**Decision Engine Platform**

### BITS ZG628T: Dissertation

by

Pramod Kumar N

2014HT13292

# Dissertation work carried out at

## Coextrix Technologies Pvt. Ltd., Bengaluru

****

**BIRLA INSTITUTE OF TECHNOLOGY & SCIENCE**

**PILANI (RAJASTHAN)**

November 2016

**Decision Engine Platform**

### BITS ZG628T: Dissertation

by

Pramod Kumar N

2014HT13292

# Dissertation work carried out at

## Coextrix Technologies Pvt. Ltd., Bengaluru

Submitted in partial fulfillment of M.Tech. Software Systems degree programme

Under the Supervision of

Mr.Ramesh Krishnamoorthy, CEO,

Coextrix Technologies Pvt. Ltd., Bengaluru

****

**BIRLA INSTITUTE OF TECHNOLOGY & SCIENCE**

**PILANI (RAJASTHAN)**

November, 2016

#### 

#### CERTIFICATE

This is to certify that the Dissertation entitled Decision Engine Platform and submitted by Pramod Kumar N having ID-No. 2014HT13292 for the partial fulfillment of the requirements of M.Tech. Software Systems degree of BITS, embodies the bonafide work done by him under my supervision.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Signature of the Supervisor

Place : \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Date : \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Name, Designation & Organization &Location

iii

**Birla Institute of Technology & Science, Pilani**

**Work-Integrated Learning Programmes Division**

**First Semester 2016-2017**

**BITS ZG628T: Dissertation**

**ABSTRACT**

**BITS ID No.** **: 2014HT13292**

**NAME OF THE STUDENT** **: Pramod Kumar N**

**EMAIL ADDRESS** **: pramod974@gmail.com**

**STUDENT’S EMPLOYING** **: Coextrix Technologies, Bengaluru**

**ORGANIZATION & LOCATION**

**SUPERVISOR’S NAME** **: Ramesh Krishnamoorthy**

**SUPERVISOR’S EMPLOYING** **: Coextrix Technologies, Bengaluru**

**ORGANIZATION & LOCATION**

**SUPERVISOR’S EMAIL ADDRESS : ramesh.krishnamoorthy@coextrix.com**

**DISSERTATION TITLE** **: Decision Engine Platform**

iv

**Abstract**

In modern day applications that provide more visibility into data irrespective of domain, require collecting data from several different sources. Collected data needs to be consolidated, normalized and then presented through an application. There is a lack of correctness in the data aggregated owing to the nature of issues with the sources of data itself. Since the original data sources are doing nothing to clean up the data they provide or generally there is no standardization maintained across domains in context. Several applications are designed to clean up the data coming through various sources and allow customers to perform complex analytics on top of the data. Transforming and reconciliation process is extensively complex, but is required to create higher accuracy in the data. The transformation and reconciliation process comprises of multiple rules, interactions with multiple components in order to achieve the accuracy. In popular existing open source technologies the way the rules are written and configured is stereotype and hard to maintain owing to the quantity of rules modelled and its ability to scale in future.

There are no matured open source decision engines built using python which satisfies applications current needs, where there is an end to end feature to model and maintain rules which could interact with multiple sources and components. Accurate data has been always sought for and is the next gen thing to get custom insights irrespective of domain. The underlying need to build a configurable and maintainable platform is highly compelling, where a Decision engine facilitates connecting to various sources and components in order to write domain specific configurable rules so as to achieve valuable decision in limited turnaround time. The Goal is to build, deploy and scale the existing stereotype rules/models to a more maintainable and configurable Decision engine and also provide an easy interface and framework to model rules on the fly build, test and deploy it without any manual intervention.

Broad Academic Area of Work: **Software Architecture**

Key words: **Rules, Engine, Decision, Python, Platform, Framework, Configurable**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Signature of the Student**

**Name: \_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Date:**

**Place:**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Signature of the Supervisor**

Name: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Date:**

**Place:**

v

**ACKNOWLEDGEMENT**

The satisfaction that accompanies the successful completion of dissertation would be incomplete without mentioning the people who made it possible, whose constant guidance and encouragement crowns all the efforts with success.

I express sincere gratitude to Mr.Ramesh Krishnamoorthy, CEO, Coextrix for being kind enough to provide me an opportunity to work on a project in this organization and also being my mentor.

I am thankful to Mr. Ganesh Guruswamy, CTO, Coextrix for providing valuable ideas, suggestions, facilities, co-operation and his encouragement at all moments of my approach towards tackling the problem.

Last but not the least, I would like to express my love and gratitude to my beloved family and friends, for their understanding & motivation, through the duration of this project.

vi

**Table of Contents**

Certificate………………………………………………………………………………………………………………………………iii

Abstract…………………………………………………………………………………………………………………………………iv

Acknowledgment………………………………………………………………………………………………………………….vi

Table of Figures ix

Chapter 1: Introduction 1

1.1 Data Source 2

1.2 Problems and Challenges with Data 2

1.3 Data Transformation and reconciliation…………………………………………………………………………………….….3

1.4 Software Architectures and Design Patterns 4

Chapter 2: Project Overview 5

2.1 Existing System 5

2.2 Problems with Existing System 5

2.3 Available Software Solutions and their issues…………………………………………………………………………….5

2.4 Proposed System…………………………………………………………………………………………………………………………………..5

2.4.1 Rules……………………………………………………………………………………………………………………………………………………..5

2.4.2 Rule Orchestration.……………………………………………………………………………………………………………………………..5

2.4.3 Decision Engine …5

**Table of Figures**

Figure 1: Knotted rules in Existing System 4

Figure 2: Proposed System Decision Engine 6

**GLOSSARY**

REST – Representational State Transfer

API – Application Program Interface

CSV – Comma Separated File

MSEXCEL – Microsoft Excel

**Chapter 1: Introduction**

**1.1 Data Sources**

Data has been most sought for in the modern era to gain deeper and meaningful insights into ones business. The insight from data is required irrespective of which domain one is working on, in order to gain such insight data has to be curated from multiple sources. The data source [1] is can be typically a connection set up to a carious computer databases running as a server or it could be as simple as a file (CSV, text, MSEXCEL, etc.) or it could just be a stream of data coming in via API or live feed.

**1.2 Problems and Challenges with Data**

The volume, correctness and consistency pose a major challenge with the data collected from various sources to be processed before they can be analysed. Some businesses require domain specific incorporations as well to meet the data quality issues. Some of the common data related problems and challenges are as follows.

1. Poor data quality such as noisy data, dirty data, missing values, inexact or incorrect values, inadequate data size and poor representation in data.

2. Integrating conflicting or redundant data from different sources and forms: multimedia files (audio, video and images), geo data, text, social, numeric, etc…

3. Proliferation of security and privacy concerns by individuals, organisations and governments.

4. Unavailability of data or difficult access to data.

5. Efficiency and scalability issues to effectively extract the information from huge amount of data in databases.

6. Dealing with huge datasets that require distributed approaches, dealing with non-static, unbalanced and cost-sensitive data.

7. Constantly updating model to handle data velocity or new incoming data.

8. High cost involved in buying and maintaining powerful softwares, servers and storage hardwares that handle large amounts of data.

9. Processing of large, complex and unstructured data into a structured format.

10. Sheer quantity of output from the data curated.

**Page 1 of 6**

**1.3 Data Transformation and Reconciliation**

Owing to multi natured problems with collected data, it has to be transformed and reconciled. Typically along with all standard process available for transformation and reconciliation, business use domain specific rules and knowledge to transform and reconcile the data so they are analysis ready.

Data transformation is the process of converting data or information from one format to another, typically from the format of a source system into the required format of a new destination system. The general process involves converting documents, but data conversions sometimes involve the conversion of a program from one computer language to another to enable the program to run on a different platform. The usual reason for this data migration is the adoption of a new system that's totally different from the previous one.[4]

In real practice, data transformation involves the use of a special program that's able to read the data’s original base language, determine the language into which the data that must be translated for it to be usable by the new program or system, and then proceeds to transform that data.

Data Transformation involves two key phases:

Data Mapping: The assignment of elements from the source base or system toward the destination to capture all transformations that occur. This is made more complicated when there are complex transformations like many-to-one or one-to-many rules for transformation.

Code Generation: The creation of the actual transformation program. The resulting data map specification is used to create an executable program to run on computer systems.

**Industrial process data validation and reconciliation**, or more briefly, **data validation and reconciliation (DVR)**, is a technology that uses process information and mathematical methods in order to automatically correct measurements in industrial processes. The use of DVR allows for extracting accurate and reliable information about the state of industry processes from raw measurement data and produces a single consistent set of data representing the most likely process operation.[5]

**1.4 Software Architecture and Design Patterns**

Software architecture refers to the fundamental structures of a software system, the discipline of creating such structures, and the documentation of these structures. These structures are needed to reason about the software system. Each structure comprises software elements, relations among them, and properties of both elements and relations, along with rationale for the introduction and configuration of each element.

**Page 2 of 6**

The architecture of a software system is a metaphor, analogous to the architecture of a building. The software architectures comprises of following activities.

Architecture supporting activities

Software architecture supporting activities are carried out during core software architecture activities. These supporting activities assist a software architect to carry out analysis, synthesis, evaluation and evolution. For instance, an architect has to gather knowledge, make decisions and document during the analysis phase.

Knowledge Management and Communication is the activity of exploring and managing knowledge that is essential to designing software architecture. A software architect does not work in isolation. They get inputs, functional and non-functional requirements and design contexts, from various stakeholders; and provide outputs to stakeholders. Software architecture knowledge is often tacit and is retained in the heads of stakeholders.

Design Reasoning and Decision Making is the activity of evaluating design decisions. This activity is fundamental to all three core software architecture activities. It entails gathering and associating decision contexts, formulating design decision problems, finding solution options and evaluating tradeoffs before making decisions. This process occurs at various levels of decision granularity, while evaluating significant architectural requirements and software architecture decisions, and software architecture analysis, synthesis, and evaluation. Examples of reasoning activities include understanding the impacts of a requirement or a design on quality attributes, questioning the issues that a design might cause, assessing possible solution options, and evaluating the trade-offs between solutions.

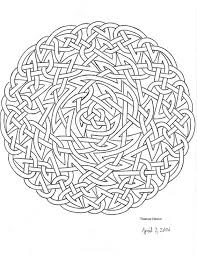
Documentation is the activity of recording the design generated during the software architecture process. A system design is described using several views that frequently include a static view showing the code structure of the system, a dynamic view showing the actions of the system during execution, and a deployment view showing how a system is placed on hardware for execution.

**Page 3 of 6**

**Chapter 2: Project Overview**

* 1. **Existing System**

The existing system consists of multiple nests of conditional statements and rules in order to reconcile the data to take decisions. The domain specific rules written are stereotype and interlock themselves and difficult to scale them by adding new rules. The natural tendency of any software is to undergo changes in the due course of time. The changes if any are very difficult to incorporate and integrating and creating the new rules and propagating through the system are very complex in nature. Due to infeasibility the original system design cannot be incorporated due to compliance issues. The goal of the existing system is to take data as input from various sources such as CSV, text files, Database systems and data feeds etc. apply certain domain specific rules to the data and give the output of the file and also state the different anomalies so that the business analysts can take decisions.



**Figure 1: Knotted rules in Existing System**

* 1. **Problems with Existing System**

There are lot of complex and nested conditional statements which are very difficult to maintain and scale it, any new addition should undergo rigorous integration testing and there is lot of resources, time and cost involved. There are many loose ends that needs to be joined together so various components have more visibility into each other.

* 1. **Available Software Solutions and their Issues**

Modern technologies have spanned across various business aiding them at different levels. Business Rule engines form the foundation in modelling multiple business rules. There are lot of matured technologies available as proprietary software. Few of the available popular

**Page 4 of 6**

softwares are Drools, InRule etc. which are expensive and learning curve is huge. Since our existing systems components consist of only open-soures technologies, integrating them with available proprietary software requires lot of effort and huge licencing cost.

**2.4 Proposed System**

The rules grow exponentially and the business analysts want to explore innovative way to model rules on the fly and also add them without any manual intervention with little scripting background they want to leverage open source options where they can have a lot of flexibility and no licensing cost. The decision engine platform provides end to end features so as to satisfy all the current application needs to be more configurable , scalable and automatically deployable to get insights. The whole systems leverages the rules design pattern and completely built using open source technologies.

The proposed system will have the following components

* Rule repository
* Rule Orchestration
* Decision Engine

**2.4.1 Rule Repository**

The Rule Repository consists of multiple modules which can be modified, deleted and incorporated on the fly. The rules in this repository is easily configurable and for the foundation for the Rule orchestration.

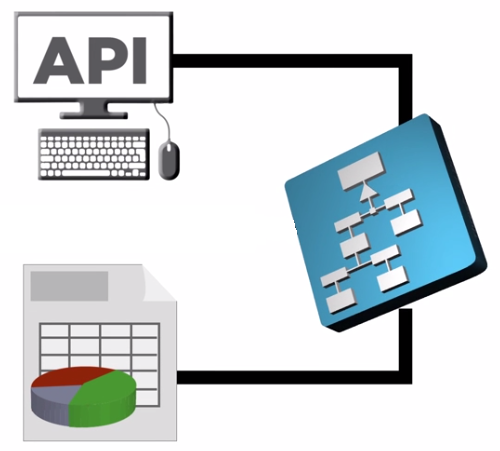
**2.4.2 Rule Orchestration**

The Rule Orchestration consists of meta class which whose modules and attributes get generated on the fly depending on which got modified, deleted and incorporated on the fly in the rule repository. The rules in repository is made ready for the Decision engine to consume.

**2.4.2 Decision Engine**

The decision engine forms the crux of this whole platform. It uses the rule orchestration class to execute the required domain specific rules on the data got either from the API or a CSV. Figure 2 shows the high level flow of how multiple rules can be applied on streams from an API or CSV.

**Page 5 of 6**



**Figure 2: Proposed System Decision Engine**

**Page 6 of 6**

**References**

1. Data Source Wiki, https://en.wikipedia.org/wiki/Data\_source
2. Computer File Wiki, https://en.wikipedia.org/wiki/Computer\_file
3. Data Stream Wiki, https://en.wikipedia.org/wiki/Data\_stream
4. Definition, https://www.techopedia.com
5. Data Reconciliation-https://en.wikipedia.org/wiki/Data\_validation\_and\_reconciliation