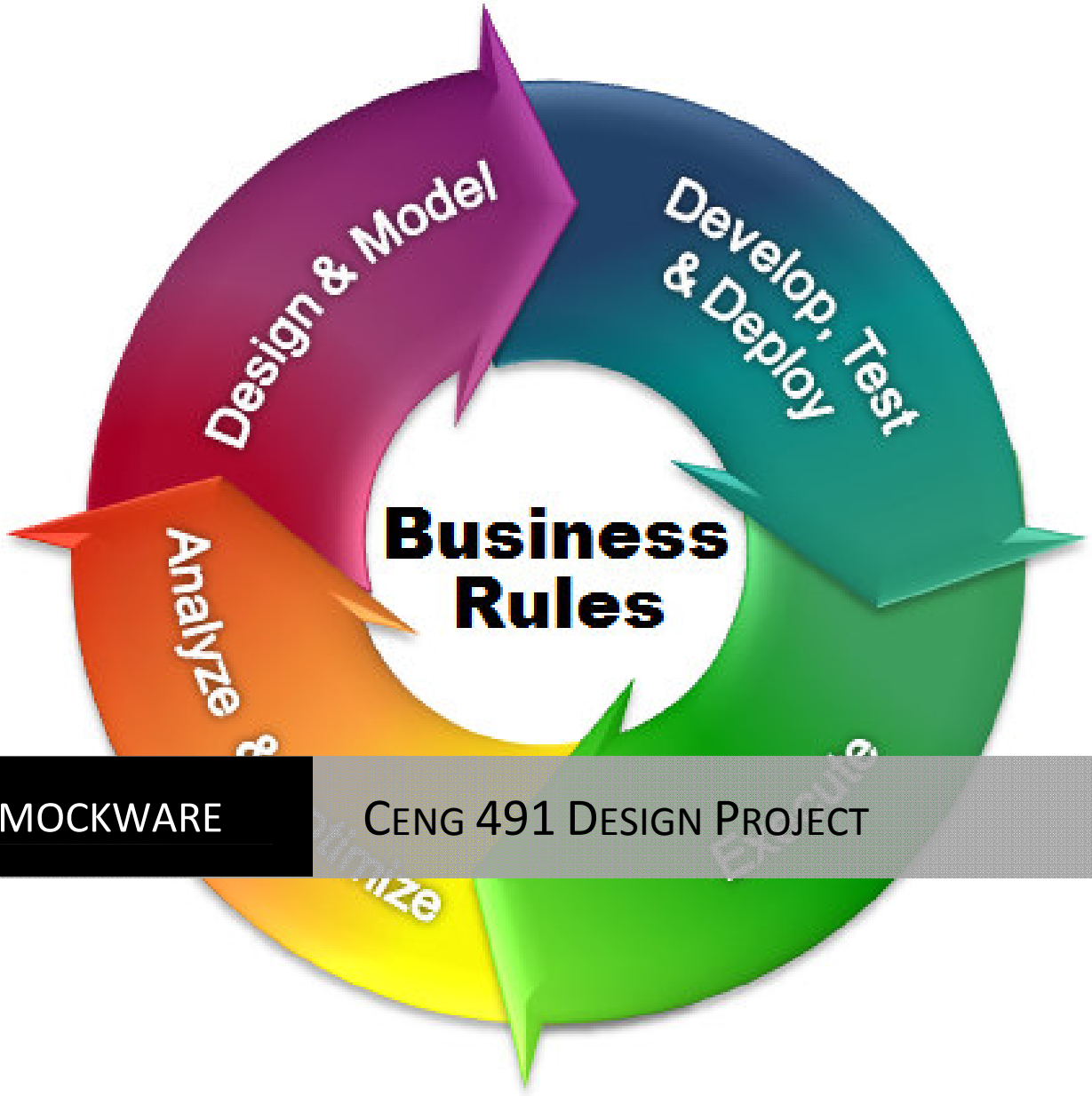


MOCKWARE

CENG 491 DESIGN PROJECT

Table of Contents

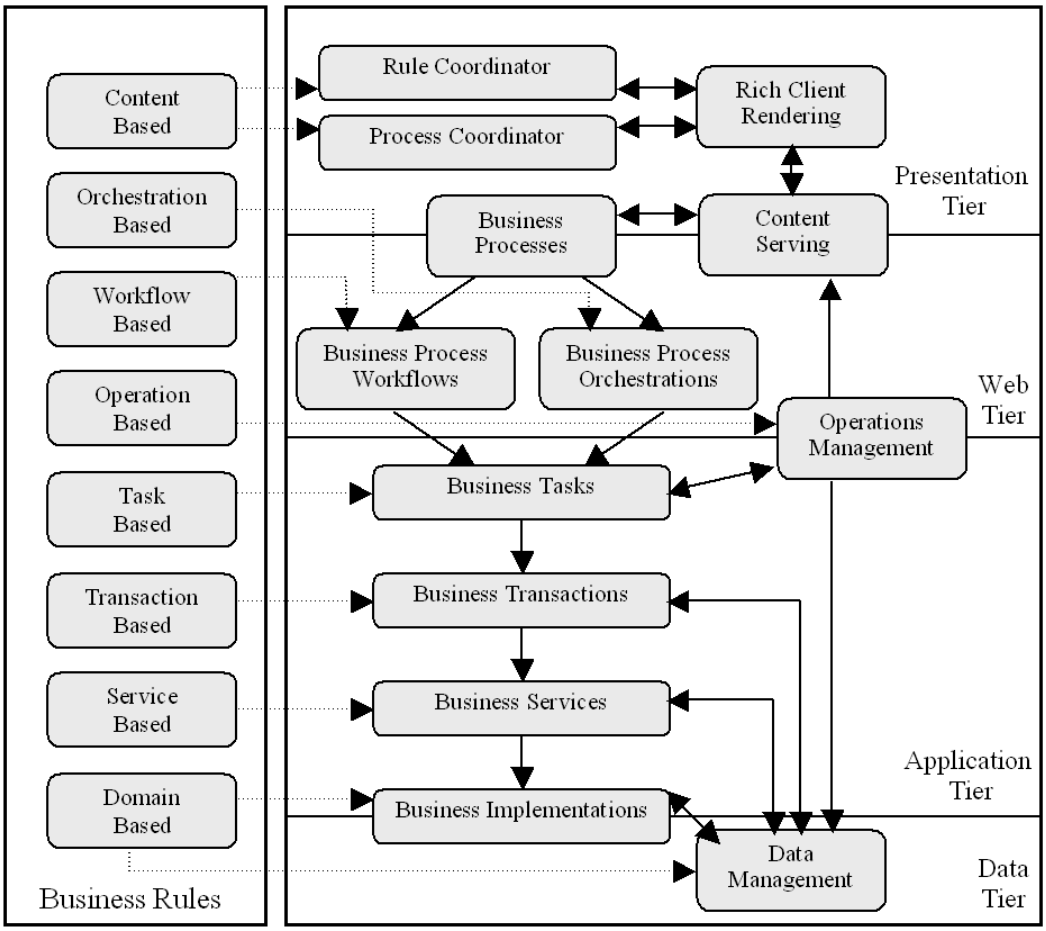
1.Introduction

The requirement analysis part has always been critical to design, so it is vital for the next steps of the design process. In this report, the below objectives will be taken into consideration :

- to describe what the customer (corporation in our case) requires
- to establish a basis for the creation of a software design
- to define a set of requirements that can be validated once the software is built

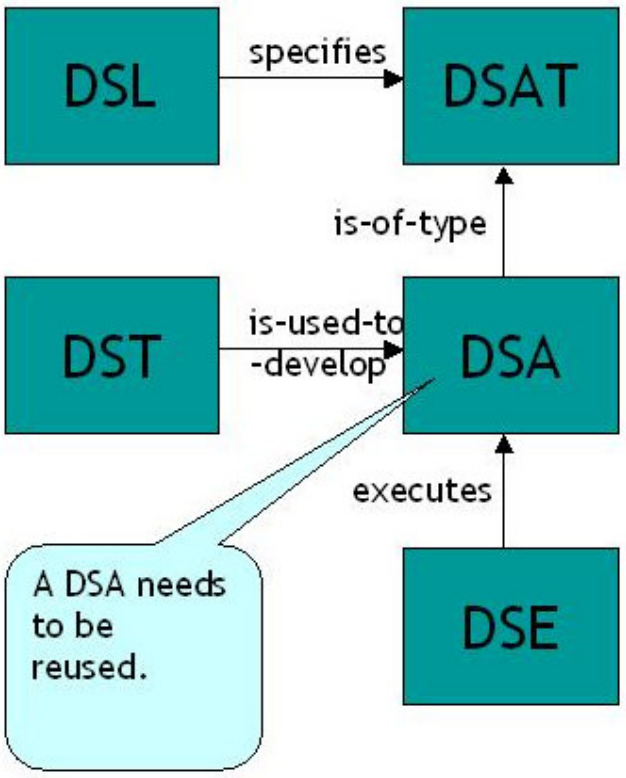
1.1. Background and Overview

Business rules are one of the crosscutting business concerns in enterprise information modeling. We can specify generally a Business Rule Management system as below :



- Business Process Workflow is the flow of internal processes within an organization.
- Business Process Orchestration is the supervision of internal and external processes.
- Business Task is an activity within an organization to be performed by or delegated to people. It may contain one or more Business Transactions.
- Business Transaction is the set of short term and long term actions to carry out a specific business activity.
- Business Service is the set of interrelated Business Implementations managed by a computational model.
- Business Implementation is the set of data processing commands under a computational model.

1.2. Project Definition



1.3 Project Goals

We hope to achieve all the goals below by the end of the project :

- Creating or extending from a known language a generic Domain Specific Language
- Creating an engine and toolset for the language
- Creating a Kit that includes all above and has a simple structure for easy use.

1.4 Scope of the Project

Scope of the project is limited with:

- Providing a generic Domain Specific Kit which includes a Domain Specific Languge , and Engine ,and,Tool for users or customers.
- Resource controlling
- Analysis of the market in terms of existing products
- Analysis of user requirements

2. Project Process

The process model ,and team organisation ,and process constraints are fundamental parts of project process. They are explained below in detail.

2.1 Process Model

Depending on the characteristics of the project, we reached an agreement on using waterfall model as the most suitable process model for our project. Since there is a strict schedule with deadlines that is given at the beginning of the project, using waterfall matches best with the case. No overlap or iteration is allowed during any period of the

process. After a phase is completed we will not turn back for large scale modifications. The most criticized property of waterfall model is the absence of evolutionary approach and supplying the product only after all the phases are over. But in the implementation phase we will provide several prototypes in order to get rid of the mentioned disadvantage.

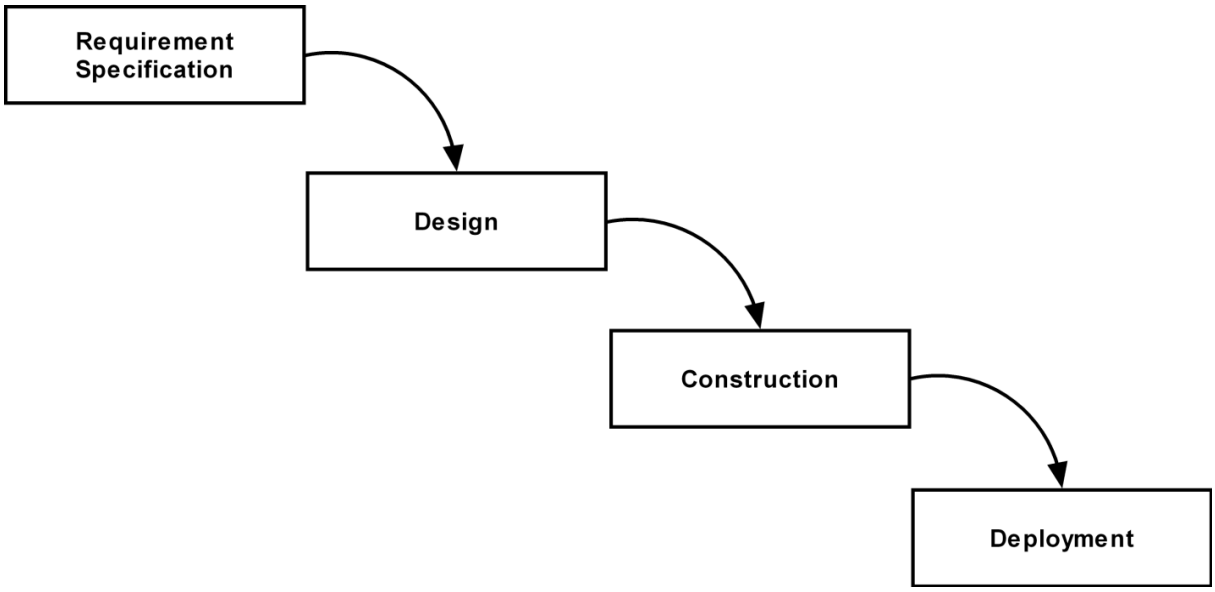


Figure 1 - Modified Waterfall Model

2.2 Team Organization

The most proper team organization category for our group is *Democratic Decentralized* (DD).

- Our reasons to choose DD are:
- Consensus plays an important role in our decision making process
- We have no permanent team leader
- Every member of the team should understand how things are handled
- Communication is horizontal
- We have coordinators for tasks, namely rotating task coordinators, but these coordinators may change as tasks change

2.3 Project Constraints

Project constraints can be grouped like the following:

2.3.1 Time Constraints

Since senior project design is a two semester course, the project will have to be finished by the end of May 2007. All design, implementation and testing must strictly meet this deadline and complete in this 7 month period.

2.3.2 Language Constraints

Since we have a limit time to finish our project and we all can code in C++ , the language for the project is decided to be C++.However, we also think about using Ruby in our project as another choice. Platform independency and code portability is an implementation constraint, thus all C++ code for this project will conform to ISO C++ standard. Development environment will be Visual C++ for Windows port, and a suitable GCC based environment for Unix/Linux ports.

2.3.3 Data Constraints

A fair amount of primary storage space is required to hold various data structures used for analyzing data flow over the network. If the user chooses to save some data for later analysis.

2.3.4 User Interface Constraints

The interface must be kept simple and easy to use. Names of *menu*s and other *gui* elements will be easy to understand and straightforward. Accessibility features must be taken into consideration for handicapped users.

3. Requirement Specifications

3.1. Interface Requirements

We will state the interface requirements for business rule system in this part.This system includes a controlled iteration of specific tasks to obtain, model, and implement business rules. This cycle of tasks implementation involves a multi-phase approach that coordinates operational an organizational demands. Once business rules are granted, explained formally and systematically managed, they become a powerful tool for business users. Users will access business rules by no longer exhaustive codes. Especially non-programmer users will easily understand and manage business rules.

A general example for configuration of this business rule system consists of inference engine,rulebase, and user interface with applications. The system also includes database system because of data transactions. The inference engine consists of inference algorithm and working space. With the usage of internal rules algorithm solves problems. The working space stores necessary data in memory. Figure 1 shows a general business rule system.

Making the model for rule expression requires two processes. First one is defining classes with associated attributes. The other one is setting relationships among the classes. The rule systems classify codes before the use in the checking progress , and store the fact as rule model in the rulebase. The business rule objects are generated from the prerequisite condition, stored in the knowledge base, and described in the rule language.

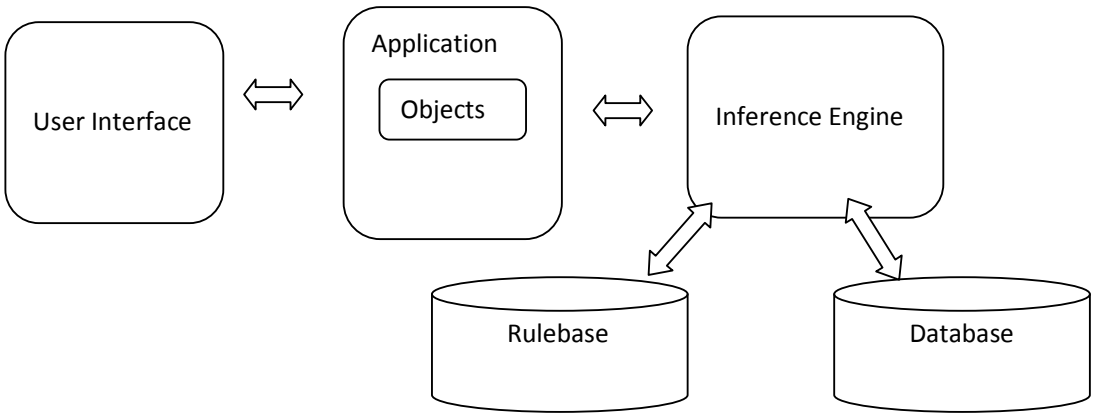
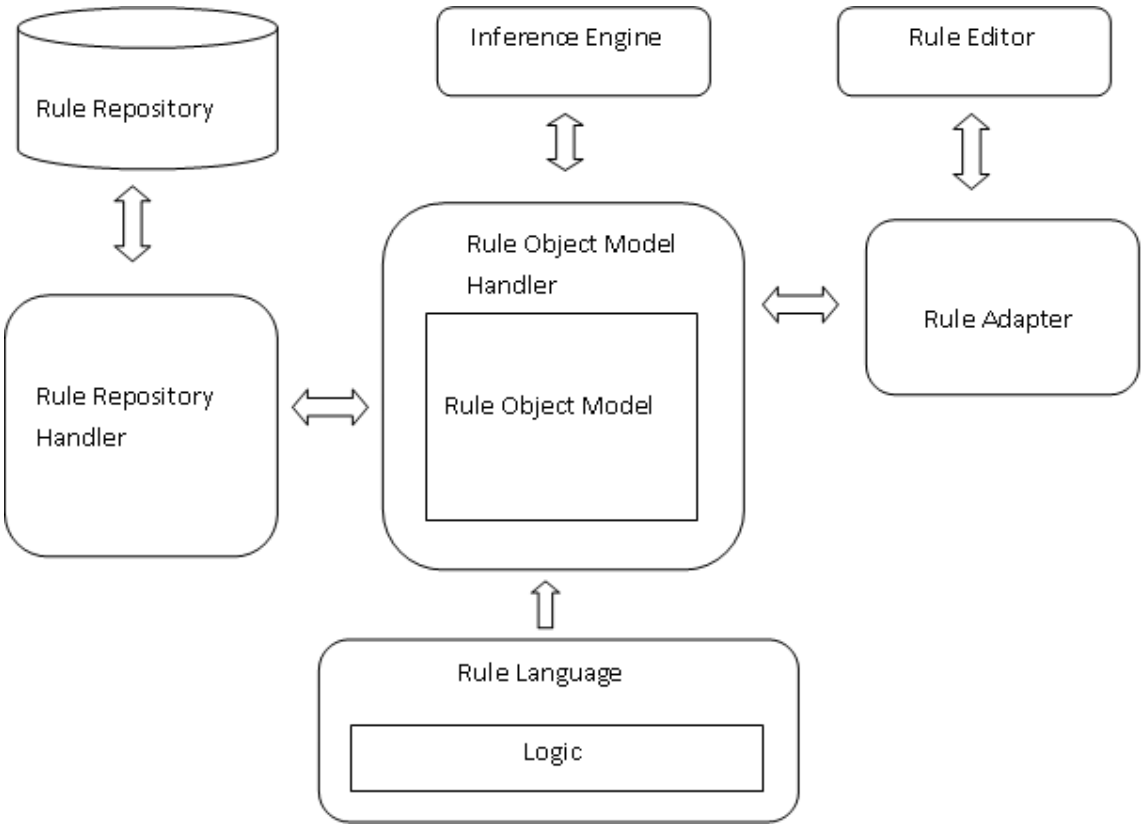


Figure 1

The architecture of proposed business rule system is shown below in Figure 2. This system has a rule editor that provides a user interface, an inference engine that applies the rules using its algorithms, a rule repository that saves information related to the rule, a rule object model handler which controls over the rule object model, the rule object model that express the rule class data, rule adapter that supply an interface between rule editor and rule object model handler,rule repository handler that control rule repository, and so on.



- Rule Editor:

This is an application program which is a component of the business rule system. This provides users with the environment of graphic user interface in order to write rules easily and conveniently. Rule editor can be replaced with other application program related to the rule.

- Rule Adapter:

The system needs to be parsed in fixed form from input rule project file. This component performs the parsing function, and defines the interface between the rule object model and the external application program.

Rule Object Model:

This is an object model representing the rule related class information such as contents of the rule project.

ROM Handler:

This is the handler component for the rule object model. This generates and controls the rule object model from the parsed result of the document that is written in the rule markup language. This also defines interface between the rule adapter and the rule object model.

- ### Rule Language:

The rule language expresses the rule. We will use a XML based rule markup language. This language will be created by us which will be based on a language already exist, modified and extended.

- ### Inference Engine:

This is a mechanism program that automates functions for the control logic and control object, and is able to apply rules easily. This component includes the routine for the inference algorithm.

- ### Rule Repository Handler:

This is a component for storage and extraction of various data through the connection with the rule repository. This component provides an easy accessible interface for the stored data in the rule repository.

- ### Rule Repository:

The rule repository is a repository that stores the rule related information and supports the flexibility of rule expression. Although the business rules are separated application program codes as business logics in the system, they can be managed concentrically in a place.

The rule repository is generally structured to facilitate a broad distribution of business rules to heterogeneous systems. It includes the support for multiple business rule definitions and the associated rule logic for multiple languages. The rule repository defines all ofthe necessary attributes for a rule that will allow it to be effectively introduced, processed and retired with minimal intervention.

The rule repository also stores all of the attributes necessary to evaluate the execution order of business rules.

3.2. Functional Requirements

3.2.1 Business Rule Requirements

In this section we are going to define business rule related requirements for all actors of the production. Each specific actor will be examined separately and in a detailed way.
We will provide support for the rule-related requirements of all users. They are namely business process owner, business rule steward, business analyst and business rule administrator.

Business Process Owners' Requirements:

Business process owners will have the ability to trace each business rule to all its related information.Defined rules will be categorized and all relations between other rules can be monitored. When the owner make a request to monitor a business rule system will search the rule and response to user's request. Furthermore they can access the definitions of all business rules currently in effect for
business rule evaluation and change.

Basic flow of usecase:

- The user enters a rule name into system or select the related category from list.
- System displays a list of matching rules to the user.
- The user selects a rule.
- System displays rule's details to the user.

Business Rule Stewards' Requirements:

As a part of a business rule change process related stewards will be able to create new versions of business rules. Through the approval process they can track the progression of proposed changes.

Basic flow of usecase:

- The user enters new rule or updates a existing rule.
- System checks the rule conditions (e.g. in use, contradict with other rules, not exist) and response accordingly and displays the progress.

Business Analysts' Requirements:

One of the actors who will use our system is business analysts. They will perform business rule change impact analysis (to other rules, process, data, objects, policy, objectives, etc.).
For possible business rule reuse research of existing business rules will be enabled for business analysts. Also they can perform business rule validation.

- Basic flow of usecase:
- The user request for a rule.
- System displays rule and enable user to monitor and report the contradictions,misusages.
- The user make correctness
- System reorganize rules if it is suitable reject otherwise.

Business Rule Administrators' Requirements:

Business rule administrators need to establish multiple physical business rule repositories. And they should maintain consistency among these multiple physical repositories using comparison and checkin/checkout procedures.Besides we will enable them to test the integrity of business rule information in the repository. One of the requirements of administrators is to administer repository access and security procedures and software.

According to our researches about business rules they are responsible for implementation and enforcement of business rule naming standards in the repository.

Basic flow usecase:

- The user requests rule repository in administration mode.
- System serve repository if appropriate.
- User make needed changes and request updating
- System has its defence mechanism if something wrong it terminates the session otherwise updates the repository.

3.2.2 DSL Requirements

Our DSL will have query and reporting capability for impact analysis, traceability and business rule reuse including web-based publication. Besides we are going to provide security for and the integrity of business rule and rule-related information. Some decided requirements of the DSL are stated below.

- **Conformity:**

  The DSL will be totally conform with business rule management. It will include specific rule engine and rule repository for this purpose.

- **Orthogonality:**

  Each construction in the language will be used to represent exactly one distinct concept in the domain.

- **Supportability:**

  It will be feasible to provide DSL support via tools, for creating, deleting, editing, monitoring, reporting and executing rules.

- **Integrability:**

  The language can be used in different environment and can be integrated to different systems.

- **Longevity**:

  The DSL should be used and useful for a non-trivial period of time in order to ensure tool support, and to make it possible to quantify to the DSL stakeholders the payoff obtained from using the DSL.Especially being maintable and being generic in its domain let the languge be long-lasting.

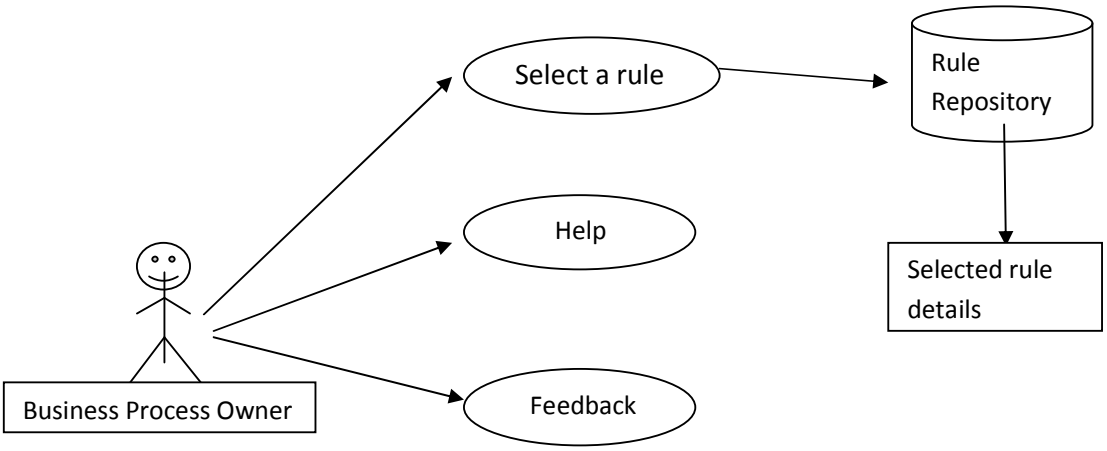
- **Simplicity:**

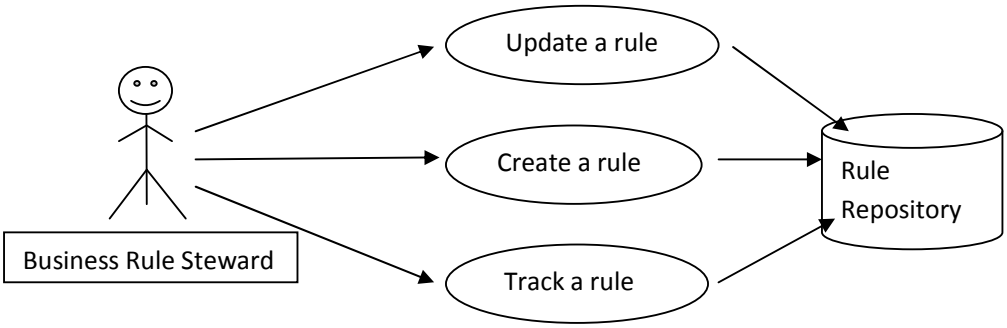
  This is a generally desirable language requirement: The language should be as simple as possible in order to express the concepts of business rule management and to support its users and stakeholders in their preferred ways of working.

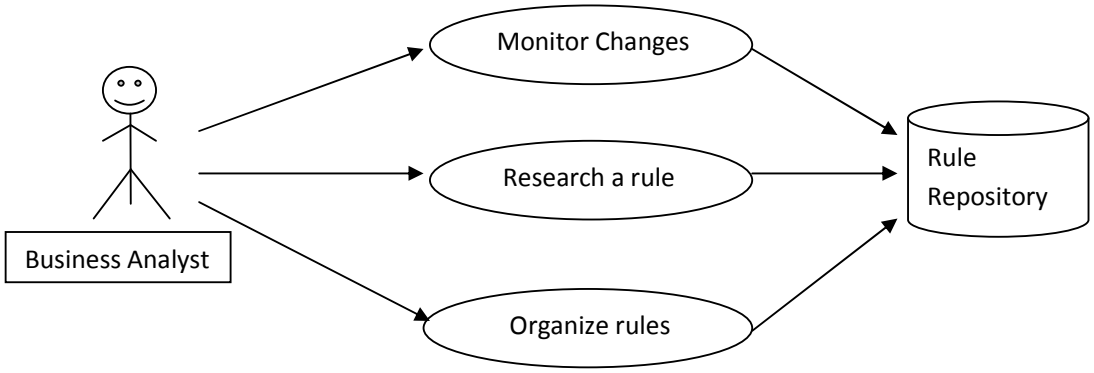
- **Quality:**

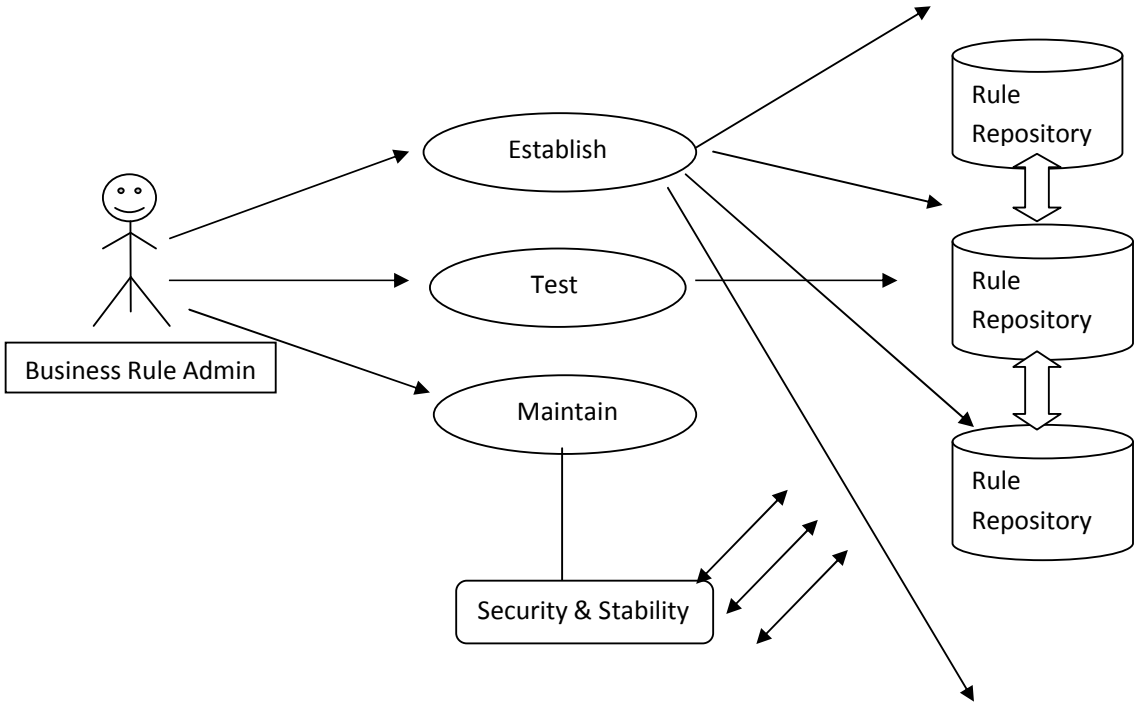
  The language shall provide general mechanisms for building quality systems. It will include improving security and stability which are very important specifications for business rule management.

4. System Analysis and Modelling








Establish
Test
Maintain
Business Rule Admin
Security & Stability
Rule Repository
Rule Repository
Rule Repository

5.Gantt Chart