2019

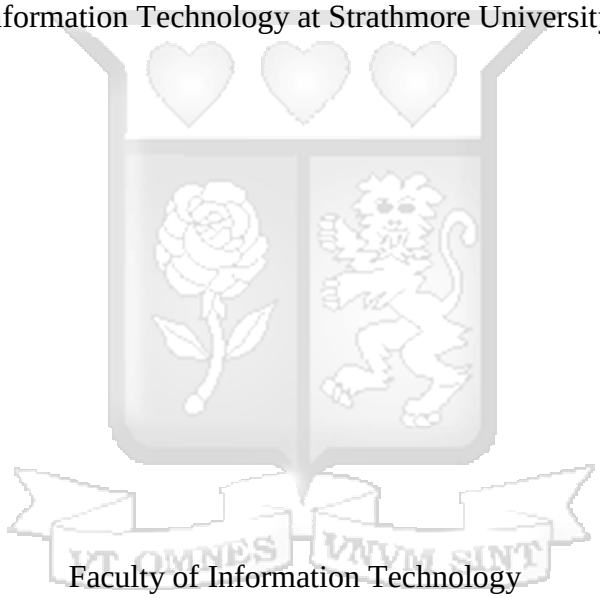
Intellibot data cleaner: a study of Kenya Revenue Authority's data cleaning exercise

Jerry O. Odero
Faculty of Information Technology (FIT)
Strathmore University

Intellibot Data Cleaner: A Study of Kenya Revenue Authority's Data Cleaning Exercise

Odero, Jerry Omondi

Submitted in partial fulfillment of the requirements for the Degree of Master of Science in Information Technology at Strathmore University



Faculty of Information Technology

Strathmore University

Nairobi, Kenya

June, 2019

Declaration

I declare that this work has not been previously submitted and approved for the award of a degree by this or any other University. To the best of my knowledge and belief, the thesis contains no material previously published or written by another person except where due reference is made in the thesis itself.

Odero, Jerry Omondi

……..

07th June 2019

Approval

This thesis of Jerry Omondi Odero was reviewed and approved by the following:

Dr. Vincent Omwenga

Senior Lecturer, Faculty of Information Technology

Strathmore University

Dr. Joseph Orero

Dean, Faculty of Information Technology

Strathmore University

Professor Ruth Kiraka

Dean, School of Graduate Studies

Strathmore University

Abstract

Data cleaning is an activity involving detecting and correcting errors and inconsistencies in a database, data warehouse or any data record of an organization. Kenya Revenue Authority (KRA) in its quest to be a fully data driven organization, is actively undertaking the data cleaning process. However, this process is currently manual and slow as it involves physical transfer of documents to be processed from the various stations, via different levels of management for approval, to the centralized return processing unit. A process, which might take at least a fortnight for the processing of one taxpayer's ledger account. Furthermore, this whole process needs lots of man-hours, since there is a vast amount of data to be cleaned due to the many ledger accounts affected during the manual filing system that ended in 2014.

There exists many data cleaning processes and approaches which are used to purge out "dirty data", before it's loaded into the data warehouse. These processes vary depending on the data source, they are time consuming and expensive for organizations, in terms of skilled staff and the tools involved, hence this research proposed the application of RPA (Robotic Process Automation) to develop an intelligent bot (Intellibot) to be used in the transactional data cleaning exercise in Kenya Revenue Authority (KRA).

With the transition from legacy system to I-Tax and I-CMS systems for domestic and customs revenue management respectively, the researcher sought to find out the current data cleaning process in the legacy system. This research led to the development of an RPA system for the current manual data cleaning process implemented and tested using the Blue Prism platform. The system detected the errors – using a knowledge-based model-, clustering them as errors due to uncaptured returns, uncaptured losses or credit re-adjustments. The intellibot system was able to load the ledgers, detect the errors and clean them with utmost precision. Experiments conducted on performance of the bots varied by seconds, in the first experiment. Also in the second performance test, there was a variance of seconds in cleaning the different errors detected, hence improving the data integrity significantly: free of errors, to be migrated to the I-Tax platform, thus support better decision making process in the organization, and a higher return on investments.

Keywords: RPA, Data Cleaning, Knowledge-based Reasoning model, KRA, Blue Prism, Legacy Systems, Intellibot.

Table of Contents

List of Tables

List of Figures

CHAPTER 1:

INTRODUCTION

1.1 Background

Organizations today have adopted complex business processes which have outlived the manual era and thus created an ardent need for automation. The sophistication and dynamicity of business environments has engendered different business automation tools in order to leverage on the timely technological advancement that is revolutionizing businesses globally. According to Mugisha (2010), the frequent and uncertain changes, greater competition between firms, the need for continuous innovations, quality enhancement and cost reduction force companies to face the challenge of improving their competitiveness and consequently their performance. This leads to the need for scouting for more advanced, feasible, automation tools that can help reduce costs, improve performance and create competitive advantage.

In a bid to further its automation, Kenya Revenue Authority (KRA) has advanced in its business automation by adopting wholly integrated tax revenue administration systems. This has led to an increase in tax collection since its inception as it caters for over 93% of total government income. KRA plays a critical role in facilitation of trade and investments, both within and without Kenyan borders, and due to the voluminous transactions, automated control is mandatory in order to achieve efficient performance. Taxation can be defined as the obligation by government of compulsory levies and contributions on the public, assets, income, merchandise and business transactions, with an aim of raising government income or revenue for expenditure. Kenya Revenue Authority (KRA) is the mandated government's statutory organization for tax revenue collection. Taxes are collected for financial and societal services such as health, national security, education and infrastructural developments; secondly, to increase the money sent or catering for the poor; to promote investments; and to protect local markets on domestic products through heavy taxes on unnecessary imports (KRA, 2012).

KRA like any other business organization needs to perform excellently in service delivery – in terms of speed of access to the systems, accuracy, quality and integrity of the data to the taxpayers thus the adoption of the I-Tax (domestic tax management system) and I-CMS (Customs

Management system); information systems for business process automation of the various tasks performed by officers in their daily business operations. According to Norton (2006), the goal of Information Systems, was not only to provide citizens, economic organizations, companies and institutions with a range of excellent and effective services, but they also created a new form of citizenship based on the participation of all individuals in the provision of services and the decision-making process which was aided by the intensive use of new Information. However, this cannot be achieved with the existence of legacy systems (ITMS –Information tax management system) which carries vast master and transactional data that affects taxpayers' accounts status.

Robotic Process Automation (RPA) comes in handy for the data cleaning process; as a software, that operates other software. Robotic Process Automation is an application of a cost-effective software that mimics human actions and connects multiple fragmented systems together through automation without changing the current enterprise IT landscape (Ernst and Young, 2018). It is automation to replace humans performing repetitive rules-based tasks, and offers cross-functional and cross-application macros able to operate any software on a client's computer. Unlike the physical robots, it's not a walking-talking auto bot, physically existing machines processing paper or voice recognition and reply software but rather a program that can emulate human tasks using existing technology user interfaces, can execute keystrokes much faster than its human counterpart can without taking breaks. This leads to increase in speed, accuracy and volume of select repetitive processes performed within a function (Price Waterhouse Coopers, 2017).

RPA is delivered through software that can be configured to perform rules-based tasks including read and write to database, log in to web/enterprise applications, move files and folders, open e-mail attachments, copy and paste, and scrape data from the web. It can also connect to system APIs, extract structured data from documents, collect social media statistics, fill in forms make calculations and follow "if/then" decision/rules (Ernst and Young, 2018).

According to Sonali and Deepali (2013), data cleansing or (data scrubbing) is an activity involving a process of detecting and correcting the errors and inconsistencies in data warehouse. Thus poor quality data i.e.; dirty data present in a data mart can be avoided using various data cleaning strategies, and thus leading to more accurate and hence reliable decision making. In view of this, the legacy systems that were initially used contains a lot of errors that cannot be migrated to the I-

Tax or I-CMS systems and therefore a duty to flag of the errors and present clean data to be presented to the new systems is inevitable.

The impetus of this study was based around an RPA model that was used to detect, cluster and clean the dirty data in the legacy system before it's migrated to I-TAX. The model followed the business rules already adopted and used them to automate the rules-based repetitive tasks involved in cleaning the taxpayers' ledgers.

1.2 Problem Statement

Kenya Revenue Authority hosts a vast amount of data from the daily transactions undertaken by taxpayers. Most of this data is in its legacy system, which has been in use until 2014, when I-Tax system was adopted for use by the Domestic Taxes Department. However, this transactional data in the legacy system suffers inefficiencies of integrity since it's erroneous.

The errors in the legacy system were introduced through the manual capture of returns, by staff who entered erroneous figures into the system from the manually filed forms by taxpayers. Others had missing figures and attachments like withholding tax certificates. Due to the transfer of returns to a central unit for capture, some returns also went missing. This error of commission and omission led to unreliable figures in the system and failure to capture some credits and losses in the system as well.

Currently, KRA has delegated the data cleaning exercise to a specific team, due to its complexity. However, the data cleaning process is currently done manually, as described hereunder: The common sources of error is either through omission or commission, failure to re-adjust credit movements, or wrong posting of receipts to the ledger. In all of these cases, confirmation of the errors has to be done by comparing the filed taxpayer returns with the records captured in the system, after which manual correction forms are filled, passed through the various managerial levels for approval, and finally to the returns processing unit for correction and reconciliation.(Domestic Taxes Department, 2010). Cleaning of at least one taxpayer's ledger takes at least a fortnight, rendering the whole process slow and costly hence ineffective.

This affects taxpayer's compliance ledger records, and the true debt position of the organization is nondescript since the collectible debt which is estimated at Kes 92.4 billion shillings is based on inadequate and indeterminate records. There is also need to transfer the legacy data to the current i-Tax system only after it's cleaned in order to reflect the true position of taxpayer records.

In view of the above, RPA will be used as a software bot that will be able to operate on other software and read documents, process spreadsheets and forms, log in to user accounts, and send emails in seconds, from the rule-based commands its set to operate on. This will increase the speed of processing taxpayer ledger records and reduce the costs immensely. The ledgers will be cleaned against the business processes governing the data cleaning processes set out in the Tax Acts. According to Ernst and Young (2018), RPA increases the speed, accuracy and volume of select repetitive processes performed within a tax function and removes the potential for transposition and other human errors that may arise.

The focus of this study therefore, was to develop an intellibot system that will be able to detect the errors and cluster them via a knowledge-based system model and process the data correction forms and input the right data in the taxpayer ledger records in seconds.

1.3 Aim

The purpose of this study was to develop an intellibot – An RPA- Knowledge-based driven system- for cleaning of transactional legacy data in KRA. The system is operated by bots and incorporates an expert module for intelligent error detection.

1.4 Specific Objectives

i. To investigate the features of errors associated with the "dirty data" in information tax management systems.
ii. To review the current data cleaning processes and approaches in information tax management systems.
iii. To develop an intellibot for the data cleaning process in KRA.
iv. To validate the intellibot for the data cleaning process in KRA.

1.5 Research Questions

i. What are the features of errors associated with the "dirty data" in information tax management systems?
ii. What are the current data cleaning processes and approaches in information tax management systems?
iii. How will the intellibot for the data cleaning process be designed?
iv. How will the intellibot for the data cleaning process be validated?

1.6 Justification

Most of the organization's legacy systems have been migrated to a new system supporting the current technology trends, but how much of repetitive tasks are involved in the data cleaning process, consuming the time of full time employees and increasing operation costs? This has created the need for RPA: a software to operate other software, thus minimizing human intervention on the system in order to be able to undertake more valuable professional tasks.

According to Burnett (2015, p.2), Robotic Process Automation is the next wave of innovation, which will change outsourcing. Through implementation of blue prism (RPA software tool), the race to become the top automation-enabled service provider in the industry is in the offing. In time, there will be an arms race for innovation in automation tools leading to new offerings and delivery models.

With rapid advancements in technology, tax should now look at or re-consider how RPA can effectively perform tasks that involve a high volume of transactions, repeatable manipulation of data, and communication with other enterprise systems efficiently—all at a much lower cost and minimal risk (Price Waterhouse Coopers, 2017).

Some of the robotic process capabilities include automated data entry, multi-system integration, repetitive tasks processing, process reconciliation, data validation/quality and processing simple business rules (Price Waterhouse Coopers, 2017).

In this research, we aimed to leverage on RPA's value for success in processes that primarily scale up by adding more labor, and processes that are prone to human error especially managing taxpayers' accounts. In addition to this, the likely outcome of this research gave us the following key benefits: transformative change in re-engineering of the core processes while automating the data cleaning functions or processes. It also anticipated 15-90% cost reduction in the operations involved in the process, standardization and optimization of the processes thus ensuring efficiency, quality and a high potential return on investments (ROI) since robots drive existing applications with low integration costs.

1.7 Scope and Limitation

This research focused its study only to the data cleaning process (ensuring the dirty data in the taxpayers ITMS legacy ledger accounts was transformed to clean data ready to be migrated onto the I-Tax platform). Further, the migration of the data to the new I-Tax platform was not considered in this study. The scope of study only involved the income tax company obligation (I.T.2.C), after which a similar model can be rolled out for the process in other tax obligations. It was also limited to detection of three types of errors: credit re-adjustment, uncaptured returns and unreflected losses.

CHAPTER 2:

LITERATURE REVIEW

2.1 Introduction

Every scientific study begins with the researcher examining reports of previous studies related to the topic of interest. Without this step, researchers cannot expect to construct and integrate a comprehensive picture of the world. They cannot achieve the progress that comes from building on the efforts of others. Also, investigators working in isolation are doomed to repeat the mistakes made by their predecessors (Cooper, 1998).

This chapter gives a review of the relevant literature to help us comprehend and better understand the research problem: the sources of errors leading to dirty data, the current data cleaning approaches and algorithms and literature on the working of RPA and Knowledge based system is also considered.

2.2 Sources of Errors in Data

The data quality problem has led to many data cleaning approaches, which have to satisfy several requirements. The approach should detect and remove all major errors and inconsistencies both in individual data sources and integrated multiple sources. Data should be performed together with schema-related data transformations based on comprehensive metadata (Rahm and Do, 2000). In this section, we classify the major problems associated with data quality that leads to “dirty data” and can be solved by data cleaning and transformations. According to Rundeinstener (1999), data transformations are necessary for supporting any changes in the structure representation and content of data in order to deal with schema evolution, migrating a legacy system to a new information system, or when integration of multiple sources is inevitable.

The data quality problems can be classified as single-source problems and multi-source problems. Under each classification cadre there exists schema and instance-related problems as shown in the figure below. Schema level problems can be addressed at the schema level by an improved schema design, schema translation and schema integration. Instance-level problems refer to errors and

inconsistencies in the actual data contents, not visible at the schema level. They are the main area of interest in data cleaning as discussed hereunder:

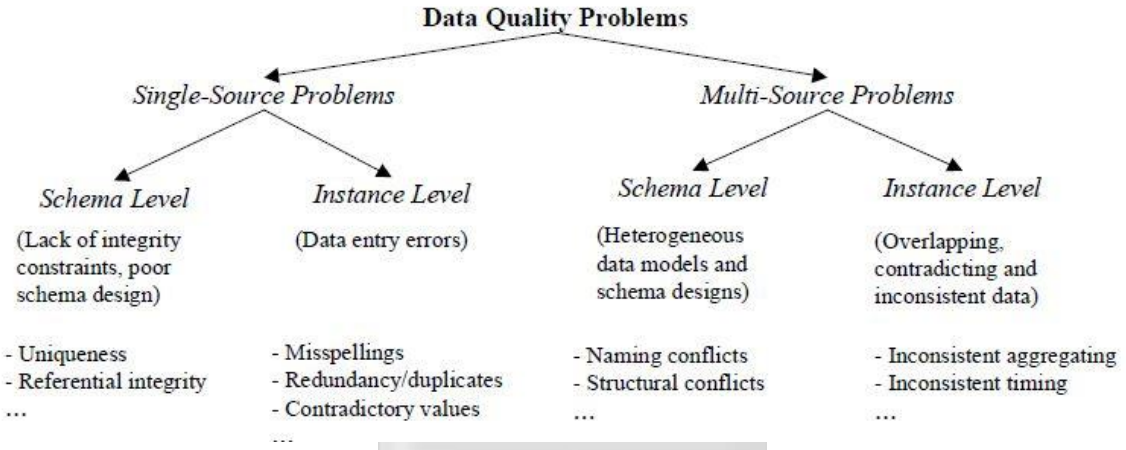


Figure 2.1: Classification of data quality problems in data sources. Adapted from "Data Cleaning: Problems and Current Approaches"

2.2.1 Single Source Error Problems

The data quality of a source is determined by the schema and integrity constraints governing data values allowed in the database. File sources have fewer restrictions leading to a lot of errors and inconsistencies. Schema-level data quality problems are due to lack of appropriate model-specific or application-specific integrity constraints. Instance-specific problems are caused by errors and inconsistencies which cannot be prevented at the schema-level e.g. omissions (Rahm and Do, 2000).

According to the table below different problem scopes can be classified as attribute (field), record, record type and source. Since the data cleaning process is expensive, it's prudent to prevent dirty data entry since it will reduce the data cleaning problem.

Table 2.1: Examples for single-source problems at schema level

(showing violated integrity constraints)

Scope/Problem		Dirty Data	Reasons/Remarks
Attribute	Illegal Values	bdate = 30.13.18	Values outside of the domain range
Record	Violated attribute dependencies	age=28, bdate=18.07.96	age = current year-birth year should hold
Record Type	Uniqueness violation	Student1=(“name=j.odero”, p_no=”12345”) Student2=(“name=s.dawson”,”p_no=12345”)	Uniqueness for p_no violated
Source	Referential Integrity violations	Student=(“name=j.odero”, “course=IT”)	Referenced course (IT) not defined

2.2.2 Multi-Source Problems

These problems are as a result of single-source problems which are aggravated on the integration of the sources, since these sources are typically developed, deployed and maintained independently for specific purposes. The main problem of cleaning data from multiple sources is identity of overlapping data particularly matching records referring to the same real-world entity, a problem also referred to as the object identity problem, duplicate elimination or the merge/purge problem. Redundancy from these sources leads to duplicate information which should be purged out and complementing information should be consolidated for a consistent view of real-world entities (Rahm and Do, 2000).

2.2.3 Sources of Errors in Legacy's Transactional Data in KRA

Data cleaning in KRA's context focuses on two critical sets of data (master data and transactional data). Master data refers to taxpayer's bio-data (PIN, contact details, identification number). However, transaction data refers to data that affects the taxpayers' ledger and is derived from the transactions carried out by the taxpayer, hence data cleaning is done in a bid to ensure correct debit/credit amounts in the ledger, debits/credit entries are in the correct tax periods that they relate to and there are correct ledger balances (Domestic Taxes Department, 2018).

Transitioning from the manual systems to the fully automated system (I-Tax) has brought with it a challenge created during the manual and semi manual systems' operations where returns and payments were captured manually by staff, which introduced a lot of errors and inconsistencies due to wrong data capture, missing records and tampering of files (integrity issues) (Domestic Taxes Department, 2018). Concisely, dirty data arises from incompleteness, inaccuracy, inconsistency and duplicity of data sets.

These errors occur mostly on while data is entered while processing returns or assessments, occurrence of credit balances due to overpayment of taxes and wrong posting of receipts to a person or for the year other than the one it was meant for(Domestic Taxes Department, 2010).

2.3 Existing Data Cleaning Approaches

Data cleaning is a significant process for data warehousing and integration with current solutions involving a series of data auditing to find errors, several transformations to fix them and applying the transformations on the dataset (Data extraction, n.d). Several commercial solutions exist for data cleaning; coming in two forms: auditing tools and transformation tools. Auditing tools like Unitech Systems' ACR/Data or Evoke Software's Migration architect are used to audit the data to detect discrepancies. This is followed by a custom script or using the ETL (Extraction/Transformation/Loading) tool e.g. Data Junction or Ascential software's Data Stage to transform the data, fixing errors and converting it to the format needed for analysis. The process of auditing and transformation is iterated severally until the data quality is good enough since data often has hidden hard –to- find special cases (Hellerstein and Rahman, 2001.) The auditing and transformation process however has limitations.

First, it suffers from lack of interactivity since transformation is typically done as a batch process operating on the whole dataset without giving any periodic feedbacks. This leads to a long frustrating wait with the users not knowing whether the transformation is effective since it acts as a black box. (Hellerstein and Rahman, 2001). Secondly, the whole process is user interactive needing significant user effort thus making the cleaning process painful and error-prone.

In view of this, there are several data cleaning frameworks as reviewed in this paper.

2.3.1 Potter's Wheel: An Interactive Data Cleaning System.

Potter's wheel is an interactive data cleaning system that integrates transformation and discrepancy detection in a single interface. In Potter's wheel, user gradually build transformations by composing and debugging transforms (transform denotes a single operation and transformation denotes a sequence of operations). The transforms are specified graphically and their effects are visible immediately on the screen, hence they can be undone if they have undesirable effects (Hellerstein and Rahman, 2001). Potter's wheel automatically infers structures for data values in the background, in terms of user-defined domains checking for constraint violations. The transforms are gradually built as discrepancies are found; hence, the data is cleaned without writing complex programs (Naumann, 2002)

2.3.1.2 Potter's Wheel Architecture

The design goals of the potter's wheel architecture is to eliminate wait time during each step, eliminate complex programming, unify detection and transformation and ensure extensibility. The architecture represented in the figure below depicts the data flow in Potter's Wheel architecture.

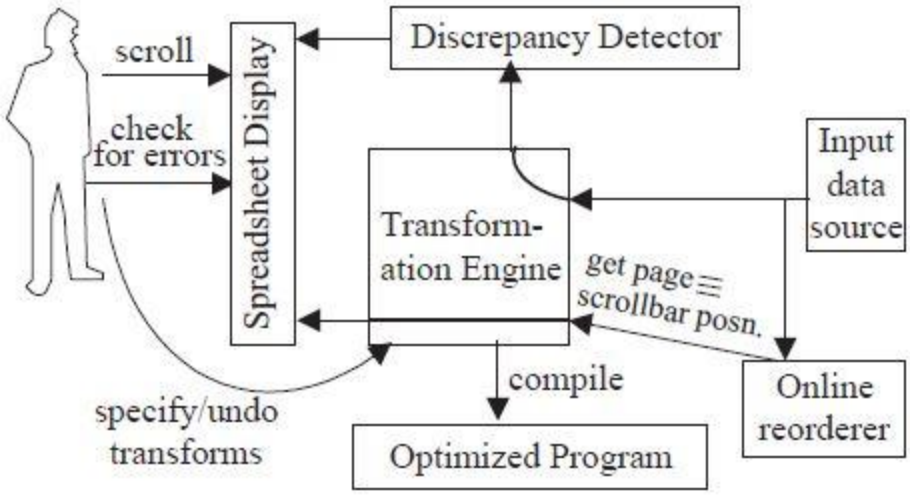


Figure 2.2: Potters Wheel Architecture. Adapted from "Matching algorithms within a duplicate detection system."

The main components of the Potter's Wheel architecture (figure 2.6) are a Data Source, a Transformation Engine that applies transforms along two paths, an Online Reorderer to support interactive scrolling and sorting at the user interface and an automatic Discrepancy Detector (Hellerstein and Rahman, 2001)

Data Source

Potter's wheel accepts input data as a single, pre-merged stream coming from an ODBC source or any ASCII file descriptor (or pipe). The ODBC source is used to query data from DBMSs or distributed sources via middleware.

Each record is viewed as a singlewide column when reading from an ASCII file whereby the user can be able to identify column delimiters graphically and split the record into constituent columns (Hellerstein and Rahman, 2001).

Interface Used For Displaying Data

Data read from the input is displayed on a Scalable Spreadsheet interface allowing users to interactively re-sort on any column, and scroll in a representative sample of the data, even over large datasets (Raman et al, 1999). Whenever the user starts potter's wheel on a dataset, the

spreadsheet interface appears immediately, without waiting until the input has been completely read: a significant action in transforming large datasets or never-ending data streams.

This behaviour is supported by the interface using an Online Re-orderer that continually fetches tuples from the source dividing them into buckets based on a (dynamically computed) histogram on the sort column, spooling them to disk if needed.(Raman V, Raman B and Hellerstein, 1999).

Transformation Engine

Transforms specified by the user is applied in two scenarios. To begin with, they need to be applied when records are rendered on the screen. Using the spreadsheet user interface, this is done when the user scrolls or jumps to a new scrollbar position. Secondly, transforms need to be applied to records used for discrepancy detection, since we should check for discrepancies on transformed versions of data (Hellerstein and Rahman, 2001).

Automatic Discrepancy Detector

According to Muller and Freytag (2000), as the user specifies transforms and explores the data, the discrepancy detector runs in the background applying appropriate algorithms to find errors in the data. The discrepancy detector does this by first parsing values in each field into sub-components according to the structure inferred for the column. The structure of a column is a sequence of user-defined domains inferred as soon as it is formed.

Then suitable algorithms are applied for each sub-component, depending on its domain. For instance if the structure of a column is <number><word><time> and a value is 21 July 12:35, the discrepancy detector finds 21, July and 12:35 as sub-components belonging to the <number>, <word> and <time> domains and applies the detection algorithm specified for those domains (Hellerstein and Rahman, 2001).

2.3.1.3 Structure Extraction

A given value will typically be parseable in terms of the default and user-defined domains in multiple ways. For instance, March 17, 2000 can be parsed as £*, as [*A-Za-z*]* [*0-9*]*;* [*0-9*]*, or as [*achrM*]* [*17*]*;* [*20*]*, to name a few possible structures. Structure extraction involves choosing the best structure for values in a column. Formally, given a set of column values V_1, V_2, …

V_n and a set of domains $d_1, d_2, ..., d_m$, we want to extract a suitable structure $S = d_{s1}d_{s2} \ldots d_{sp}$, where $1 \leq s_1 .....s_p \leq m$.

In general, the inferred structure must be approximate, as the data could have errors in the structure itself. It first starts by a description of how to evaluate the appropriateness of a structure for a set of values then look at ways of enumerating all structures so as to choose the best one (Hellerstein & Rahman, 2001).

2.3.1.4 Interactive Transformation

Specification of the data cleansing process is done interactively since immediate feedback of performed transformations and error detection enables the users to gradually develop and refine the process as further discrepancies are found (Hellerstein and Rahman, 2001). Potter's wheel supports interactive transformation by enabling users to construct transformations gradually adjusting them based on continual feedback in order to achieve ease of specification, ease of interactive application and ability to undo transforms and track data lineage (Chen, et.al., 1993).

2.3.1.5 Critique Of Potter's Wheel Framework

Strengths

Potter's wheel allows users to gradually build transformations to clean data by adding transforms through graphical operations and examples and see the effect immediately hence allowing easy experimentation with different transforms due to its interactive nature. Secondly, string parsing using structures of user – defined domains result in a general and extensible discrepancy detection mechanism. These domains provide a powerful ground for specifying Split transformations via example values. Also end user has a lot of power since there are graphical representations of transformations on data such that users do have a chance to add or even undo effects on transformations.

Weaknesses

Scrolling by the end-user is directly related to the scope of the automatic discrepancy detection, meaning; if the end user does not scroll up to that record, a particular discrepancy may be available but won't be detected. Secondly, the sequence of the data, fetched in samples of the source data; remains unknown, thus may lead to undetected errors in the data. Thirdly, there is a possibility of the same sampled data being fetched and retested continuously hence leading to time wastage. Since greater time will be used in detecting discrepancies within the data warehouse. Finally, an interactive querying system needs to be used to measure the effectiveness of the user interface.

2.3.2 IntelliClean: A Knowledge-Based Intelligent Data Cleaner

This is a knowledge-based Intelligent Data cleaner. It proposes a knowledge-based framework for effective data cleaning implementing existing cleaning strategies; employing a new method to compute transitive closure under uncertainty which handles the merging of groups of inexact duplicate records. This framework can identify duplicates and anomalies with high recall and precision (Lee, et.al, 1999).

The Intelliclean model receives "dirty" datasets having different errors, wherein cleaning strategies are applied to the dataset with the objective of obtaining consistent and correct data as the output.

Pre-processing dirty data prior to the data cleaning process leads to more consistent data and better de-duplication. The two metrics that benchmark the effectiveness of data cleaning strategies include Recall Also known as percentage hits. It is defined as the percentage of duplicate records being correctly identified. Secondly, False-Positive Error: This is the antithesis of the precision measure, sometimes referred to as false merges. It is defined as the percentage of records wrongly identified as duplicates, i.e:

$$No.\,of\ wrongly\ identified\ duplicates \div Total\ no.\,of\ identified\ duplicates\ \times 100\%$$

The intelligent knowledge-based framework provides a systematic approach for representation standardization, duplicate elimination, anomaly detection and removal in dirty databases (Mydanchik, 1999).

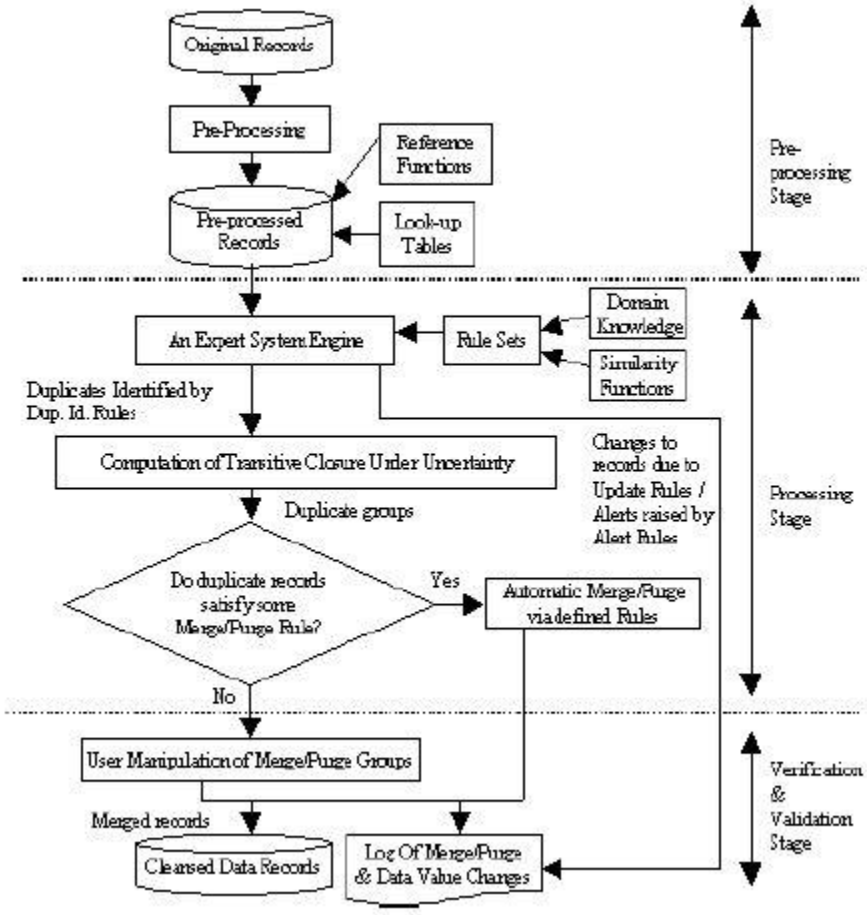


Figure 2.3 : A Knowledge-based Cleaning framework. Adapted from "Data Cleaning: Problems and Current Approaches"

The framework consists of three stages, namely: Pre-processing Stage: Data records are first conditioned and scrubbed of any anomalies that can be detected at this stage. Data type checks and format standardization can be performed (e.g. 12/02/2018, 12th February 2018, can be standardized to one format). Inconsistent abbreviation used in the data can be resolved at this stage (e.g. occurences of '1' and 'A' in the sex field will be replaced by 'Male' and occurrences of '2' and

'B' will be replaced by 'Female'.) The output of this stage will be a set of conditioned records which will be input to the processing stage (Lee et, al., 2000)

Secondly, processing stage: Conditioned records are next fed into an expert system engine together with a set of rules. Each of the rules will fall into one of the following categories: duplicate Identification Rules, Merge/Purge Rules, Update Rules and Alert Rules.

Finally, validation and verification Stage: Human intervention is required to manipulate the duplicate record groups for which merge/purge rules are not defined. The log report provides an audit trail for all actions and the reasons for actions made during the pre-processing and processing stages (Lee et, al., 2000).

2.3.2.1 Critique of IntelliClean Framework

IntelliClean framework has its own strengths which are considered hereunder:

Strengths

The introduction of an expert system within the framework makes it much more generic since the system is able to learn, hence changes made in the textual databases will be detected and identified as a transformed data automatically by the framework. This makes the framework more efficient than other frameworks.

Secondly, it provides effective rules for resolving the recall-precision dilemma which is a shortcoming in other frameworks, thus it enhances the effectiveness of the framework.

Weaknesses

It only considers textual forms of data in the data warehouse, implying that duplicates of data of other formats will not be identified, for instance: image, video, graphics and other file formats are not applicable with the IntelliClean.

Secondly, data source from the web is not applicable to the IntelliClean system hence there is need for a generic and data source independent system to be developed to resolve the problem.

2.3.3 ARKTOS: A Tool For Data Cleaning and Transformation in Data Warehouse Environments

ARKTOS is a framework capable of modelling and executing the Extraction-Transformation-Load process (ETL process) for data warehouse creation. Data cleansing is considered an integral part of the ETL process, which consists of single steps that extract relevant data from the sources, transforming it to the target format and cleanse it, then loading it into the data warehouse (Vassiliaadis et al, 2001).

A meta-model is specified allowing the modeling of the complete ETL process. The single steps (cleansing operations) within the process are called activities wherein each activity is linked to input and output relations. An SQL statement declaratively describes the logic performed by an activity; where each statement is associated with a particular error type and a policy specifying the behavior in case of error occurrence.

According to Muller and Freytag (2000), six types of errors can be considered within an ETL process specified and executed in the ARKTOS framework: PRIMARY KEY VIOLATION, UNIQUENESS VIOLATION AND REFERENCE VIOLATION; which are special cases of integrity constraint violations. The elimination of missing values is handled by the error type: NULL EXISTENCE. The remaining error types are DOMAIN MISMATCH and FORMAT MISMATCH referring to lexical and domain format errors.

The policies for error correction simply are IGNORE, but without explicitly marking the erroneous tuple, DELETE as well as WRITE TO FILE and INSERT TO TABLE with the expected semantics. The last two providing high user interactivity. Eventually, the success of data cleaning can be measured for each activity by executing a similar SQL statement counting the matching/violating tuples (Panos, et.al, 2000).

2.3.3.1 Critique of ARKTOS Framework

Strengths

The ARKTOS is capable of modelling and executing practical ETL scenarios by providing explicit primitives for the capturing of common tasks (like data leaning, scheduling and data transformations).

Also, the system makes it easier for authoring by providing three ways to describe an ETL scenario: a graphical point-and-click front end and two declarative languages: XADL (An XML variant), which is more verbose and easy to read and SADL (an SQL-like language) having a quite compact syntax.

Weakness

No current solutions exist to optimization problems (Identification of a small set of algebraic operators, Local optimization of ETL activities and, Global (multiple) optimization of ETL activities).

2.3.4 AJAX

Ajax is an extensible and flexible framework that tries to separate the logical and physical levels of data cleaning. The data cleaning workflow and specification of cleaning operations are supported in the logical level, while the physical level supports their implementation. (Galhards et al, 2000).

According to Galhards et al (2000), AJAX's main goal is to facilitate the specification and execution of data cleaning programs, either for a single source or integrated multiple data sources. However, this paper focuses on data from a single source and transforming it into the intended data, to be input into the system. Ajax focuses on five major transformations including: mapping, viewing, matching, clustering and merging. The mapping transformation standardizes all data formats simply producing a more appropriate data format by applying splitting and merging operations. Matching compares several records to find pairs matching the specified criteria – a process that is highly going to be implemented in this project to compare the taxpayers' records with the input data already existing in their individual records.

In addition to this, AJAX also has another transformation known as clustering groups together matching pairs with high similarity by applying a given group criteria, hence the merging transformation is then applied to each of these clusters in order to eliminate duplicates or produce a new record based on the integrated results.

Ajax is more complex than other approaches.

2.3.4.1 Critique of AJAX

Strengths

Its extensible nature allows customization. For instance, extension of functions from other libraries, combination of primitives with SQL, all of this adding dynamism to the functionality of AJAX. It also has a high system interactivity.

Weaknesses

It highly depends on a human expert to resolve exceptional cases arising as a result of executing a micro-operator. This makes it less attractive as compared to IntelliClean system which having an expert module dealing with such exceptional cases.

Secondly, it uses a very complex approach in creating quality data as compared to other frameworks, hence takes greater cleaning time in cleaning data as compared to the other frameworks.

From the foregoing, a comparative analysis of the four data cleaning approaches based on four parameters i.e :Interactivity, Data format/structure, Human Dependency and Maintenance, reveals that there is high interactivity with the end user and a high human dependency except for the intelliclean framework which uses an expert module embedded in the system. However the maintenance aspect was not considered. This means that the four approaches might not be appropriate for this project as we would like to eliminate high user interactivity by automating the human interactive tasks via a robotic process, and we would like to work with processing and manipulation of figures (discrepancy detection and auditing) rather than texts.

The comparative analysis is summarized in the table below:

Table 2.2: Comparative Analysis of Data Cleaning Frameworks

Parameter	Porter's Wheel	AJAX	IntelliClean	ARKTOS
Interactivity	Very Interactive, hence easy to use.	Complex interface hence unfriendly to non-technical persons.	Interactive with end user, however requires little input from end users.	Highly interactive, has graphical interfaces for loading and executing validations on loaded files
Data Format/Structure	Text	Text	Text	Text
Human Dependency	High human dependency for exceptional errors	High human dependency, for instance evaluation and validation of errors is fully dependent on human expert.	Very minimal because of the expert module embedded in the system	High dependency on human experts for error correction even though it has complex modules for dealing with duplicates
Maintenance	Not Considered	Not considered	Not considered	Not Considered

2.4 Existing Data Cleaning Algorithms

2.4.1 Alliance Rules algorithm

According to Rajiv, et.al, (2009), this algorithm addresses the duplicity error of string data type (name field) and uses the algorithm of de-duplicity in the name field of the data warehouse- where the data warehouse is formed by merging data from different sources(data marts) having different field formats. It involves the following steps:

i. Preprocessing

At this point the strings in the name field are converted into a numerical value which is stored in another file called score; for reference. The integer values are called scores of the name. The string is converted into values using relation:

[((radix)ˆplace value)×face value]mod m

Where the number of words in a name defined as N, e.g Joe Xavier has N=2, and the total number of scores would be N+1

- Radix is 27 characters(26 alphabets and '.')
- Face value is the sequence of occurrence of characters in the world of alphabets starting with 0-a- -25-z and 26-(.)
- The place value is marked from right to left starting from 0
- M is any large prime number and
- Letters are case sensitive

ii. Alliance rules application

In this step, the two data marts are considered such that a name from DM1 is to be checked and marked for duplicity with all the names in DM2.

iii. Detection of errors

Errors in the name are evaluated using the concept of q-grams testing, where the q-grams are the substring of a given name string. The length of the substring can be of any value smaller than the name of the name string itself.

2.4.1.2 Limitations of Alliance Rules Algorithm

It specifically deals with study and detection of string data type errors. It also focuses on the 'name' string format only neglecting all other formats. It cannot detect duplicity error efficiently in some cases; hence need another field as a reference. If the other field for reference is incorrect or blank, then the data cleaning process could stall. Manual work required in the preprocessing phase for calculation of scores, hence error-prone.

2.4.2 HADCLEAN Algorithm

This algorithm deals with detection and cleaning of spelling errors- where the usage of standard dictionary is commonly used for the data cleaning process. However, many organizations have different terms used by them; different from others; which serve as jargons, hence the organizations use organization specific dictionary (Yan, 2008).

Transitive closure algorithms can be used in the case whereby there exists records with blank fields.

The steps involved include:

I. PNRS (Personal Name Recognition Strategy)

Used to correct the phonetic and typo errors using the standard dictionaries and it employs two strategies:

i. Near miss strategy: addresses the errors in the words which are nearly missed and shows errors. This is done by inserting a blank space for instance: co-education, by interchanging

two letters e.g.:'rgreen' with 'green', by changing/adding/deleting a letter. All these actions are taken with the help of reference to standard dictionaries.

ii. Phonetic algorithm: Applies the concept of phonetic codes which is calculated for every word and then it is matched with the phonetic code of the standard dictionary. Helps to detect errors for words like 'seen' and 'scene' which sound same (phonetic) but have different meaning and different phonetic codes respectively.

iii. The modified PNRS-some organizations involve the usage of jargons and often have their organizations designation in regional languages, for such situations an organization specific dictionary can be used as a reference.

II. Transitive Closure

Transitive closure uses attribute keys to group the records and math them as a set of related records and then errors are detected and corrected after detailed analysis. This aids in filling the blanks cells (fields) and remove the duplicity errors and redundancies.

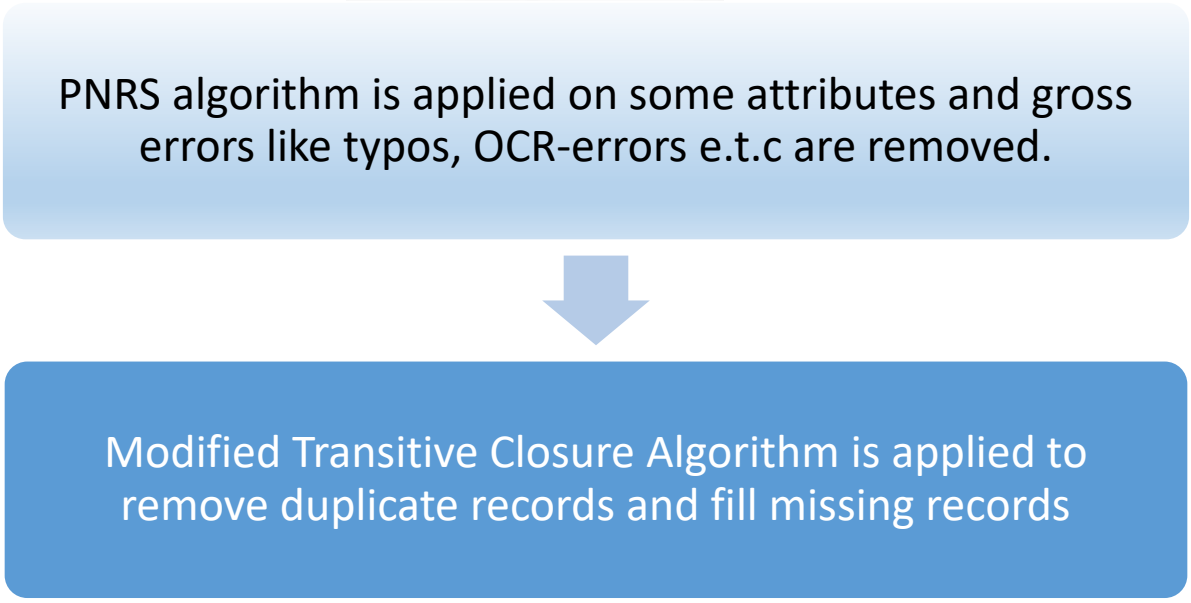


Figure 2.4 Summary of the HADCLEAN ALGORITHM

The flowchart above summarizes the working of the HADCLEAN algorithm. Starts with the PNRS strategy followed by modified transitive closure algorithm.

2.4.2.1 Limitations of The HADCLEAN algorithm

The algorithm is data specific due to the prioritization to the attribute keys (Arindam, 2012). The modified transitive closure algorithm has some specifications defined in order to combine the records as related. The rules being strict, sometimes we are not able to combine the records even if they are related because it has only one secondary key and two tertiary key matches (Deepali and Sonal, 2013). Semantic Data Matching Principles have to be applied to the data in order to get better results in situations where the records have values : Mumbai' and some 'Bombay'

2.4.2 Comparative Aspects of the Algorithms.

The following table summarizes the analysis of the two algorithms.

Table 2.3: Comparison of data cleaning algorithms

Factors	Alliance Rules	HADCLEAN
Approach	Inter-related	Hybrid
Steps Required	Pre-processing, Alliance rules, Detection and Q-gram	PNRS and Transitive closure
Strategy	Scores calculation and comparison	Uses dictionary based approach and comparison using phonetic codes
Dependency	Pre-processing forms the base of the other steps, alliance rules is applied on the basis of the scores obtained	The steps are independent of each other, yet they are used a s an input to another with the intention of hybrid approach.
Complexity	Since the algorithm involves dependencies in the steps performed, complexity is bound to be greater as compared to HADCLEAN.	The independency of the steps performed in the algorithm makes it relatively less complex. Nevertheless, the hybrid approach makes the algorithm slightly more complex.

	In addition to this, mathematical calculation in the pre-processing stage increases the complexity.	
Types of dirty data addressed	Misspelled data Duplicate data	Nearly misspelled data, phonetic errors, and typographical errors.
Accuracy	Far better because it uses absolute matching(q-grams)	Approximate because transitive closure uses the rules of keys matching to group records
Ease of Implementation	The huge mathematical calculations to define scores makes it difficult.	Less calculations are involved hence less easy.
Application	Used in customer oriented data warehouse	Organization specific data warehouse
Output	Error is identified and detected	Error is detected and corrected
Drawback	Many calculations involved	The dictionary based approach for PNRS works only for English Language.

2.5 Data Cleaning Process

According to Deepali and Sonal (2013), data cleaning is viewed as a two-step process involving detection, then correction of errors in a dataset.

The steps involved are as follows: Identification of errors/dirty data: The source records/data could have incomplete, missing or corrupted data. Perform Error Verification: Confirmation, whether an error is valid or not, especially where the institution uses lots of organizational jargons. Extract the data to be cleaned: Extraction and storage of data in a temporary table, operations are performed and the data is repaired and verified, then replaced in the target table. Perform data cleaning: A process which can be manual or automated.

However, the manual process is highly time consuming and tedious in nature. Human beings have limited capabilities in speed, accuracy in error detection and correction, hence leading to more

error prone performances and degrading data quality, thus leading to increase in operational costs and poor decision making.

2.5.1 Data cleaning Process in KRA

The main objective of data cleaning is to have correct and accurate transactions posted in the I-tax and legacy systems as this will give a clear and correct picture of the taxpayer's affairs in terms of whether they are in debit, credit or nil position, and to verify that all returns and payments are captured, and verify that all the credits in the system are correct (Domestic Taxed Department, 2018).

The ledger movement transactions that may affect the amount posted on the taxpayer's ledger and bring about dirty data can be classified as: data entry error correction, credit movements, and reversal of wrong postings of receipts (Domestic Taxes Department, 2010).

To start with, data entry error correction occurs due to an omission or commission when data is entered while processing returns or assessments. On confirmation of the error; data correction form for the income tax is completed with full information of the error and the correction to be made. The data correction form is signed by the officer, checked by the program manager and approved by the station manager. Approved data correction form is forwarded to the returns processing unit. Return or assessment is opened at the return processing unit on evaluation of the correction recommendation, and finally the return processing unit makes the correction, and reconfirms the return or assessment (Domestic Taxes Department, 2010).

In the case of data correction of errors arising from the case of credit movements – which occur on overpayment of taxes and the taxpayer may seek to utilize such overpayments by moving the balances to other years that have debit balances. Overpayments may genuinely occur when instalment taxes are paid but when the final accounts are prepared, the taxpayer has overpaid taxes, or when an audit is carried out and a manual assessment issued and on raising the assessment in the system, the manual assessment figures are higher due to losses carried forward from the previous year and finally when an audit case goes to the court and a ruling is made in favour of the taxpayer, while the taxpayer had paid (Domestic Taxes Department, 2010).

If any of the three scenarios occurs, the taxpayer will request for transfer in writing to the commissioner. On receipt of the letter the payment is validated and if confirmed a receipt for transfer is identified (preferably the latest payment). The taxpayer is requested to submit the original receipt to be transferred or to be split, a form COLL 10(Appendix 1) is then completed in duplicate to recommend movement of the overpayment year to the underpayment year and shall be approved by the debt manager. Approved form COLL 10 is forwarded to the chief cashier who evaluates the recommendation and if confirmed effects the transfer in the system ledger (Domestic Taxes Department, 2010).

Eventually, for the case of wrongly posted receipts, a taxpayer must write a letter to request for reversal pointing out the error and how it occurred. Officer will complete COLL 10 in duplicate, quoting all details. COLL 10 is approved by the program manager or station manager, then the approved COLL 10 is forwarded to the chief cashier, who evaluates the transfer and re-opens the receipt, the receipt is then transferred to the correct Pin or date and the transfer is re-confirmed (Domestic Taxes Department, 2010).

From the foregoing, it is evident that the existing data cleaning processes are manual, time consuming since they involve all levels of management's confirmation to validate the error, and back and forth the between the taxpayer and the officers in charge, which can take a number of days before one account is fully cleaned. This can in turn lead to more errors in the process, hence inefficient and high operational costs involved. The business processes are so bureaucratic, thus the need for a faster robotic process that is going to ensure utmost accuracy and integrity is upheld, hence high quality data and better decision making.

2.6 How R.P.A works

Some of the attractive RPA key features include its technology agnostic capability enabling it to work across legacy ERP's, mainframes, custom applications, and any technology that can be utilized by a human can be navigated by an RPA robot (Price Waterhouse Coopers, 2017) .

Secondly it's non-intrusive, since it leverages other application software through the existing application's interface, hence not technically integrated. RPA programs can be launched in a matter of days or weeks resulting in low cost of implementation and high return on investment

(Price Waterhouse Coopers, 2017). Its scalability and traceability makes the bots subject to full audit with visibility to security access and modifications (Price Waterhouse Coopers, 2017).

RPA can be easily deployed and managed from a central controller to interact with a wide range of business applications. To start with, we have process developers who specify and publish detailed instructions for robots to perform. Secondly we have the robot controller used to assign jobs to robots and monitor their activities. Thirdly, the robot (virtual or physical) which interacts directly with business applications. Fourthly, business users who review and resolve any exceptions or escalations and eventually the applications which robots are capable of interacting with (Deloitte, 2018).

2.6.1 RPA Architecture

The RPA architecture is a combination of several tools, platforms and various infrastructure elements to form a complete RPA tool. The blocks available in the RPA solution is as follows:

Applications under Robotic Process Execution: well suited for enterprises applications like ERP, SAP or any other record processing application. These applications are data intensive and they are loaded with repetitive tasks.

RPA Tool: To develop software robots to automation of applications in Desktop, Web and Citrix environment. Exception handling, ability to write to/from various data sources and to build reusable components.

RPA Platform: RPA Software bots can be stored in a shared repository and they can be shared across software robots libraries. RPA platform has the ability to develop meaningful insights on the bots and execution process.

RPA Execution Infrastructure: They act as a bank of parallel physical or virtual lab machines which is controlled based on usage patterns. Machine scale up or down in parallel to achieve the automation can also be performed.

Configuration Management: Updating of bots to newer version is performed. Branching and merging of RPA bots is also performed since they are reusable across the libraries.

The various layers of applications and tools that makes the whole architecture is summarized as below:

Table 2.4: RPA Architecture

Layer	**Purpose**	**Benefit**
Process	• Business rules. • Hand off point. • Prioritization if not in management control	• Focus on business rules without needing to create links • Simplify changes
Sub Process	• Reusable business logic • Identity • Verification Reconciliation	• Reusability • Avoid multiple changes in process when logic changes.
Object	Procedures for performing specific tasks E.g log on, enter name, password	• Reusability within systems • Development does not require business rule understanding
Component	Individual screen interaction Eg. Enter address in Line 1	• Lower risk, Faster changes • Target application integration can be changed without a risk of changing business.

2.7 Conceptual framework

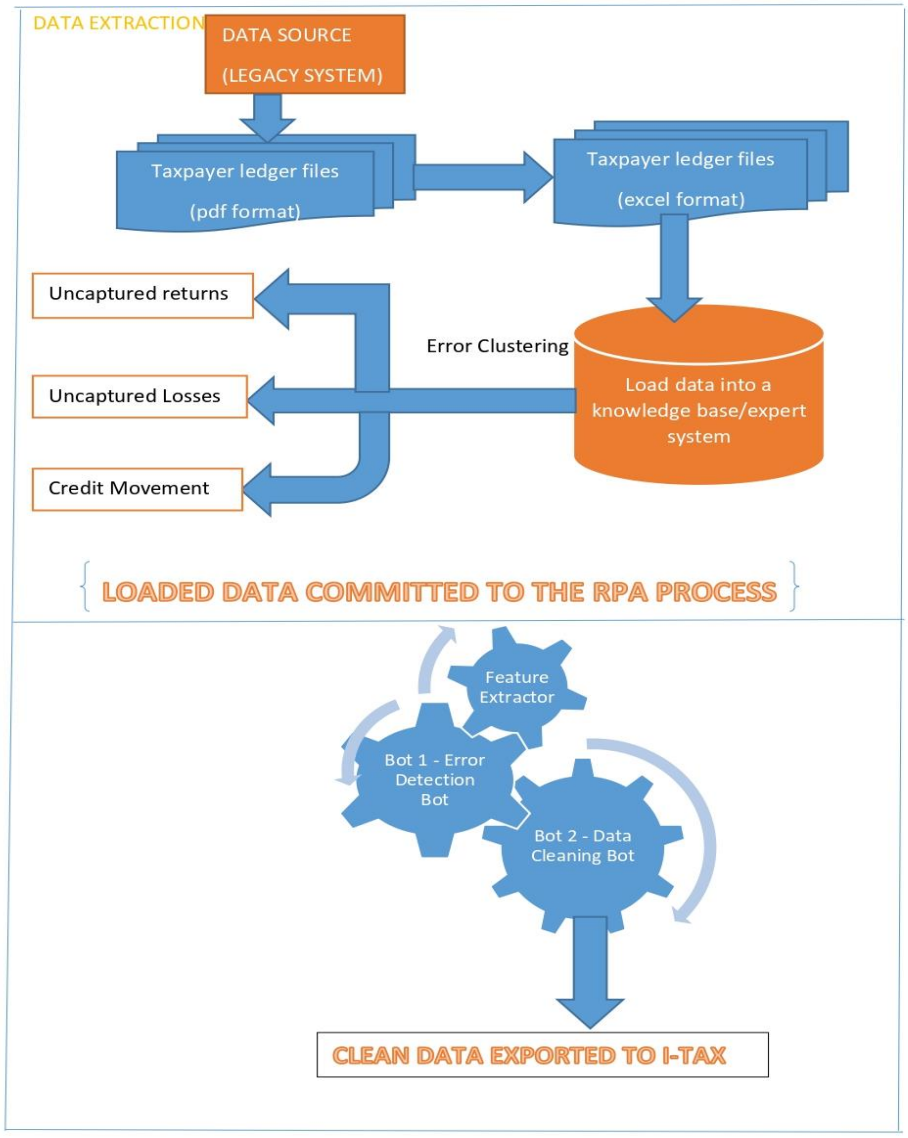


Figure 2.5: Conceptual Framework

The conceptual framework describes how the different input variables are manipulated to get an output: in our case its clean ledger data to be exported to i-Tax.
The system accepts ledger files in excel format as input- which have already been converted from pdf. The files are then loaded to the bot, which does feature extraction; to get the desired features of the ledger to be used in detection of the errors.

These features are then loaded to the expert system module in form of answers to questions, by the bot. After error detection, the second bot undertakes the data cleaning process by using via RPA's process and object studio.

CHAPTER 3:

RESEARCH METHODOLOGY

3.1 Introduction

Research can be defined as the process of systematically solving problems (Bhatnagar & Singh). This section described the main research methodology that was adopted in carrying out this study. It highlighted the research design implemented, population studied, sampling technique and sample size, instruments used in data collection and procedure, pilot tests, data analysis and presentation.

This research methodology was guided by the researcher's objectives and the research approaches that were used and reviewed under the literature review section.

3.2 Research Design

According to Cooper and Schindler (2006) research design can be described as the structuring of data collection and analysis in order to meet the research objectives through empirical evidence economically. Thyer (1993) describes it as a blueprint describing how a research study is to be completed: operationalizing variables so that they can be measured, selecting a sample of interest to study, collecting data to be used as a basis for testing hypotheses, and analysing the results.

This research took an experimental design approach, involving the identification of research objectives, building the RPA system; through identification of the features of interest in the ledger files, manipulating them to identify the errors in the ledger by passing the feature results through the Knowledge-based system to introduce intelligent process automation to the entire system. After clustering of the errors for the appropriate data cleaning action, the model was validated using a number of experiments to ensure optimal performance was achieved.

3.3 Location of the Study

The study was located in Kenya Revenue Authority, Domestic taxes Department's data cleaning section. This is since we were studying the business processes of data cleaning and used the ledger files of companies PINs (Personal Identification Numbers) to test the intellibot in clustering and cleaning of the errors.

3.4 Target Population

Bryman (2012) defines population as the total number of units in a study environment from which a sample may be selected. KRA's corporate Income Tax ledger Personal Identification numbers (PINs) was chosen as the population of the study.

3.5 Sampling

Purposive sampling was to be applied in the research, by choosing the ledgers of the PINs which have the features of interest suitable for the study. These include ledgers having credit or debit balances as well as data capture errors. The population of the study was all corporate PINs countrywide but the sample size was limited to only ten Income Tax ledger PINs, after which the results thereof could be replicated to other "dirty" ledgers.

3.6 Data Collection

Document review was used as a data collection method to study the features inherent in the ledgers, and identify the ledgers appropriate for sampling in order to determine the correct data cleaning procedure on the corporate income tax ledgers. In addition to this, participant observation of the processes involved in the manual data cleaning process was inevitable as this ensured that the routine rules-based tasks which are to be done by the bots are well understood and implemented, notwithstanding, the need for a better understanding of the business rules to be used in developing the knowledge-based system for clustering of the errors in the ledger.

3.7 Data Analysis

The features extracted from the corporate taxpayer ledgers were used in clustering the ledger depending on the error type(s) inherent in the ledger and analysed using python's data analysis tools implementing the matplotlib library for a graphical representation of the comparative results of the data cleaning action and time taken to complete the process.

3.8 System Development Methodology

The prototype was developed using the Rapid Application Development (RAD) system development methodology which helps in creation of applications within a limited period of time (Naz & Khan, 2015).

3.8.1 Phases of RAD

The RAD process is divided into four distinct phases according to the James Martin approach as shown in figure 3.2 (Orawit, 2006).

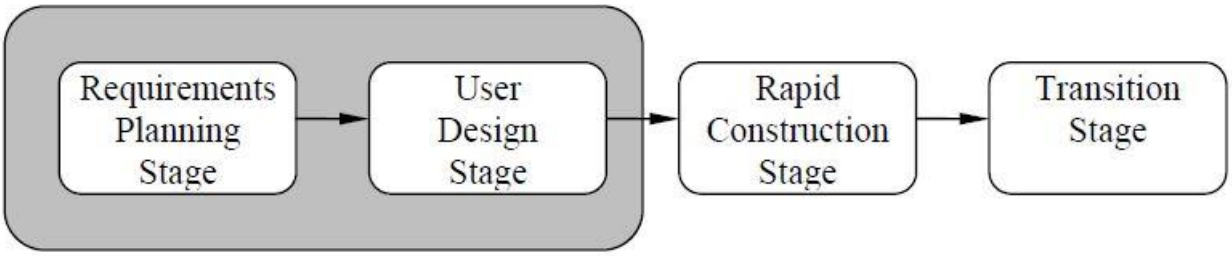


Figure 3.1: Phases of RAD: Source: Adapted from "SDLC RAD MODEL"

3.8.2 System Analysis Phase

In this phase, the requirements of the intellibot for data cleaning were obtained through participant observation to gain a general understanding of the business process functions guiding the data cleaning process in KRA. In addition to this, document reviews were also used to gain an insight of the different features depicting errors inherent in the taxpayer ledger files.

3.8.3 System Design

In this phase the structure and architecture of the prototype was designed. Unified Modelling Language (UML) diagrams designed to show various components and aspects of the system including use case diagrams, context diagrams, data flow diagrams and sequence diagrams.

3.8.4 System Implementation Phase

After completion of the design phase of the proposed system, the Intellibot was implemented using Blue Prism's enterprise RPA platform, and python as the development language using the pycharm IDE. Two different process studios for the error clustering and data cleaning bots were created, synchronized to the various visual business objects. The Knowledge based system, which provided the clustering results, was developed in python implementing the pyknow library and later bundled as an executable file that could be launched independently using the PyInstaller. After implementation, the test results were validated and presented graphically implemented by the Matplotlib library for visual representation.

3.8.5 Robotic Process Automation Tools

The Blue Prism Enterprise RPA platform was chosen since it's designed with enterprise IT architecture, security and compliance requirements in mind. UiPath is another alternative which can be used to implement RPA but not quite user friendly.

3.8.6 Programming Tools

Python was used as the programming language for the knowledge-based system to be used in sync with RPA since it has useful libraries for artificial intelligence and machine learning for instance PyKnow, PyInstaller – which provides suitable modules for development of the intellibot.

3.9 Research Quality

The performance of the RPA model was evaluated experimentally through integration with the I-Tax platform (Feldman & Sanger, 2007). The research results were subjected to tests of reliability and validity, wherein it was confirmed that the intellibot was able to provide consistent, precise and dependable results, when the error detection and cleaning process was repeated over again over different circumstances and platform, hence the system was proven trustworthy.

In addition to this, the intellibot results were proven valid, since it provided results, which were correct as compared to the results of the manual data cleaning process, hence it measured what it was intended to measure.

3.10 Ethical Considerations

This research ensured that all the ethical guidelines were strictly adhered to. The interview process was done after the participants granted informed consent, as it is one of the major hallmarks of modern ethical research.

In addition to this, no fabrication or falsification of data was entertained since all cited authors are acknowledged to avoid plagiarism and give credit for their work.

CHAPTER 4:

SYSTEM DESIGN AND ARCHITECTURE

4.1 Introduction

This chapter presents the overall architecture and detailed design and analysis of the proposed system by incorporating the various requirements. An intellibot that is synchronized to a knowledge based System to cluster the errors in readiness for the data cleaning process. The Knowledge based System was developed in python programming language to aid in intelligent process automation of the bots.

The basic parameters required as input to the system comprised ledger files that are read in excel format and the data therein loaded by the bot into the knowledge base. The output from the Knowledge based system is loaded to another bot as a collection depending on the type of error(s) detected. This bot undertakes the data correction process of the error(s) identified in line with the business processes thereof.

This chapter further analysed both functional and non-functional requirements, the design of the proposed system will incorporate UML diagrams to describe the overall architecture of the system and give detailed description of the various components of the system. Use case diagrams with detailed use case descriptions, sequence diagrams, context diagrams and data flow diagrams.

4.2 Requirements Analysis

This research aimed at developing a Robotic process Automation system for data cleaning. Based on this objective, the section underneath outlines the various requirements for the proposed solution. The requirements were mainly gathered through participant observation of the researcher, and in the daily duties undertaken which were majorly repetitive and rule-based tasks.

4.2.1 Functional Requirements

These are important aspects of the data cleaning system that must meet the user specifications and needs, they refer to the functionality of the system; the services that the intellibot data cleaner provides to the user. These include:

i. Access the blue prism main interface through user login
ii. The bot accesses the ledger file stored in the computer in excel format
iii. The bot inputs the features of interest from the excel sheet to the knowledge base which identifies the type of error.
iv. The bot selects the cleaning process depending on the type of error identified
v. The bot presents the clean data in excel format ready to be exported into i-Tax system.

4.2.2 Non-Functional Requirements

According to Bruade (2001), these are constraints the system must work within; hence the following constraints must be taken into consideration.

4.2.3 Usability

The system is intended for use by the data cleaning team and they should be trained so that they can understand their interaction with the bots, but actually, the system is intended to eliminate redundant tasks, hence it will only need a lean skilled staff operating the bots after its full deployment.

4.2.4 Scalability

Any increase in the number of records in the ledger should be accommodated by the system. In addition to this, since the scope is limited only to income tax company returns, the system should also be able to clean individual and VAT returns in future.

4.2.5 Persistent Storage

The blue prism platform should be properly configured to the MSSQL server which provides the database connectivity to the relational database created by the bots during reading of the excel files and data cleaning process.

4.2.6 Security

The system ensured user passwords are encrypted, to restrict unauthorized access, accidental or unintended usage and provide access only to legitimate users. In addition to this, different access levels were created for the different system users by the super admin.

4.2.7 Hardware Requirements

The major hardware requirement is an efficient network system if two or more computers are operating multiple bots to perform different tasks seamlessly as well as a dedicated blue prism server and MSSQL server common in a production environment. The computers in the network should be above 1.8 GHz (Dual or Quad core processors and above) and a minimum of 4 GB RAM (8 GB recommended) especially if the host operates virtual machines.

4.2.8 Software Requirements

The main software requirement is a 32-bit or 64-bit operating system. Currently the blue prism platform works only on windows and works only with internet explorer for bot processes on web applications. The system needs an installation of the .NET frameworks, MSSQL server and Blue prism platform installation.

4.3 System Architecture

The proposed system architecture outlines the general layout of the RPA system. The major steps that take place in the multi-bot system is as follows:

- The ledger file to be cleaned is presented in excel format and loaded into the new process studio of the first bot.
- Desired features which are needed by the KBS to identify the errors are extracted by the bot from the excel file.
- These features are fed into the rule-based KBS system that implements IF-THEN rules and accepts inputs in form of answers to the questions it poses on the features of interest in the ledger files.
- The output from the KBS identifies the errors in the ledger files

- The second bot takes over the cleaning process depending on the classification of the error identified i.e: data entry error due to omission or commission, credit movements, or error due to uncaptured receipts.

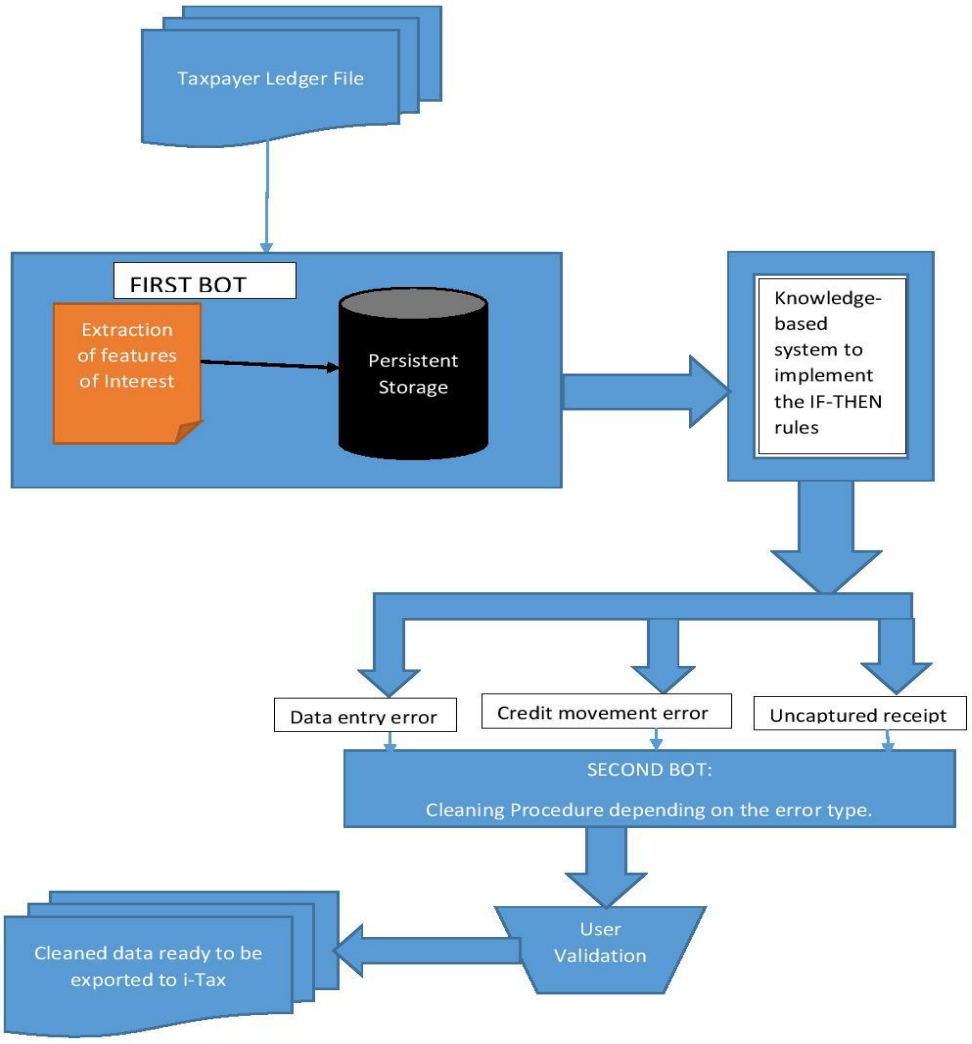


Figure 4.1: RPA-KBS System Architecture

4.4 Use Case Diagram

Use case diagram shows the interaction between various actors and the system. The Use Case diagram below illustrates interactions between the various actors and the RPA data cleaning system. It also depicts the functionality that the proposed system will have.

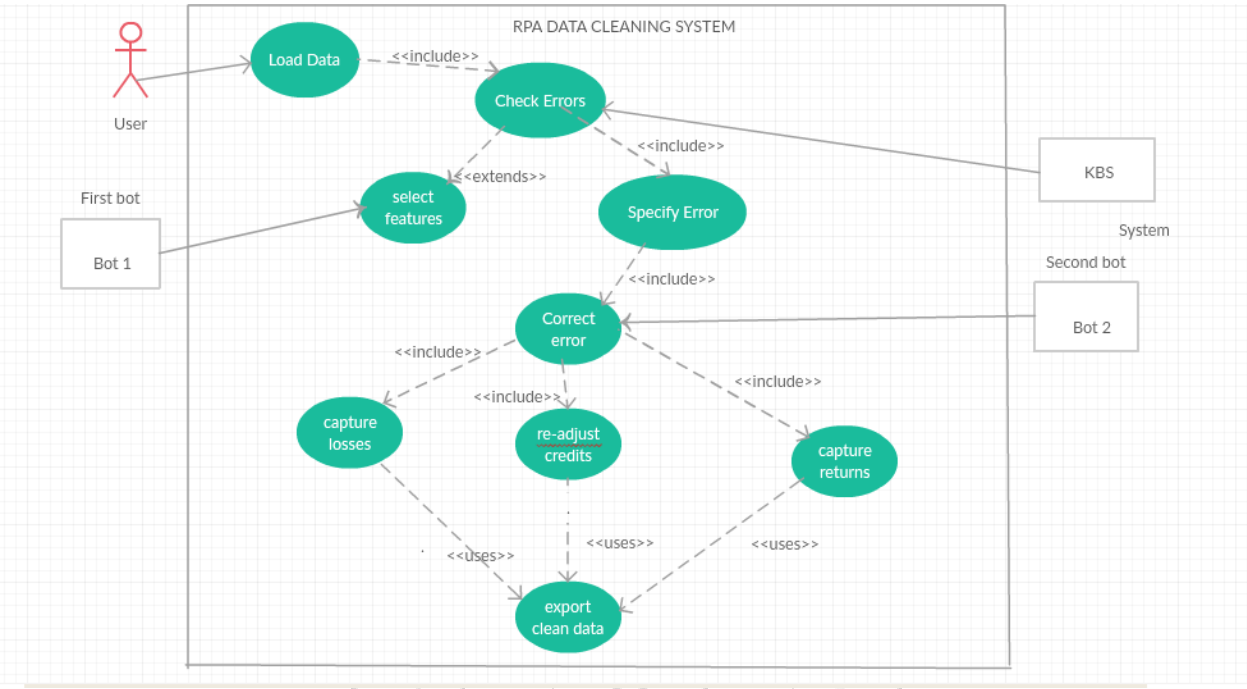


Figure 4.2: Use case diagram

4.4.1 Detailed Use Case Descriptions

Comprehensive description for the use cases above is provided in this chapter in a two-column fully dressed format.

Use Case: Load Data, Check Errors.

Primary Actors

User

KBS

Preconditions

Load Data use case completed successfully

User successfully logs in to launch the first bot

Post conditions

KBS classifies the error in the ledger into the specified error types for data correction

Main Success Scenarios

Actor Intention	**System Responsibility**
1. User logs in to launch the first bot	
	2. Pass the credentials and the instructions to launch the first bot
	3. Loads the ledger in form of excel sheet.
	4. Bot extracts features of interest from the data in the excel sheet.
	5. IF-THEN rules from the KBS are used to classify the error type
6. View the classification of the error from the ledger.	

Use Case: Correct Error

Primary Actors

System

User

Preconditions

Ledger error type classification was done successfully.

Post conditions

Data from the ledger cleaned successfully.

Main Success Scenarios

Actor Intention	**System Responsibility**
1. Error type classification triggers launching of the second bot.	2. Appropriate data cleaning method depending on the cleaning process. 3. Transforms the figures of the data. 4. Presents the clean data.

Use Case: Export clean data

Primary Actors

User

System

Preconditions

Successful cleaning of the data

Post Conditions

User views the clean data ready to be exported.

Main Success Scenarios.

Actor Intention	**System responsibility**
1. Second bot triggers export of data.	2. Return the clean data in excel ledger format. 3. Display the system log including the time taken.
3 Exit the system	

4.5 Sequence Diagram

The sequence diagram shown below illustrates the relationship between the user and the proposed system as well as the interactions among the various internal components of the system. The user loads the "dirty" ledger to be cleaned through the object studio of the first bot in form of an excel file. Once the ledger is loaded, the features of interest are extracted via the feature extractor function of the object studio of the

first bot, by passing the message getFeatures(). The features of interest are passed to the KBS via the checkErrorType() message which consequently returns the type of error(s) in the ledger. This result initiates the cleaning process depending on the error type passed in the message. This process can be repeated depending on the error types detected, till the ledger is fully cleaned. The user can finally request for the data cleaning results out of the process studio of the second bot.

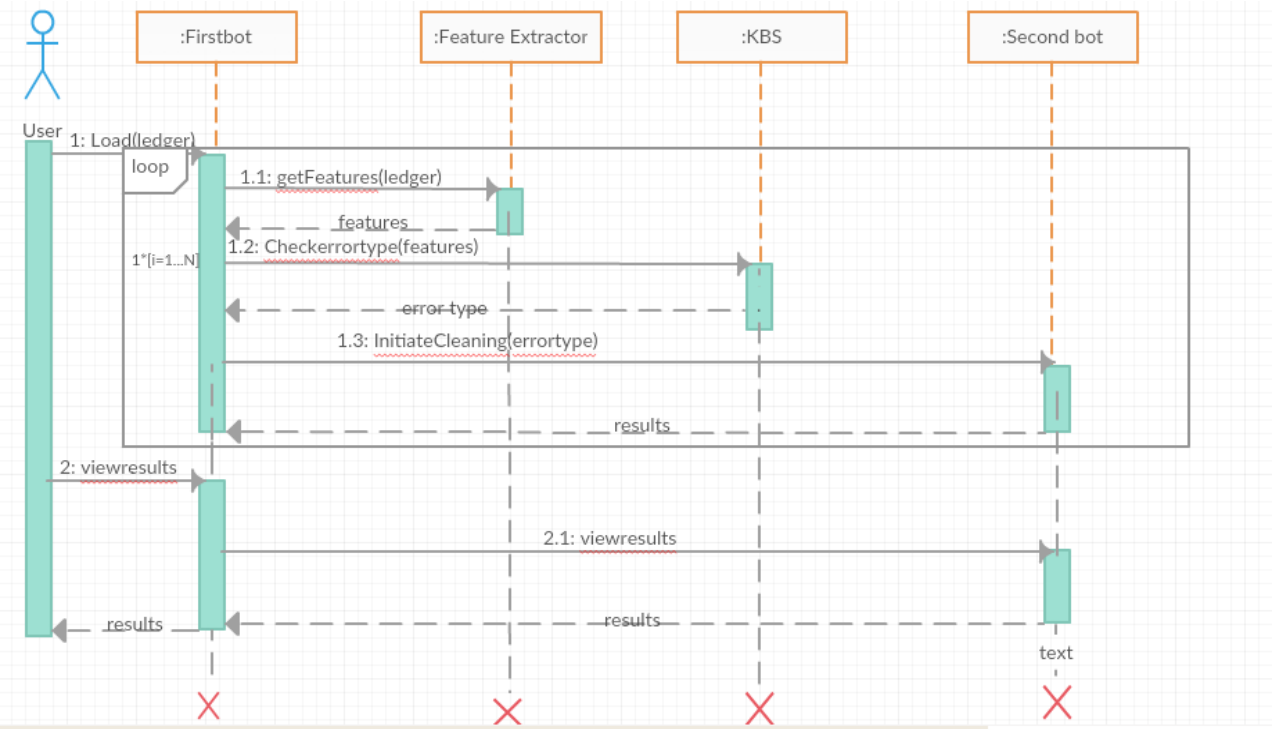


Figure 4.3: Sequence diagram

4.6 Context Diagram

Figure 4.4 below depicts the context diagram well illustrating the boundary of the prototype, its environment and the entities that it interacts with. In addition it also shows the various inputs and outputs from the prototype to the entities. The main entities interacting with the proposed prototype are the user, the RPA Data cleaning system and the KBS. The user is given a request to load the ledger to the RPA data cleaning system which in turn issues a data error classification request to the KBS. The KBS retrieves the error type, which initiates the data cleaning process. Eventually, the cleaned ledger is returned to the user.

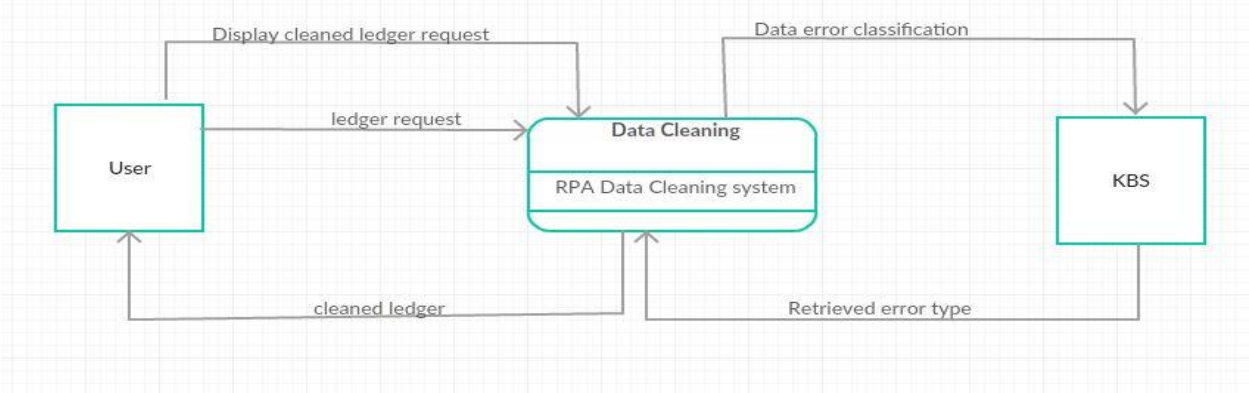


Figure 4.4: Context diagram

4.7 Level 0 Data Flow Diagram

The level 0 data flow diagram shown in the figure below gives details of the system illustrating the various processes contained in the modules, data stores and entities. The arrows shows the direction of flow of data among various components of the DFD. The first process called clean data receives a request from the user and passes the request to the KBS for data error classification. The KBS upon sufficient classification returns the retrieved error type. Process 1 stores the dirty data in data store D1 called dirty ledger data. Process 2 (Extract features) receives the dirty ledger data and extracts the necessary features for classification of the error type. The features thereof are stored in data store D2 called: Features of interest; which are then passed to process 3, known as classify data error for classification. The results thereof are stored in data store D3 known as Labelled data error. The ledger with labelled data error is passed to process 4 and the results thereof passed to data store D4 as clean data. Process 5 (Give feedback) receives a feedback request from the user in form of clean ledger data from data store D4

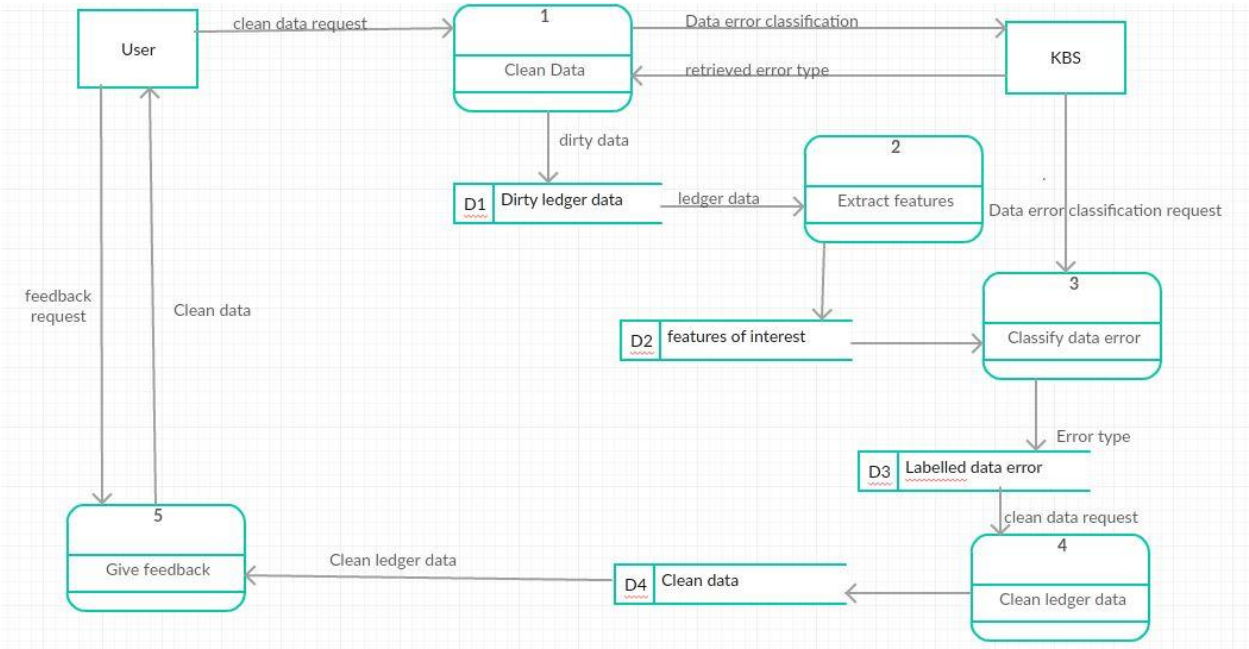


Figure 4.5: Level 0 Data flow diagram

CHAPTER 5:

SYSTEM IMPLEMENTATION

5.1 Introduction

This chapter describes how the Intellibot was developed. It starts by describing the pseudo code algorithm of the whole system. Since this is an integration of RPA and Knowledge-based system, it then describes the algorithm and the business processes used in developing the rules and facts implemented in the rule-based Knowledge-based system.

In addition to this, the development of the first bot (for error detection and clustering), and the second bot (for data cleaning), through Blue Prism's process studio and object studio; is also discussed plus the visual business objects integrated therein. The test results are then discussed in the next chapter.

5.2 The Pseudo-code Algorithm

The RPA-KBS (Intellibot) data cleaner, is a data-cleaning tool developed to detect and clean errors (specifically for this case – uncaptured losses, uncaptured returns and credit movement errors).

The following are the steps undertaken to clean errors in the ledgers, which can be implemented by the following pseudo code:

1. **Start**
2. Load ledger data in excel format into the process studio of the first bot, which is implemented by the bot accessing the location of the file, opening it and storing the records thereof into a collection
3. First bot extracts the values of the features of interest from the ledger file which are stored in data items
4. First bot feeds the values of the features of interest through an object studio to the Knowledge based system
5. The values - which are in form of answers to questions determined by the rules from the business process – are used to cluster the ledger data error(s).
6. **IF** ledger data error is 'Uncaptured Losses' or 'Uncaptured Returns' or 'Credit Re-adjustments' then go to step **(8)**

7. **ELSE** go to step **(9)**
8. Determine the specific error type
    a. Begin loop
        i. Second bot loads the respective data cleaning excel file
        ii. For credit re-adjustments, it splits the receipts and moves the credits to the respective years
        iii. For Uncaptured Losses, the bot confirms the losses from the respective returns and are captured, therefore readjusting the liability.
        iv. For Uncaptured Returns, the bot captures the return values and feeds them to the ledger
    b. End Loop
9. The second bot presents the cleaned ledger file in an excel format for validation
10. The ledger data is cleaned; ready for exportation to i-Tax.
11. **Stop**

5.3 Intellibot's Knowledge-Based System

The knowledge-based system developed has the facts in the knowledge base and is able to reason with incoming facts to be proved with the existing rules in the rule base via the inference engine.

The system is able to accept inputs from the user-where in this case, the bot and respond will feed it in by giving the error cluster in the ledger and the steps to be undertaken to do data cleaning on the ledger. The said input parameters will include the discrepancies between the debits and credits; consistency of credits or debits between the years; the status of the filed returns and the presence or absence of the filed return.

The knowledge-based system is rule-based and built using python and pyknow library for developing expert systems in python. The system is limited to detecting errors associated by the corporate income tax ledger return entry.

5.4 Production System Rules Used to Implement the System

FACTS:

Self-assessments were captured for years between 1992 – 2014.

Self-assessment status for captured returns =’1’

Self-assessment status for Uncaptured returns =’0’

The following **rules** will be used to implement the system

RULE 1:

IF (Self-assessment status for any year = ‘0’)

And (Self-assessments captured for years between 1992 – 2014 = ‘yes’)

THEN (‘Uncaptured returns’)

Explanation = ‘If there exists a status ‘0’ in any year under column of self-assessments status then it was uncaptured hence it should be captured by using copies of returns from the taxpayer, audited accounts and tax computations’

RULE 2:

IF (credit balances for successive years = ‘yes’)

And (Self-assessments captured for years between 1992 – 2014 = ‘yes’)

THEN (‘Credits not carried forward to successive years’)

Explanation = ‘Credit balances for successive years means losses were not captured for the first year hence not carried forward to subsequent years hence request all returns; audited accounts and corresponding tax computations for all the years affected and capture, then transfer to subsequent years. ’

RULE 3:

IF (debit balances for successive years = ‘yes’)

And (Self-assessments captured for years between 1992 – 2014 = ‘yes’)

THEN (‘Uncaptured Losses for preceding years’)

Explanation = ‘if successive years have debit balances and they are greater than credit balances for years between 1992-2014 in the ledger then losses for the preceding years were uncaptured, hence request for returns and capture the losses for the years affected’

Forward chaining was used as the inference mechanism, by having the features of the ledger data that can determine the error cluster thereof, it moves from the data given in order to reach the goal.

In order to be able to prove a given rule, the system will be able to answer four questions corresponding to the antecedents of the various rules.

The questions are as follows:

1. What is the self-assessment status of that particular year? (0 - missing, 1 - present)
2. Are any self-assessments existing? (yes/no)
3. Are there Credit balances for successive years? (yes/no)
4. Are there debit balances for successive years? (yes/no)

RESULT:

The result to these questions will be the proof of the rule.

5.4.1 Demonstration

To prove that:

IF (Self-assessment status = ‘0’)

And (Self-assessments captured for years between 1992 – 2014 = ‘yes’) **IS TRUE**

THEN ‘Uncaptured Returns’

5.4.2 KBS Code snippet

The system code will be as follows in the code snippet and the results of the questions as answered below, hence: the rule fires.

```
import tkinter as tk
from random import choice
from pyknow import *

window = tk.Tk()

window.title("RPA-KBS APP")

window.geometry("600x400")

#-----CLASSES AND FUNCTIONS-----
class Ledger_errors(Fact):
    pass

class LedgerErrorsInferenceEngine(KnowledgeEngine):

    def returns_missing(self):
        x = '\n The returns for the year(s) is uncaptured. '
        self.x=x

    @Rule(AND(Ledger_errors(self_assessment_status='0'),
              Ledger_errors(self_assessments_captured='yes')
              )
          )
    def credit_balances(self):
        x = '\n Uncaptured losses for preceding years. '
        self.x=x

    @Rule(AND(Ledger_errors(debit_balances_status='yes'),
              Ledger_errors(self_assessments_captured='yes')
              )
          )
    def erroneous_balances(self):
        x = ' \n credits not carried forward to successive years. '
        self.x=x

    @Rule(AND(Ledger_errors(credit_balances_status='yes'),
              Ledger_errors(self_assessments_captured='yes')
              )
          )

    def undefined_errors(self):
        print('\n')

def results_generator(x):
    self_assessment_status = str(entry1.get())
    self_assessments_captured = str(entry2.get())
    credit_balances_status = str(entry3.get())
    debit_balances_status = str(entry4.get())

    engine = LedgerErrorsInferenceEngine()
    engine.reset()
    engine.declare(Ledger_errors(self_assessment_status=choice([self_assessment_status]),

self_assessments_captured=choice([self_assessments_captured]),
                                 credit_balances_status=choice([credit_balances_status]),
                                 debit_balances_status=choice([debit_balances_status])))
    engine.run()

    y = engine.x
    return y

def results_display():
```

```
    display_to_screen = results_generator(1)

    #---To Create the Text Field--
    results_field_display = tk.Text(master=window, height=10, width=50)
    results_field_display.grid(column=0, row=7)

    results_field_display.insert(tk.END, display_to_screen)

#LABEL
label1=tk.Label(text="Welcome to the Intellibot Data Cleaner App")
label1.grid(column=0, row=0)

label2=tk.Label(text="1. What is the self assessment status? (0 or 1) ")
label2.grid(column=0, row=1)

label3=tk.Label(text="2. Are there self assessments existing? (yes/no)  ")
label3.grid(column=0, row=2)

label4=tk.Label(text="3. Are there credit balances for successive years? (yes/no) ")
label4.grid(column=0, row=3)

label5=tk.Label(text="4. Are there debit balances for successive years? (yes/no) ")
label5.grid(column=0, row=4)

#----ENTRIES-----
entry1=tk.Entry()
entry1.grid(column=1, row=1)

entry2=tk.Entry()
entry2.grid(column=1, row=2)

entry3=tk.Entry()
entry3.grid(column=1, row=3)

entry4=tk.Entry()
entry4.grid(column=1, row=4)

#Button
button1=tk.Button(text="VIEW RESULTS", bg="red", command=results_display)
button1.grid(column=0, row=6)




window.mainloop()
```

On running the above code snippet, it pops a user interface for the user to input the answers to the questions asked which will then be evaluated by the inference engine to give a result:

The user interface is as shown here below, with the questions answered as shown below. On evaluation by the inference engine, via forward chaining, the result is as shown below.

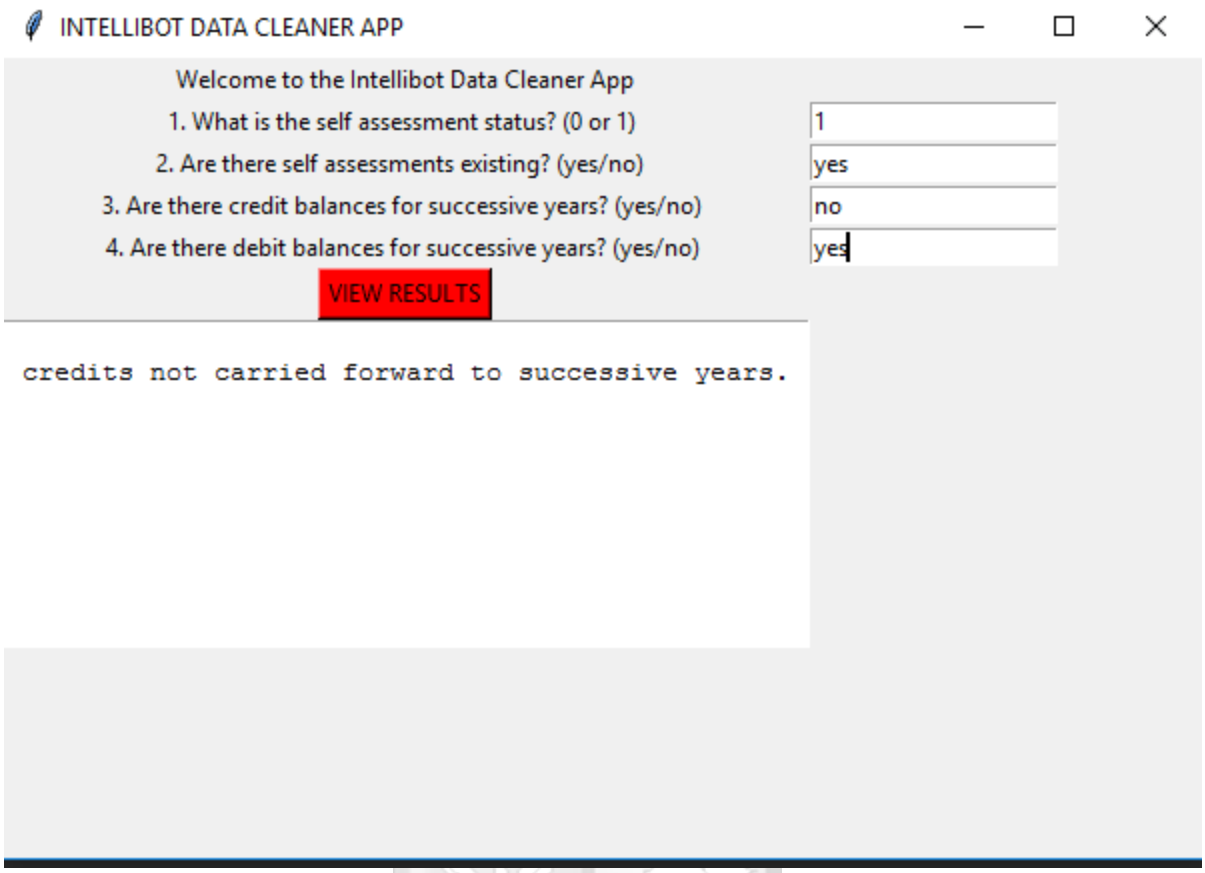


Figure 5.1: RPA-KBS Data Cleaner Application

5.5 Graphical User Interface for the system

The following graphical user interface (GUIs) demonstrate the various procedures the user through the Intellibot must implement to clean data from the ledger files. A brief description is given to show the operations of these interfaces.

5.5.1 The Main Interface

The Intellibot allows the user to begin running the bot by pressing the run button after which the bot loads the data from the ledger which is in excel format and processes it by extracting the features and feeding to the KBS.

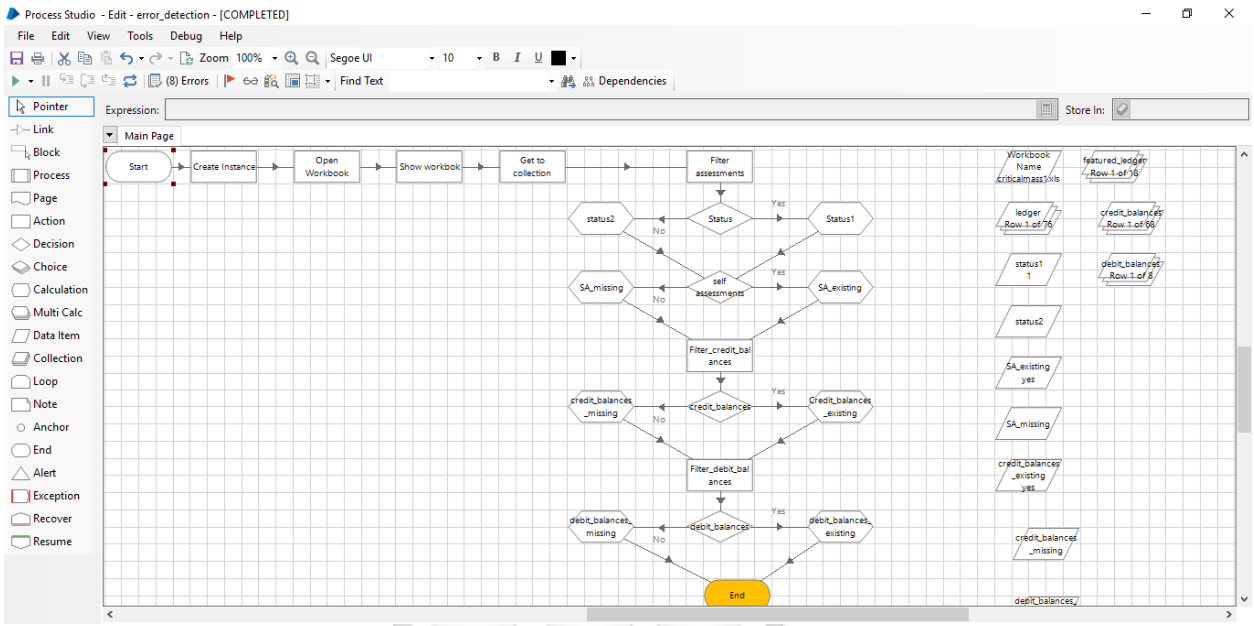


Figure 5.2: The Main Interface

5.5.2 Create Excel Instance

This interface creates an excel instance and allows the bot to load the ledger file into the process studio.

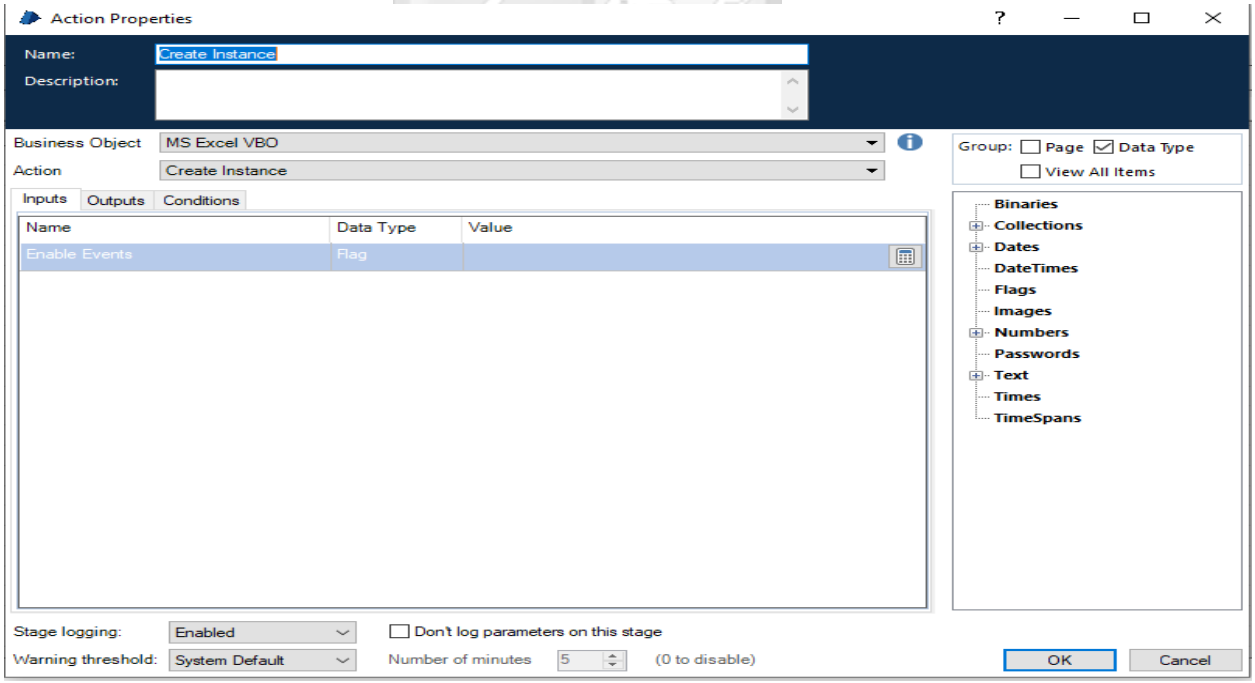


Figure 5.3: Create Excel Instance

5.5.3 Get to Collection interface

This interface allows the excel records to be loaded into a database-like collection ready for manipulation and feature extraction.

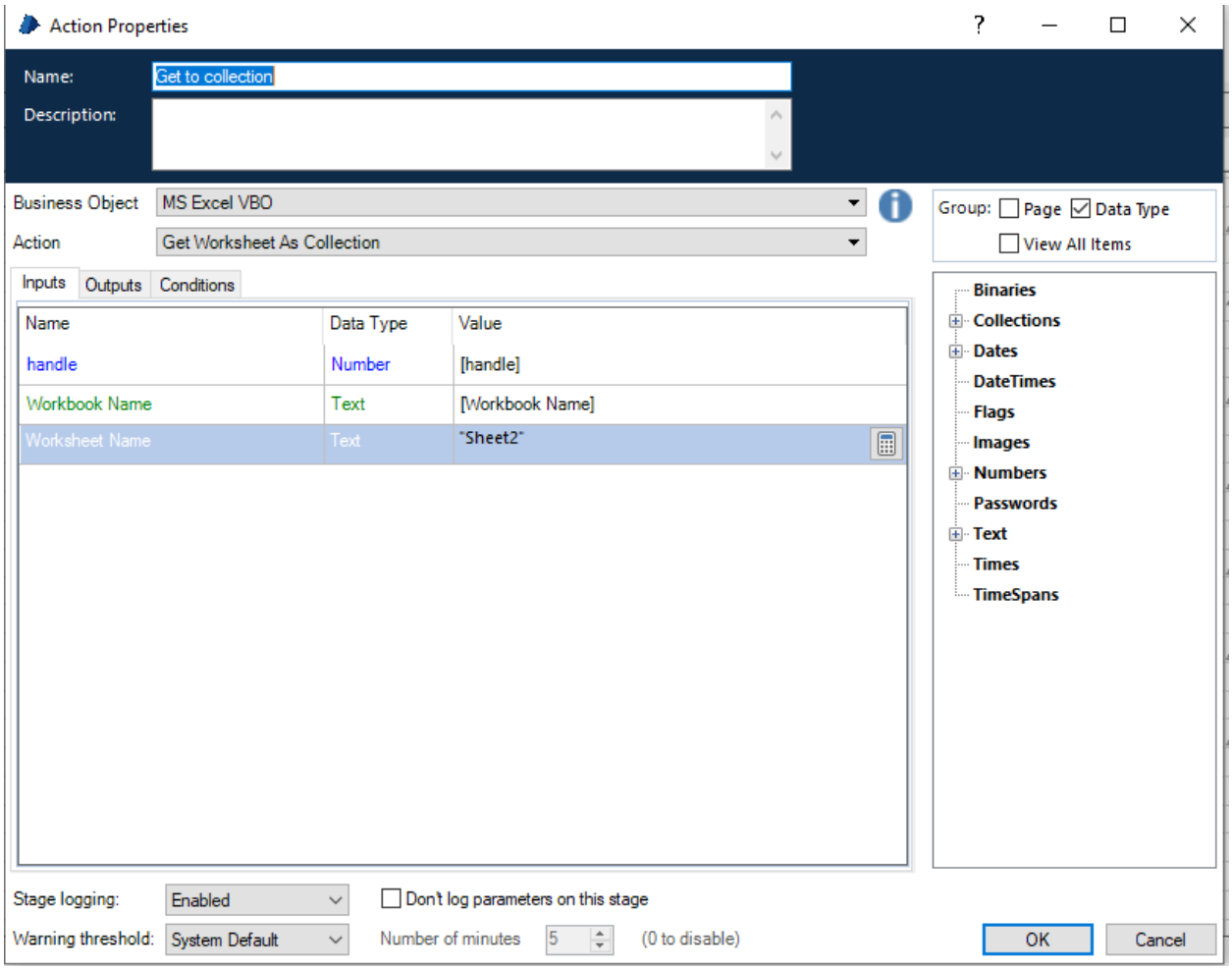


Figure 5.4: Get to collection interface

CHAPTER 6:

DISCUSSIONS

6.1 Introduction

This chapter discusses the results of the research according to the objectives set out in chapter one. The objective of this research was to develop an Intellibot system for data cleaning of KRA's legacy data. A KBS was developed to introduce intelligence into the system by detecting the errors in the ledger through the inference engine. The Intellibot was also developed using RPA's Blue prism platform for fast processing and for feature extraction. A number of experiments were conducted to validate the researcher's approach in data cleaning.

6.2 Experimental Test Results

6.2.1 Time taken by the Intellibot

The manual data cleaning process takes about a fortnight for the officers involved in the process to be able to identify the errors and start the data cleaning process, (Domestic Taxes Department, 2015), therefore the time taken for the Intellibot to complete the data cleaning process for the ten ledgers was measured by the blue prism's analytics control tool, and is as discussed hereunder. The error detection bot's processing time was compared to the data cleaning bot's processing time, in comparison to the size of the ledger files and the results in table 6.1 shown below:

Table 6.1: Time taken by the two bots

Ledger Data Size(KB)	Bot 1(Error Detection bot) Time(seconds)	Bot 2(Data Cleaning Bot) Time(seconds)
48	5.8	15.7
56	6.7	20.2

25	3.4	7.8
52	4.5	18.6
15	2.9	5.2
20	3.1	6.6
45	4.6	12.1
32	3.8	7.2
25	3.0	6.9
78	8.9	25.2

The Intellibot's performance of the two bots was tested and from the graph below its evident that the running time is quadratic for both bots with bot 2 having the highest running time. The smaller the input size of the ledger file, the smaller the time it takes the intellibot to process it- both for error detection and data cleaning. Bot 2 has a higher running time since the processes involved in data cleaning are CPU intensive and due to the user interaction, the processing time is longer.

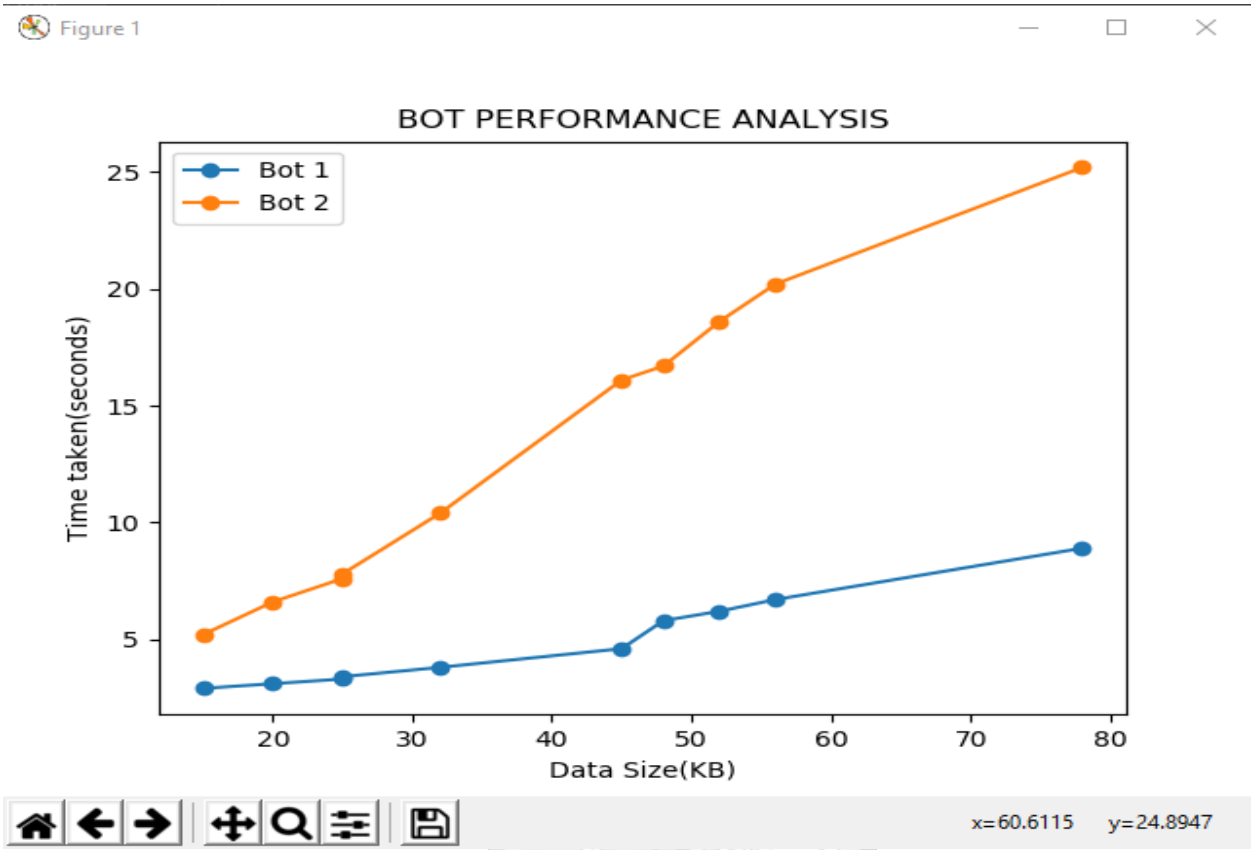


Figure 6.1: Graph of Bot performance analysis

6.2.2 Performance Depending on the Error Detected

From the discussion in the preceding chapters, the types of errors, which are being detected, are: credit movement errors, uncaptured returns and uncaptured losses. From the test results of the experiment, the following table illustrates the complete time taken to clean ledger data depending on the error type. From the ten ledger records that were used during the experiment, the following data was taken:

Table 6.2: Time taken to clean Uncaptured Returns Error Type

Data size(KB)	Time taken(seconds)
20	9.7

25	10.9
32	14.2

Table 6.3: Time taken to clean Uncaptured Losses Error Type

Data size(KB)	Time taken(seconds)
25	11.2
15	8.1
56	26.9

Table 6.4: Time taken to clean Credit movement Error Type

Data size(KB)	Time taken(seconds)
48	21.5
52	24.8
45	20.7
78	34.1

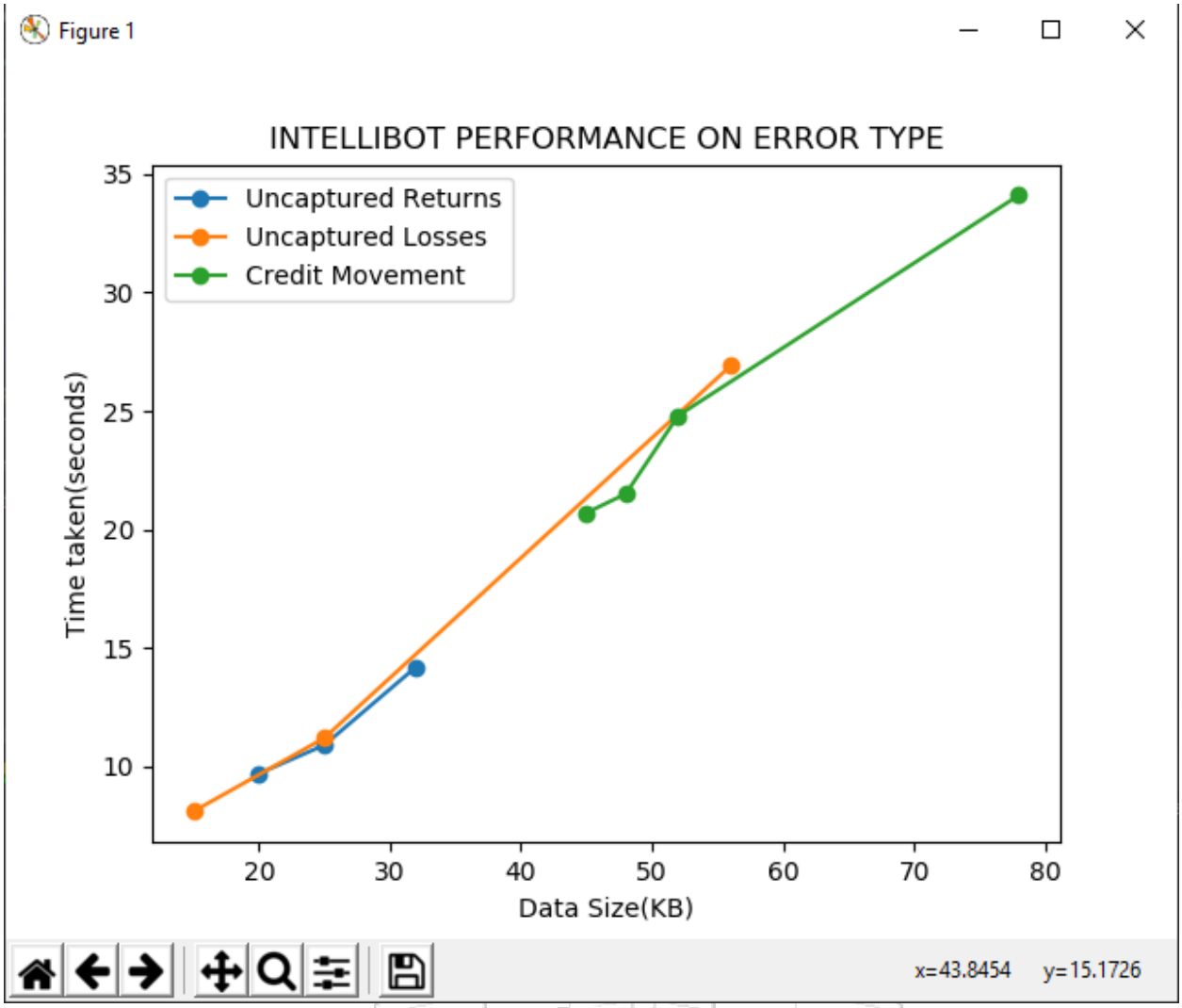


Figure 6.2: Graph showing intellibot performance depending on error type

From the graph above, it's evident that ledgers which have uncaptured returns take the least amount of time to clean, followed by ledgers which have uncaptured losses whose cleaning time varies depending on the number of years, which had losses that were not forwarded to the succeeding years. Finally, ledgers which have credit movement errors, take the greatest time to clean, due to the several excel sheets that have to be loaded into the Intellibot.

It is also evident that the more the user interaction on the system is reduced, the better and faster the performance as this reduces the time taken to pop up, several excel sheets or interfaces. Also on full deployment to dedicated servers, the Intellibot's data cleaning rate will be faster since it uses several distributed CPU's to access the various functionalities, hence the data cleaning tool can achieve a variable

amount of work done in a faster rate- a characteristic known as the scale up factor (amount of speed up or gained throughput an application achieves when the number of CPU's on the system is increased).

The bot performances are relatively efficient as compared to the performance of C.P.U's in data cleaning(Kofi, 2013) as shown below. From this graph its clear that the average cleansing time was faster than the loading time. This is the opposite of the intellibot's performance wherein the cleaning time is higher than the time taken for error identification.

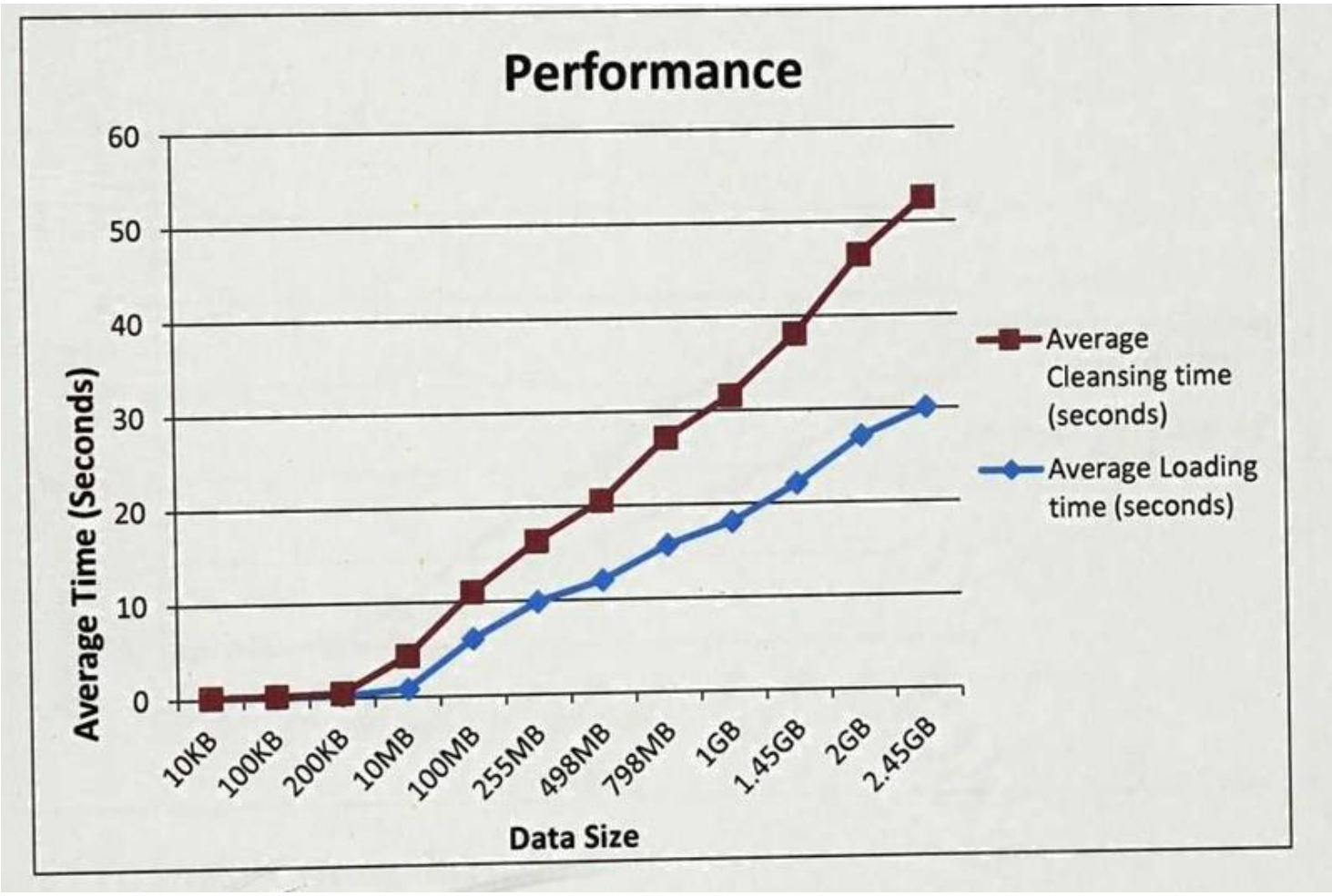


Figure 6.3: graph of missing data fields. Adapted from "A conceptual Framework for Data Cleansing in a data warehouse".

6.3 Test Phases

The Intellibot's algorithm and functionality was tested based on the various phases of testing. Listed below are the test phases included in the overall System Test timeframe in the table below. The testing phases include the integration, system testing, user testing, security and end-to-end testing.

Table 6.4: Test Phases

Test Type	Focus
Integration Testing	The system was tested to find errors in the programs through debugging and the processes were checked to confirm that they were implemented according to the business processes.
System Testing	During the system testing, it was validated that the system performed as specified and produced the right results. Before effective deployment of the system, its been confirmed that the functional requirements have been defined to show how the system should perform.
End-to-End Testing	An operation is validated after a process has been initiated. At the entry and exit points, the system was tested.
Security Testing	Eliminates security accessibility errors

User Testing	Several Users have been incorporated to test the system in order to validate the functional requirements

Further to the system tests, a comparative analysis between the four systems reviewed under the literature review section were compared to the Intellibot system developed. The table below shows the comparison between the systems.

Table 6.5: Comparison between reviewed systems and the Intellibot

Parameter	Potters Wheel	AJAX	IntelliClean	ARKTOS	IntelliBot
Interactivity	Very Interactive, hence easy to Use	Complex Interface and hence not friendly to non-technical persons	Interactive with end user, However, requires little input from end users.	Highly Interactive, it has graphical interfaces for loading and executing validations on loaded files.	Little interactivity with the user, since the bot interacts with GUI's, does the loading, cleaning, validation and exporting data files.
Data Format/Structure	Text	Text	Text	Text	Numbers

Human Dependency	High human dependency for exceptional errors	High human dependency for evaluation and validation of errors is fully dependent on human expert	Minimal because of the expert module embedded in the system	The system has complex modules for dealing with duplicates, however, there is a high dependency on human experts for error correction	Little human dependency due to the expert module for error detection and the two bots involved in error detection and data cleaning
Maintenance	Not considered	Not considered	Not considered	Not considered	Relatively expensive due to the bots involved
Detailed Cleaning Process	Not considered	Not considered	Not considered	Not considered	Not considered

CHAPTER 7:

CONCLUSIONS AND RECOMMENDATIONS

7.1 Conclusion

Poor quality data has been a menace to organizations for decades costing them yearly as this leads to poor decision making, as well as mistrust in customer relationship. Data still stands as the core business asset that needs to be managed if an organization is to generate a high return on Investments from its business processes (Porwal and Vora, 2013).

A lot of research work has been conducted in the field of data cleaning in the past to combat the arising inconsistencies and errors that are dependent on the organization's activities; most of which have presented different methods and approaches of which some have been discussed herein.

The impetus of this research paper, however, was to provide a new approach to data cleaning by considering elimination of human effort since they are deemed to introduce more errors into the system, and instead introducing an expert system to detect the errors. The Intellibot system is meant to help in cleaning legacy data in KRA, and due to the man hours involved in data cleaning, the speed and time it takes to undertake this process was of essence since there are thousand s of ledgers that needs cleaning, hence the need to incorporate robotic process automation. The bots are meant to do the repetitive rules based tasks that are involved in the data cleaning process, thereby reducing the cleaning process to seconds instead of the days or weeks that could be taken to clean one ledger record.

In this research it was necessary to understand the background of KRA's data cleaning processes by reviewing relevant literature as well as participant observation of the processes involved. The literature also helped gain an insight on the various errors in data and data cleaning approaches that are used. The main objective of this research was to develop the Intellibot system through incorporating the KBS expert module, for error detection. The various rules were extracted from the data cleaning business processes. The algorithm was restricted to identification of only three types of errors in the ledgers. Finally, several experiments were carried out to determine the data cleaning time taken by the different bots to accomplish the tasks as well as the time taken to clean the different errors, which is a quadratic function and hence very efficient as compared to the manual data cleaning procedure.

7.2 Contributions to the Research

The research presented a new dimension to data cleaning approaches, addressing the problem of erroneous data in most organizations, by incorporating the robotic process as well as the KBS expert module in the same system. The study considered the replacement of huge and complex calculations used by other algorithms, and adapting the use of rule-based KBS system implementing the business processes for data cleaning. The difference in computational time between this work and the existing works greatly depends on the amount of input data size loaded into the system for the cleaning operation to be performed and the system requirements. Further, a comparative analysis was done to show how the proposed system was measured with the reviewed systems using the same parameters.

7.3 Recommendations and Future Work

This research proposed the development of an intelligent robot for data cleaning, which works faster than all the other data cleaning methods and approaches. This Intellibot can be deployed even in other organizations having different data sets from KRA's.

However, it was limited to cleaning only ten ledger records for the sake of this research, however future research can enlarge the scope to more ledger records, for better results. Also, the errors detected were only limited to three, notwithstanding the many errors inherent in KRA's ledger records. However, the system is scalable to accommodate detection of more errors and therefore, future research work should concentrate on detection of more errors. Further to this, this research was only limited to cleaning of errors in the corporate income tax ledger returns only, however, there are other ledger records including the VAT ledger, withholding tax ledgers, individual income tax edger records, which should be considered for cleaning in future work.

In conclusion, RPA is an emerging technology that can be incorporated in all the disciplines and it is recommended that the Intellibot data cleaning tool should be extended to be used in other organizations and it will give out a huge return on investments.

REFERENCES

Ajzen, I. (1985). From intentions to actions: a theory of planned behaviour, in Kuhl, J. and Beckmann, J. (Eds), Action-Control: From Cognition to Behaviour, Springer- Verlag, Heidelberg, pp. 11-39.

Angehrn,A.A. (1992) . Stimulus Agents: An Alternative Framework for Computer-aided Decision-making, *Proc. DSS 92*, M.S. Silver, ed., Inst. of Management Science, pp. 81-92,

Angehrn.A, & Dutta.S. (1998). Case-Based Decision Support, *Comm. ACM*, to appear

Arindam P, Varuni G. (2012), *HADCLEAN:A Hybrid Approach to Data Cleaning in Data Warehouses*, IEEE Press, pp 136-142.

Bandura, A., & Cervone, D. (1986). Differential engagement of self-reactive influences in cognitive motivation. *Organizational Behavior and Human Decision Processes*, 38(1), 92-113.

Bhatnagar, M., & Singh , K. (2013). Research Methodology as SDLC Process in Image Processing. *International Journal of Computer Applications*, Vol 77 No 2.

Burnett, S. (2015), A Conversation with Wayne Butterfield, Head of Digital Service Innovation & Transformation at Telefónica, Everest Group Practitioner Perspectives, EGR-2015-4-0-1422.

Business Daily Africa. (2012, 06 24). *business daily africa*. Retrieved from business daily africa website: https://www.businessdailyafrica.com/news/KRA-expands-number-of-tax-collectors-to-meet-targets-/539546-1434764-8fojbp/index.html

Bryman, A. (2012). *Social Research Methods.* New York: Oxford University Press.

Carr, V. H. (1999). Technology adoption and diffusion. The Learning Center for Interactive Technology.

Chen,W., Kife,M., & Warren.D.S, (1993) HiLog: A foundation for higher-order logic programming. In *Journal of Logic Programming*, vol 15, pp 187–230,

Cooper, D., &Schindler. (2006). Business Research Methods, (9th Ed.). New Delhi, Tata

Cooper, H. (1998). *Synthesizing Research: A Guide for Literature Review 3rd ED.* Carlifonia, USA: Sage Publications, Inc.McGraw-Hill Publishing Company Limited

Data extraction, transformation, and loading tools (ETL). *Retrieved from www.dwinfocenter.org/clean.html*

Davis, F. D., Bagozzi, R. P., & Warshaw, P. R. (1992). Extrinsic and intrinsic motivation to use computers in the workplace1. *Journal of Applied Social Psychology*, 22(14), 1111-1132.

Deepali, V., &Sonal, P. (2013). A Comparative Analysis of Data Cleaning Approaches to Dirty Data. *International Journal Of Computer Applications* (0975 -8887), Vol 62 No 17

Deloitte. (2018). *Tax Value of Robotic process automation.* Retrieved from https://www2.deloitte.com/us/en/pages/tax/solutions/tax-robotic-process-automation.html

Domestic Taxes Department. (2010). *Operational Manual: Compliance and Debt Management (2nd ED).* KRA.

Domestic Taxes Department. (2018). *Data Cleaning Project Charter*.

Duncan.N.M (1989), *Case-Based Reasoning Applied to Decision Support Systems*, masters thesis, Queen's Univ., Kingston, Canada,

Ernst and Young. (2018, 04 02). Retrieved from ey.com Web Site: http://www.ey.com/Publication/vwLUAssets/EY-what-role-will-robots-play-in-your-tax-function-rpa/$FILE/EY-what-role-will-robots-play-in-your-tax-function-rpa.pdf

Feldman, R., & Sanger, J. (2007). *The Text Mining Handbook: Advanced Approaches in Analyzing Unstructured Data.* New York: Cambridge University Press.

Fishbein M., Ajzen, I.(1975). *Belief, Attitude, Intention, and Behavior: An Introduction to Theory and Research*, Addison-Wesley, Reading, MA.

Forrester Research (2014). *Building a Center of Expertise to Support Robotic Automation*.

Galhards,H,, Florescu.D, Shasha.D, Simon.E. (May 2000). AJAX: An extensible data cleaning

tool. Proceedings of the ACM SIGMOD on Management of data, Dallas, TX USA, pp. 21-22.

Hammond,K. (1986), *CHEF: A Model of Case-Based Planning," Proc. AAAI 86, AAAI Press/MIT Press, Cambridge, Mass.*,

Hellerstein.J.M. & Rahman.V. (2001). *Potter's Wheel: An Interactive Data Cleaning System* control.cs.berkeley.edu/abc, also UCB CSD-00-1110,

ICPAK. (2015, September). *ICPAK*. Retrieved from ICPAK.COM: https://www.icpak.com/wp-content/uploads/2015/09/iTax-Presentation.pdf

Income Tax Act (CAP 470), (2010).

International Tax compact. (2015). *taxcompact*. Retrieved from taxcompact.net: https://www.taxcompact.net/documents/ITC_iTax-case-study.pdf

Kenya Revenue Authority Strategic plan (2012).*Sixth Corporate Plan*, 2012/2013

Kolodner,J.L. and Simpson.R.L (1989), The MEDIATOR: Analysis of an Early Case-Based Problem Solver, *Cognitive Science*, vol. 13, no. 4, pp. 507-549,

Kumar, R. (2017, December 18). *b2c.businessinnovation*. Retrieved from B2C: https://www.business2community.com/business-innovation/machine-learning-powered-robotic-process-automation-rpa-next-step-toward-smarter-automation-01976004

Lacity, M., Willcocks, L., & Craig, A. (2015, April). *Robotic Process Automation at Telefonica O2*. The Outsourcing Unit Working Research Paper Series.

Lee.M, et al. (1999). Cleansing data for mining and warehousing. In *DEXA*.
Manheim M.L. (1988). An Architecture for Active DSS, *Proc. 21st Ann. Int'l Conf. System Sciences*, IEEE CS Press, vol. 3, Hawaii, pp. 356- 365,

Maydanchik.M. (1999). Challenges of efficient data cleansing. DM Review,Retrieved From http://www.dmreview.com/editorial/dmreview/ print action.cfm?EdID=1403.

Mortadha, M.' &AbdulkarJihad, A. (2011). An Enhanced Technique to Clean Data in The Data Warehouse. IEEE

Mugisha, S. (2010). Using ICT in development: the case of Uganda international conference

paper on Information Technology. *Communications and Development ITCD* (36), 29th-30th November, Kathmandu, Nepal.

Muller,H., Freytag.J.C., (2003). Problems, Methods, and Challenges in Comprehensive Data Cleansing, pp. 21.

My Tax. (2013). Retrieved from mytax.com: http://my-itax.com/general-features

Naumann.F.(2002), quality Driven Query Answering for Integrated Information Systems, Lecture Notes in Computer Science, LNCS 2261, Springer. Pp.34

Naz, R., & Khan, M. (2015). Rapid Applications Development Techniques: A Critical Review. *International Journal of Software Engineering and Its Applications*, 163-176.

Norton, P. (2006). Introduction to Computers. McGraw-Hill Technology Education. China

Orawit, T. (2006). *Rapid Application Development.* Chiang Mai University.

Pandas. (2017, March 21). *Pandas*. Retrieved from Pandas: http://pandas.pydata.org/

Panos V., Zografoula V, Spiros S., and Nikos K. (2000). ARKTOS: A Tool For Data Cleaning and Transformation in Data Warehouse Environments. Bulletin of the IEEE Computer Society Technical Committee on Data Engineering, pp.1-6

Price Waterhouse Coopers. (2017, May). *Tax Functions of The future series.* Retrieved from price waterhouse coopers: https://www.pwc.com/gx/en/tax/publications/assets/pwc-tax-function-of-the-future-focus-on-today-robotics-process-automation.pdf

Rahm. E., & Do H.H. (2000). *Data Cleaning: Problems and Current Approaches.* University of Leipzig. Germany.

Rajiv A,Payal P., & Shubha B. (2009*). Alliance Rules for Data Warehouse Cleansing, 2009.IEEE Press, pp 743-747.*

Rogers, E. M. (2003) *Diffusion of Innovations*, Free Press, New York, Fifth Edition .

Rundensteiner, E. (1999). *Special Issue on Data Transformation. IEEE Techn. Bull. Data Engineering 22(1),*

Soumitra. D. (1993), *Knowledge Processing and Applied Artificial Intelligence*. Oxford, England: Butterworth-Heinemann,

Tack, G. (2017). *What role will robotics play in your tax function.* EYGM LIMITED.

Tax Procedures Act (CAP 29), (2015).

Thompson, R. L., & Higgins, C. A. (1991). Personal Computing: Toward a Conceptual Model of Utilization. *MIS quarterly*, 15(1), 125.

Thyer, B. (1993). Single-systems Research Design. In R. Grinnell, *Social Work Research and Evaluation* (pp. 94-117). Illnois: F.E. Peacock.

Vallerand, R. J. (1997). Toward a hierarchical model of intrinsic and extrinsic motivation in Advances in Social Psychology (29), M. Zanna (ed.), Academic Press, New York,, pp 271-360.

Vassiliadis.P. et.al, (2001).ARKTOS: *towards the modelling, design, control and execution of ETL Processes. Information systems*, vol.26, pp 537-556.

Yan H, (2008). *Research on Information Quality Driven Data Cleaning Framework.IEEE* ,pp 537-539

Jerry Thesis

ORIGINALITY REPORT

29%	27%	9%	11%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	research.ijcaonline.org Internet Source	9%
2	paul.rutgers.edu Internet Source	3%
3	intellipaat.com Internet Source	2%
4	ijarcsse.com Internet Source	1%

5	www.cs.uiuc.edu Internet Source	1%
6	www.pwc.nl Internet Source	1%
7	Submitted to Strathmore University Student Paper	1%
8	Submitted to Asia Pacific University College of Technology and Innovation (UCTI) Student Paper	1%
9	www.hollandlaw.nl Internet Source	1%
10	jisr.szabist.edu.pk Internet Source	<1%

11	www.dbis.informatik.hu-berlin.de Internet Source	<1%
12	Hao Yan. "Research on Information Quality Driven Data Cleaning Framework", 2008 International Seminar on Future Information Technology and Management Engineering, 11/2008 Publication	<1%
13	widit.slis.indiana.edu Internet Source	<1%
14	Lup Low, W.. "A knowledge-based approach for duplicate elimination in data cleaning", Information Systems, 200112 Publication	<1%