induction
- If the knowledge is very complex and/or the domain is large then it may not be possible to develop expert systems. If there is a 'bank' of old cases then rule induction may allow development of a system.
- The developer of a system does not have to have all the skills of a knowledge engineer.
- This approach may well allow for new knowledge to be acquired.
- Once a base set of rules have been developed using rule induction these can be modified by the knowledge engineer and expert together.

Disadvantages of rule induction

- Challenges on how are the attributes chosen
- Algorithms are inefficient.
- Only suitable for rule based systems
- Challenges on how to choose the training set
- The algorithms cannot usually deal with exceptions

Automatic Knowledge Acquisition Tools

i). Auto-Intelligence - A PC based tool that allows interactive KA with some rule induction

ii). The Knacq - A methodology for KA 'by exception' that provides a tool for entering the knowledge and automatic code generation for various shells.

Knowledge Acquisition Modes

Presently, the knowledge elicitation modes could be divided into three categories: manual mode, automated mode and combined manual and automated mode

i) Manual Mode

This mode requires a direct interaction between the knowledge engineer and expert.

ii) Automated Mode

A number of software products exist in the market that are designed for automatic elicitation of knowledge. The automated mode may be divided into two categories:

- Automated mode with the expert: - The expert interacts with the computer. The software is designed to help the expert elicit knowledge without the assistance of a knowledge engineer.
- Automated mode without the expert: - The system is fed examples to produce rules or decision trees.

iii) Combined Manual and Automated Mode

- At the start of the process, the KE could use manual-based methods for conceptualization, modularization, and gaining insights into the knowledge domain.
- In some of the modules for which enough examples are available and the accessibility to the expert is limited, the knowledge engineer can use the automated mode to accelerate the elicitation of the knowledge.
- Later, the expert should review the results, and may make suggestions to modify the automatically developed rules.

Issues with Knowledge Acquisition

- Hard to get experts to express how they solve problems
- Representation on machine requires detailed expression i.e. at a very low level. Must be represented in a structured way.
- Bringing together the ideas of all those involved in the knowledge transfer process. Many participants are involved and they have varied backgrounds causing communication challenges (Expert, System designers, Users, etc.).
- Mismatch between the way experts hold their knowledge and the way computers represent knowledge.

Various attempts have been made at overcoming these difficulties. This forms an important and active research area. One approach is the use of natural language interfaces so that experts can communicate directly. Computer aided knowledge acquisition tools are an alternative automated solution.

VALIDATION AND VERIFICATION

These are terms used quality of a knowledge base. The methods are collectively known as *evaluation techniques.*

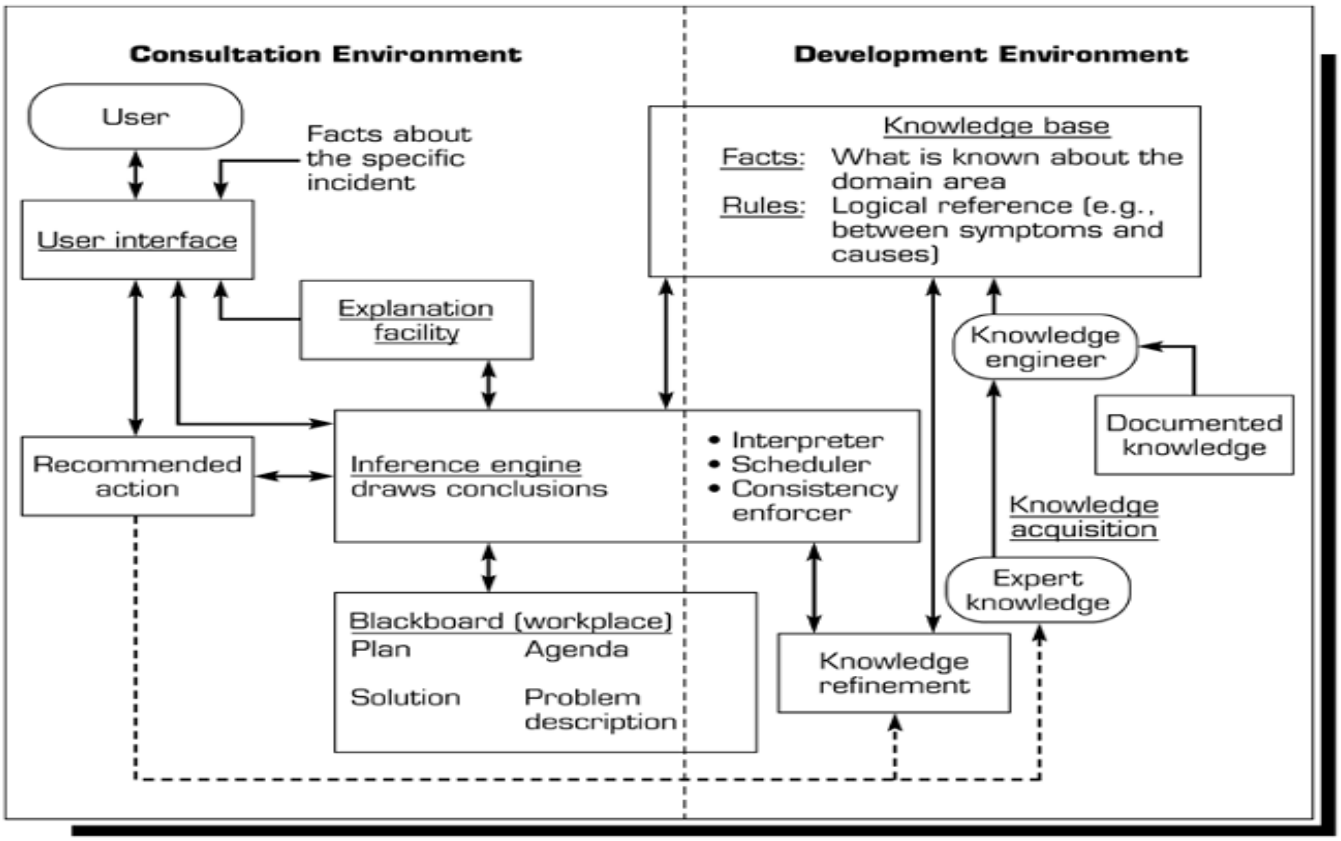
i). Validation: This deals with the performance of the system i.e. how well it is making conclusions. The common method of evaluation is the TURING TEST.

ii). Verification: This deals with the building of the system to the correct specification.

EXPERT SYSTEMS

Expert system is a set of computer programs that mimic the human expert. The programs will take and solve problems in restricted problem domains or areas of expertise. Expert systems must therefore have knowledge similar to the ones held by human experts and use it to solve problems of the type solved by human beings.

STRUCTURE OF AN EXPERT SYSTEM



Knowledge base

The knowledge base contains the know-how of the human experts in a particular field. Such know-how is of two types:-

- Facts: also known as "deep knowledge". These are based on generalized learning from School and books and are well known and widely accepted by experts in the field.
- Heuristics: known as "surface knowledge". Heuristics consist of "rules of thumb", i.e. knowledge attained through experience.

Knowledge base is derived and implemented from knowledge analysis and representation. It is the result of knowledge acquisition and representation. For example, if you are using CLIPS to develop a rule-based production system, the knowledge base will be presented in the form of production rules.

Inference engine is the brain of the expert system and, which has two primary tasks:

- Inference: the inference engine employs reasoning to examine existing facts and rules. Updates stored knowledge and draws conclusions.
- Control: the inference engine's mechanism for controlling the search of the knowledge base. A time consuming task if the knowledge base is large and complex.

The parts of the inference engine consist of:

i). *Interpreter* (rule interpreter in many systems) that executes selected agenda items by applying relevant rules from the knowledge base.

ii). *Schedule* that maintains control over the agenda. It uses priorities and other criteria to estimate the effect of applying rules on the agenda.

iii). *Consistency enforcer* maintains a consistent representation of the emerging solution.

Working memory (Blackboard) is an area of the Computer's RAM reserved for storing information regarding the current status of the problem. The blackboard may contain a plan of how the problem should be solved, an agenda of potential actions awaiting execution and a solutions indicating candidate hypotheses and alternate courses of action.

External data source: Some expert systems receive input data from external sources other than the user. Common forms of this data - known as sensor data - include X- rays, audible sounds and visual images. The system interprets the data and makes inferences based on that data. E.g. a medical system used to monitor a patient's blood pressure. If the blood pressure reaches a certain critical level the system will automatically take some action, perhaps sending an alarm signal to the user (the nurse).

User interfaces the part that enables the users or builders to submit their items to the expert system and also the expert system to respond to the users or builders.

Explanation subsystem (justifier) is the part that traces responsibility for conclusions to their sources. It may explain:

- Why some questions are asked by the expert system;
- How some conclusions are reached;
- Why some alternatives are rejected;
- The plan used to reach the solution;
- The remaining facts to establish before the final conclusion.

Knowledge refining system is the part that enables the analysis and use of knowledge so that learning may take place and improvements may be made.

Participants in Expert Systems

The people involved in developing an expert system are: - The Knowledge Engineer(s), expert(s), user(s), programmers and management.

The people are involved in various stages of the development process and interact heavily. It should be noted here that one person can fulfil more than one role. For instance, an expert might also build the expert system and, indeed, is a member of the Management team but this is not usually an ideal situation.

i). **The Knowledge Engineer:** The Knowledge Engineer is the person who actually develops expert systems. This person requires skills and expertise that are peculiar to the world of expert systems. Obviously a knowledge engineer needs to have a good technical grasp of expert systems software. The job of the knowledge engineer involves very high interaction with people - the experts, users and management. This means that he or she requires very good communication skills. The ability to talk at the right level using tact and diplomacy is a very rare skill. It can be compared to a systems analyst/programmer role with extra requirements of communication that extend beyond a systems analyst.

ii). **The Expert:** The expert (or experts) is the person who provides most of the knowledge that is to be encapsulated in the expert system. He or she will be the final arbiter on the correctness of the finished system. The expert has experience or knowledge about a particular subject area (domain) and the expert is chiefly responsible for providing domain knowledge.
The ideal expert is one who is: - articulate, enthusiastic and computer - literate
Computer literacy is ideal but not necessary as if he/she has an understanding of what computers can and cannot do it will greatly enhance the discussions.

iii). **The User:** The User (or, more usually, users) is the person who is actually going to use the finished system. If the expert system is a consultative or advisory system he/she will interact heavily with the finished system. If the system is not primarily consultative (e.g. plant control) the user will play less of a role in development but will still need to be consulted at various stages. Problems occur when individuals in a group of users have different skills and capabilities and care should be taken to ensure that the target user is well defined.

iv). **Management:** Any expert system development that is to be successful has to have full support of management at all levels. A good deal of ignorance and misinformation regarding their capabilities can lead to wariness amongst management to commit scarce resources to an expert systems project. Therefore, management should be aware of expert systems their potential benefits and realistic costs.

Classic Problem Areas Addressed by Expert Systems

Expert systems have been used in several typical problem areas. Several application systems have been demonstrated in these areas.

Category	**Problem area addressed**
Interpretation	Infer situation descriptions from observations. e.g. surveillance, image analysis, signal interpretation (e.g. PROSPECTOR)
Prediction	Infer consequences of given situations. e.g. weather forecasting, traffic predictions, demographics (e.g. PLANT)
Diagnosis	Infer malfunctions from observations e.g. medical, mechanical, electronic, software diagnosis (e.g. MYCIN)
Design	Configure objects under constraints e.g. circuit layouts, building design, plant layout (e.g. R1/XCON)
Planning	Develop plans to achieve goals e.g. project management, routing, communications, financial plans (e.g. MOLGEN)
Monitoring	Compare observations and plans, flag exceptions e.g. air traffic control, fiscal management tasks (e.g. NAVEX)
Debugging	Prescribe remedies to malfunctions e.g. mechanical and software
Repair	Execute plans to administer prescribed remedy. Generally they identify weaknesses in knowledge and appropriate remedies
Instruction	Diagnose, debug, and correct student performance Generally, they identify weaknesses in knowledge and offer appropriate remedies (e.g. SOPHIE)
Control	Interpret, predict, repair and monitor system behavior e.g. life support, artificial environment (e.g. VENTILATOR MANAGEMENT ASSISTANT)

Benefits of Expert Systems

i) *Increased productivity and output* since expert systems work faster than humans.
ii) *Decreased decision-making* time as expert systems can make decisions faster.
iii) *Increased process and product quality* as errors can be significantly reduced.
iv) *Capture of scarce resource* as ES can store the expertise held by humans who may be taking long to train.
v) *Flexibility* as ES can sense changing needs and advice accordingly, such as a product out of production.
vi) *Easier equipment* operation where ES is used to operate a complex equipment.
vii) *Operation in harzadous environments* where ES is used where humans are not safe such as in nuclear power plants, toxic environments.
viii) *Accessibility to knowledge and help desks* where ES is used to supply information and scarce knowledge or support help desks.
ix) *Increased capability* of other computerized system where ES integrate with other systems to make applications work faster or produce higher quality results.
x) *Ability to work with incomplete or uncertain information* where ES may use existing knowledge to solve a problem even though some other facts may still be missing or uncertain.
xi) *Provide training where ES* is used by novices because of its contents in the knowledge base.
xii) *Enhanced problem solving and decision making* where ES integrates analysis and judgement of top experts.
xiii) *Improved decision quality* as ES is reliable and does not become bored, tired or hold attitudes.
xiv) *Knowledge transfer to remote locations* as ES can hold specialist knowledge that may be used in remote locations such as eye disease diagnosis and treatment system demonstrated in Algeria and Egypt by World Health Organization.

Limitations of Expert Systems
The following problems are associated with expert systems:

- Knowledge is not always readily available;
- Difficulty in extracting expertise from humans;
- Variations in problem assessment by different experts;
- Human experts cannot abstract when under pressure;
- Only work well in a narrow domain;
- Experts may not always validate their conclusions;
- Experts may not always use understood vocabulary;
- Knowledge engineers are few and expensive;
- End-users may not trust expert system;
- Subjectivity and biases in knowledge transfer.

Success Factors for Expert Systems
Implementation of expert systems may succeed because of the following factors:

- Management support and user involvement;
- There must be a high level of knowledge;
- There must be at least one cooperative expert;
- The problem must be qualitative (fuzzy);
- The problem must be sufficiently narrow in scope;
- There should be a good ES shell (store & manipulate knowledge naturally);
- The should be a friendly user interface for novice users;
- The problems must be difficult enough to warrant the use of ES;
- There should be competent knowledge-based system developers;
- The ES should positively impact on end-users.

BUILDING EXPERT SYSTEMS
The software that is used for constructing expert systems range from programs used for building expert systems to programs that can aid the knowledge acquisition process. The main software tools for developing expert system fall into the following categories.

- **Development software**-programming languages (AI languages (Prolog and Lisp) and general purpose languages); shells (e.g. JESS, KAPPA PC, CLIPS) and AI toolkits
- **Development support tools**- I/O facilities; debugging aids; explanation facilities; Editors
- **Systems building tools**-e.g. rule induction engines

Example
Write an expert system to diagnose the problem of a car that won't start.

Solution

```
%Problem: Write an Expert system tha can help diagnise
%for a problem of a car that won't start
%Diagnostic session: System probes for possible causes
%(Asks,notes response and give Recommendation)

car_wont_start:-
            problem_with_battery;
            problem_with_cables;
            problem_with_starter;
            problem_with_switch;
            problem_with_fuel.
car_wont_start:-nl,nl,
 write('I am sorry, I can''t diagnose the car problems ,maybe you never answered the
question correctly'),nl,nl.

problem_with_battery:-nl,
                        write('Is the battery voltage ok(Yes/No)'),
                        read(Battery),Battery=no,nl,nl,
                        write('Recharge battery and turn the KEY'),nl.
```

```
problem_with_cables:-nl,
                        write('Are the cables faulty(Yes/No)'),
                        read(Cables),Cables=yes,nl,nl,
                        write('Replace cables and turn the KEY'),nl.

problem_with_starter:-nl,
                        write('Is you starter motor ok(Yes/No)'),
                        read(Starter),Starter=no,nl,nl,
                        write('Do you want to repair or replace the motor?'),nl,nl,
                        write('============================'),nl,nl,
                        write('1:     Replace'),nl,
                        write('2:     Repair'),nl,
                        write('Enter your Choice:'),
                        read(Choice),starter(Choice).

starter(Choice):-Choice=:=1,nl,write('Replace the motor with new one'),nl.
starter(Choice):-Choice=:=2,nl,write('Repair the starter and find out if it works'),nl.

problem_with_switch:-nl,
                        write('Is your ignition switch ok(Yes/No)'),
                        read(Switch),Switch=no,nl,nl,
                        write('Replace the switch and turn the KEY'),nl.

problem_with_fuel:-nl,
                        write('Is your petrol tank empty(Yes/No)'),
                        read(Fuel),Fuel=yes,nl,nl,
                        write('Buy petrol and turn the KEY'),nl.
```

EXPERT SYSTEM SHELLS

An expert system shell is a tool that has been specifically designed to enable speedy development of expert systems. Building expert systems by using shells offers significant advantages. A system can be built to perform a unique task by entering into a shell all the necessary knowledge about a task domain. If the program is not very complicated and if an expert has had some training in the use of a shell, the expert can enter the knowledge himself. The following are key features of expert systems shells:

- The knowledge base and inference mechanism are totally separate.
- Shells offer a high level easy to use language for encoding the knowledge.
- The knowledge engineer does not have to write the inference engine since this is supplied with the tool.
- The development of a sophisticated user interface is usually facilitated by providing good screen designers.
- Most modern shells have tools such as editors, file handling facilities, debugging facilities and so on.

The User Interface

Since the user interface of expert systems is so important most shells offer facilities for developing an effective user interface.
Facilities are usually available for developing good explanation and help facilities 'How' and 'Why' facilities are usually built into the shell

Examples: There are literally hundreds of expert system shells available. Some examples are: CLIPS, JESS, Mycin, Babylon, G2, etc

CASE-BASED REASONING (CBR)

CBR depends on domain-specific knowledge to solve problems. However, instead of depending on general rules, it mainly depends on concrete cases, which are records of typical problems and their solutions collected in the past. When a new problem shows up, it is compared with the solved problems. After the most similar case is located, the corresponding solution is adapted to the new problem. Generally it is solves a new problem by by making an analogy to an old one and adapting its solution to the current situation. It requires storing, retrieving and adapting past solutions to similar problems. Application example is legal reasoning.
The crucial issue in CBR is how to represent and index a case, how to measure the similarity between cases, how to adapt a solution to a new situation, and so on.
Retrieving a case starts with a problem description and ends when a best matching case has been found

All case-based reasoning methods have in common the following process (5 Rs):

- identifying a set of relevant problem descriptors (representation)
- retrieve the most similar case (or cases) comparing the case to the library of past cases(retrieve)
- reuse the retrieved case to try to solve the current problem(reuse)
- revise and adapt the proposed solution if necessary(revise)
- retain the final solution as part of a new case(retain)

Case: This consist of information about the situation, the solution, the results of using that solution and key attributes that can be used for quickly searching for similar patterns of attributes. There are three major parts in any case:

i). A description of the problem/situation:- the state of the world when the case is available
ii). Solution:- the chain of operators that were used to solve the problem (solving path)
iii). Outcome/consequence: - the state of the world after the supervention of the case (description of the effect on the world)

In addition to specific cases, one also has to consider the case memory organization

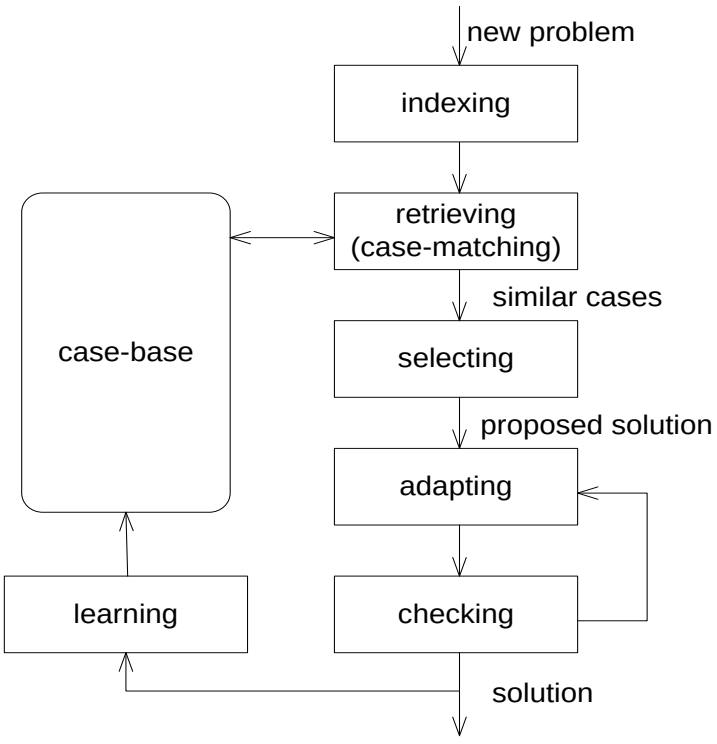
Case – indexing: The most important problem in CBR is how do we remember when to retrieve what? Essentially, the indexing problem requires assigning labels to cases to designate the situations in which they are likely to be useful. The following are issues concerning indexing of cases :-

- indexing should anticipate the vocabulary a retriever might use
- indexing has to be by concepts normally used to describe the items being indexed
- indexing has to anticipate the circumstances in which a retriever is likely to want to retrieve something

MAIN COMPONENTS OF CASE-BASED SYSTEMS

- Elements in a case-based reasoning system are: case base - set of cases, and index library – used to efficiently search and quickly retrieve cases that are most appropriate or similar to the current problem.
- Similarity metrics - used to measure how similar the current problem is to the past cases selected by searching the index library
- The adaption module - creates a solution for the current problem by either modifying the solution (structural adaptation) or creating a new solution using the same process as was used in the similar past case (derivational adaptation).
- Learning: If no reasonably appropriate prior case is found then the current case and its human created solution can be added to the case base thus allowing the system to learn.

STRUCTURE OF CBR



Suitable domain:

i). A large volume of historical data already exists.
ii). Experts talk about their domain by giving examples.
iii). Experience is as valuable as textbook knowledge.
iv). Problems are not fully understood (weak models, little domain knowledge available).
v). There are a lot of exceptions to rules.
vi). There is a need to build a corporate memory and transfer expertise among personnel.

Advantages of CBR:

i). Case-base is more objective and formal than the expert's interpretation (knowledge of expert's)
ii). Knowledge are represented in an explicit way
iii). Case can be defined for incomplete or badly-defined notions
iv). CBR is suitable for domains for which a proper, theoretical foundations do not exist
v). CBR is applicable in default of algorithmic method
vi). Easy knowledge acquisition (get well during usage)

Disadvantages of CBR:

i). CBR solves only the problems covered by cases
ii). CBR might use a past case blindly without validating it in the new situation
iii). Solution is time-demanding (also in case of proper indexing)

Comparison of Rule-based systems and Case-based systems

Rule-based systems	**Case-based systems**
Rule: symbolic pattern	Case: collection of data, constants
Rule: individual unit, independent of the other rules, consistent piece of field of interest	Case: depends on the other cases (often overlap each other), individual unit of the field of interest
Retrieving rule: exact matching	Retrieving case: partial matching
Using of rules: general iterative cycle	Using of cases: several steps (approximate retrieval, adaptation, refinement)
The model of the problem have to be developed (sometimes it is hard or impossible)	The model of the problem needn't be developed
The knowledge-acquisition of field of interest is hard and time-demanding	The knowledge-acquisition of field of interest is limited to collecting and analysing the past cases
Development time is long	Development time is short
Slow, handling of many data is difficult	Many data is treatable with the using of database –handling techniques
Enlargement is hard (the validation have to be repeated after enlargement)	Enlargement and development is easy.
Learning is not supported	It is able to learn (preserving new cases)