Analysis of Rule Engine

Deepali Joshi
M.E Computer
Mumbai

ABSTRACT
The technological advancement is rapid and dynamic. In a small amount of time there are various changes to be made. With change in time there are different trends. In order to help people new techniques, methods are being developed day by day. In an organization or different fields there are various rules that should be followed. The rules are also called as constraints. The constraints can be classified as hard and soft rules/constraints. Some of the rules are very critical and should be followed strictly and these type of rules are called as hard, while there are some rules which are not to be followed or satisfied strictly and they specify the working of an organization are called as soft constraints.

In order to achieve this a method that can be utilised and is known as Rule Engine. The term Rule Engine is quite ambiguous in that it can be any system that uses rules, in any form that can be applied to data to produce outcomes.

Keywords
Rule Engine, Inference Engine, Knowledge Representation Model

1. INTRODUCTION
The rules and regulations are very essential for any organization or institute for proper functioning. In some situations the rule may vary or certain modifications are required. The changes may be small or have to be applied for small system then it can be done easily. If the changes are to be applied frequently and the system is large then it becomes a tedious process. The Rule Engine is a technique which allows to change the rule or modify it completely. It consists of various components and with these components the rule implementation can be done. Every component has specific function which should be carried out. When all components function smoothly then the expected output can be achieved. The basic component of rule engine is Inference or Execution engine, Rule base, Knowledge engineering. Apart from this the rule engine can also be viewed in high level. The high level rule engine consists of components which are a bit different then the rule engine as they have some additional functions.

The main objective is to study and demonstrate the working of Rule Engine. There are number of techniques, methods, software's which are available to the users due to which the activities or task they perform will completed easily. As the time goes by some changes or updates have to be made. Similarly the rules might change and the new rules should be applied immediately. If the rules are changing once or twice the changes can be easily done, but if the changes are to be done frequently then it becomes difficult to update the rules since it becomes tedious process.

In order to solve the problem the Rule Engine is utilized. The main objective of rule engine is that it helps to modify the rules. The new rules are specified in a component called Knowledge Base. The change will take place only in knowledge base and then these changes are applied to the data. So due to this the tedious task of making changes to the entire system, the change in rule will only be reflected in knowledge base.

Rules are very essential for smooth working of an organization or institute. In short rules are essential in all fields and sectors and phases. There are some areas where rule remain the same till the end, while in some areas the rules are changing quite repeatedly. Whenever the rules change they have to be incorporated immediately so that the proper functioning will be done. If the rules are changing frequently then modifications should be done accordingly.

A method is available through which frequently changing rules can be implemented and is known as Rule Engine. The rule engine is made up of different components and these components implement the rules. There are various areas where rules are changing, some of the areas can be Medical, business, Academics. In the rule engine only one component is responsible for keeping track or storing the rules. As the rule change the changes are to be done only in that component and not the entire system.

2. RULE ENGINE
Rule engine can be defined as software component of a rule-based system that assesses individual rules and determines their applicability in a specific case or situation. At the most basic level, a rules engine is comprised of two components, a knowledge-base (consisting of rules represented in a computational format), and an inference or execution engine which can reason about incoming data based upon the contents of the knowledge-base in order to generate some form of output such as an instruction set or alert. An additional critical component of a rules engine is the knowledge engineering facility, which provides the capability to curate the contents of the knowledge-base on an automated, semi-automated, or manual basis. Basic rules engine architecture is given below.

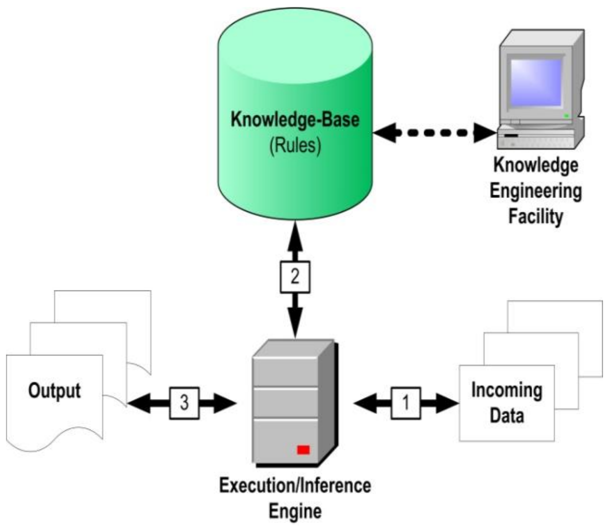

Fig 1 Rule Engine Architecture

In the architecture (see Figure 1), incoming data (1) is reasoned upon by the execution or inference engine using the rules encoded in the knowledge-base (2), in order to generate some form of output (3), such as instructions or alerts. An addition critical component of this architectural model is the knowledge engineering facility, which allows for the automated, semi-automated, or manual cu ration of the contents of the knowledge-base.

3. INFERENCE OR EXECUTION ENGINE

The inference or execution engine is the rules engine component that actually acts upon specific incoming data via the application of rules embodied in the knowledge-base, in order to generate some type of output. At the most basic level, the inference or execution engine can be defined as a software component that takes as its input one or more elements of data, applies a set of rules to that data, and generates some form of output. For the most part, reports on this type of rules engine component have described implementations which are specific to the type of knowledge representation model being employed.

Types of Execution Engine are:

1. Production rules
2. Logical or axiomatic rules
3. Hybrid rules

4. KNOWLEDGE REPRESENTATION MODEL

In order for the execution or inference engine described in section 3 to reason about incoming data per a set of pre-defined rules, it is necessary for those rules to be represented in a computationally tractable format. The format for such rules as used by a specific rules engine implementation can be referred to as the knowledge representation model. Examples of knowledge representation models commonly employed in the biomedical and computer science domains include context-specific Boolean rules such as those found in the Mycin clinical decision support system, complex axiomatic statements such as those employed by the Arden syntax for medical logic modules, and frames such as those employed in numerous biomedical ontologies and intelligent agents. The process of representing application-specific rules falls within the broader context of what is known as knowledge engineering (see Figure 2).

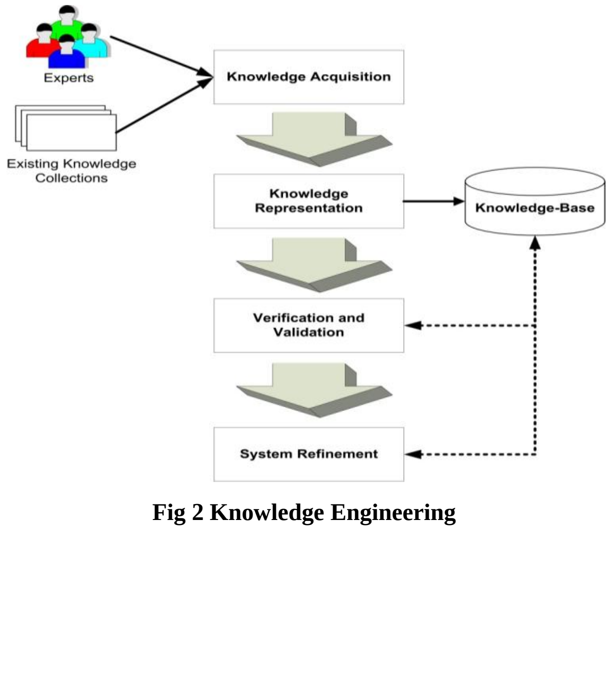

Fig 2 Knowledge Engineering

The KE process consists of various steps:

1) Acquisition of knowledge, to acquire knowledge.
2) Representation of that knowledge in a computable form.
3) Implementation of knowledge-based agents or applications using the knowledge collection generated in the preceding stages.
4) Verification and validation of the output of those knowledge-based agents or applications against one or more reference standards. These reference standards can include expert performance measures, requirements acquired before designing the knowledge-based system, or requirements that were realized upon implementation of the knowledge-based system. In this context, verification is the process of ensuring that the knowledge-based system meets the initial requirements of the potential end-user community. In comparison, validation is the process of ensuring that the knowledge-based system meets the realized requirements of the end-user community once a knowledge-based system has been implemented.

5. RULE ENGINE FEATURES

The rule engine is very essential for various fields. Through the rule engine it becomes simple to implement new rules and make changes in the existing rules. As the rule engine is used, if there are changes frequently they can be easily incorporated. Due to this advantage the rule engine is very essential and can be applied to different domains. Each and every component has specific function which it has to carry out and generate the required output.

The features of Rule Engine are:

- Declarative Programming

Rule engines allow you to say "What to do", not "How to do it". The advantage of this point is that using rules can make it easy to express solutions to difficult problems and consequently have those solutions verified. Rules are much easier to read than code. Rule systems are capable of solving very hard problems, providing an explanation of how the solution was arrived at and why each "decision" along the way was made.

- Centralization of Knowledge

  By using rules, you create a repository of knowledge which is executable. This means it's a single point of truth, for business policy, for instance. Ideally rules are so readable that they can also serve as documentation.

- Explanation Facility

  Rule systems effectively provide an "explanation facility" by being able to log the decisions made by the rule engine along with why the decisions were made.

- Tool Integration

  Tools such as Eclipse provide ways to edit and manage rules and get immediate feedback, validation and content assistance. Auditing and debugging tools are also available.

6. ARCHITECTURE OF HIGH LEVEL RULE ENGINE

The Rule Engine architecture can be of two type's basic and high level. The high level rule engine has different components like Production Memory, Agenda, and Pattern Matching. The Rules are stored in the Production Memory and the fact that the Inference Engine matches against are kept in the Working Memory. Facts are asserted into the Working Memory where they may then be modified or retracted. A system with a large number of rules and facts may result in many rules being true for the same fact assertion; these rules are said to be in conflict. The Agenda manages the execution order of these conflicting rules using a Conflict Resolution strategy.

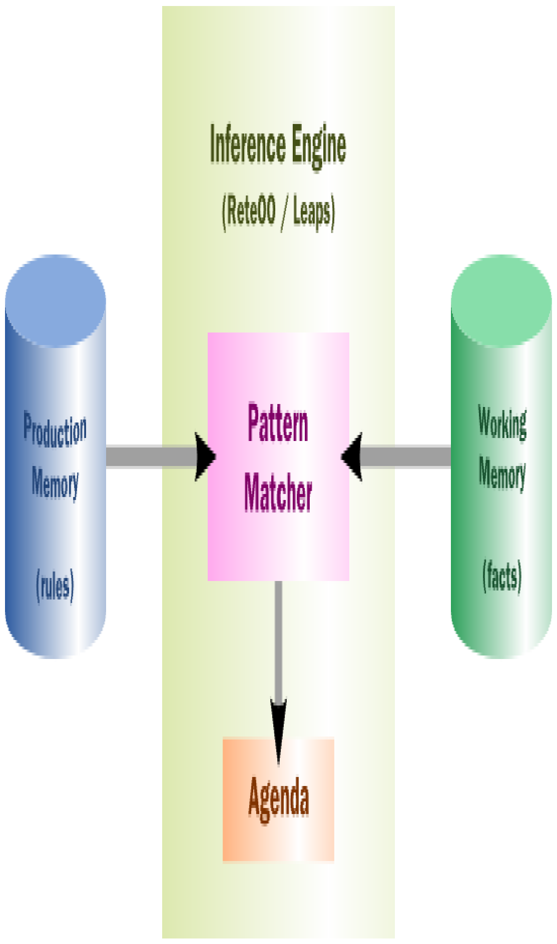

Fig 3 High Level Rule Engine

There are two methods of execution for a rule system: Forward chaining and backward Chaining; systems that implement both are called Hybrid Chaining Systems. Forward chaining is "data-driven" and thus reactionary, with facts being asserted into working memory, which results in one or more rules being concurrently true and scheduled for execution by the Agenda (see Figure 4).

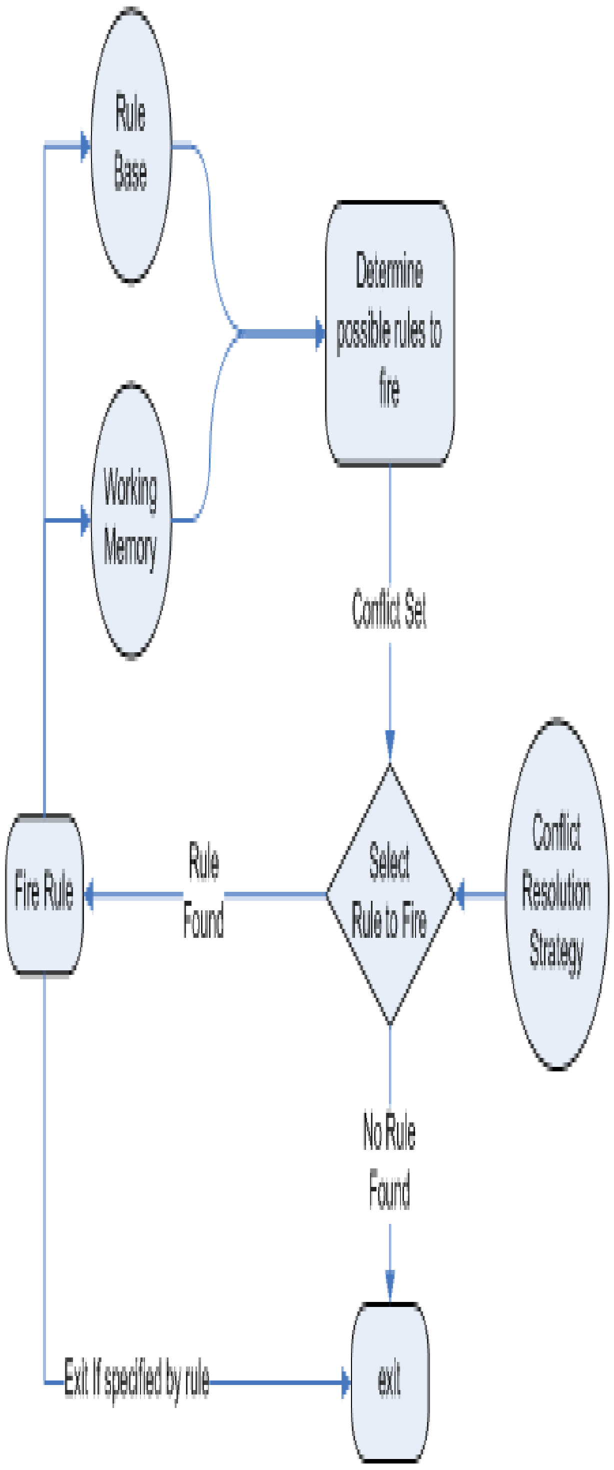

Fig 4 Forward Chaining

Backward chaining is "goal-driven", meaning that we start with a conclusion which the engine tries to satisfy (see Figure 5). If it can't it then searches for conclusions that it can satisfy; these are known as sub goals that will help satisfy some unknown part of the current goal. It continues this process until either the initial conclusion is proven or there are no more sub goals.

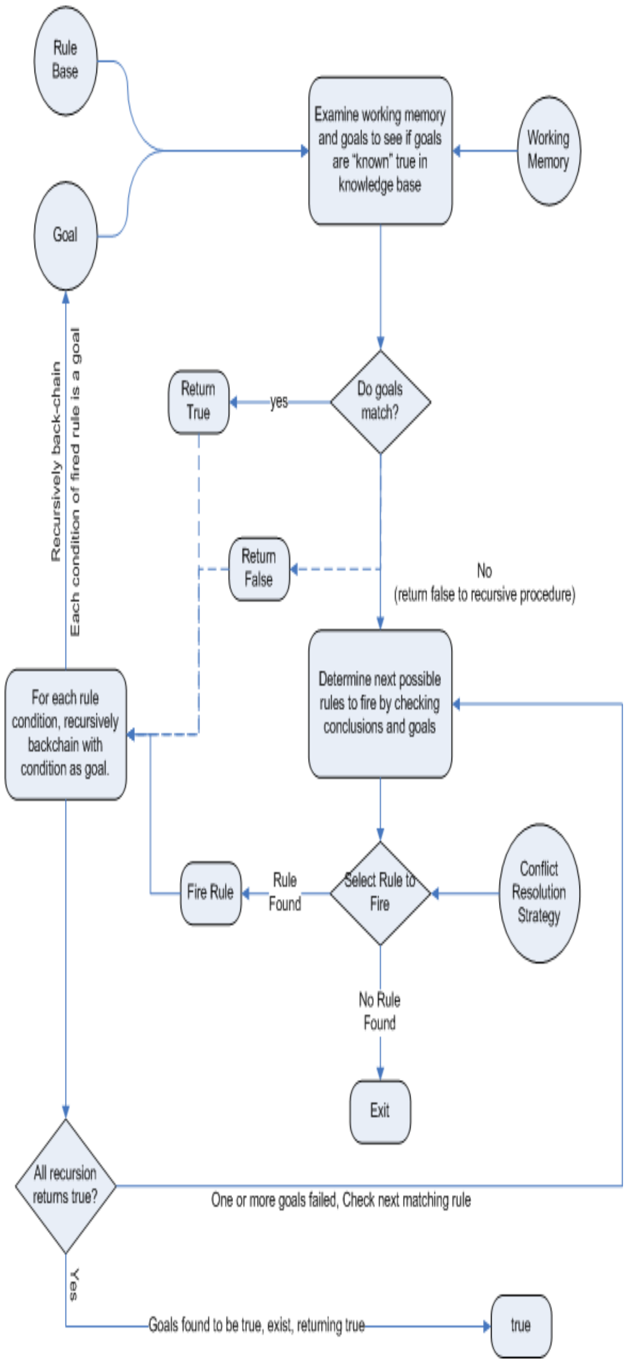
Rule Base
Goal
Examine working memory and goals to see if goals are "known" true in knowledge base
Working Memory
Do goals match?
yes
Return True
Return False
No
(return false to recursive procedure)
Recursively back-chain
Each condition of fired rule is a goal
Determine next possible rules to fire by checking conclusions and goals
For each rule condition, recursively backchain with condition as goal.
Fire Rule
Rule Found
Select Rule to Fire
Conflict Resolution Strategy
No Rule Found
Exit
All recursion returns true?
One or more goals failed, Check next matching rule
Yes
Goals found to be true, exist, returning true
true

Fig 5 Backward Chaining

7. RESULTS AND DISCUSSION

There are number of problems faced by the organization, in these situations the rule engine provides solutions to all the difficulties. Most of the organizations are now deploying the rule engine. The results of our research are: The rule engine has various advantages and features which help the organization to adapt to the changes in their environment. As and when there will be modification in the rules of organization they can be easily implemented via the rule engine.

8. CONCLUSION

The Rule Engine is very essential as it has various advantages and features also. Rule engines allow you to say "What to do", not "How to do it". The key advantage of this point is that using rules can make it easy to express solutions to difficult problems and consequently have those solutions verified. Rules are much easier to read than code. Rule systems are capable of solving very, very hard problems, providing an explanation of how the solution was arrived at and why each "decision" along the way was made. Rule engine is commonly used where It is a complex problem to solve, there are no obvious traditional solutions, or basically the problem isn't fully understood. The logic changes often. The logic itself may even be simple but the rules change quite often. In many organizations software releases are few and far between and pluggable rules can help provide the "agility" that is needed and expected in a reasonably safe way. Rules could be used embedded in your application or perhaps as a service. Often a rule engine works best as "state full" component, being an integral part of an application. However, there have been successful cases of creating reusable rule services which are stateless.

9. REFERENCES

[1] "Proposal of an Inference Engine Architecture for Business Rules and Processes."

[2] "A Rule-Based Inference Engine which is Optimal and VLSI Implementable."

[3] "Rules Engine Technologies Across caBIG^{tm} Workspaces."

[4] **"**Research of dynamic rule engine in financial management software", Bing Xu;Shi-yi, IEEE 2008

[5] Thong Chee Ling ; Yu Dian Gong. "A knowledge-based tool in facilitating course exemption process for institution of higher learning", , IEEE 2011

[6] Ziqin Yin, Xiaolin Li, Chaofan Wu, Gehao Lu. "Research of Rule Engine in web service environment", Springer