Addis Ababa University

College of Natural Science

Design of Automatic Open Government Data Linker to the Linked Open Data Cloud: The Case of Ethiopia

Yemaneberhan Lemma Kebede

A Thesis Submitted to The Department of Computer Science in Partial Fulfillment for the Degree of Master of Science in Computer Science

Addis Ababa, Ethiopia

February 2020

Addis Ababa University

College of Natural Science

Design of Automatic Open Government Data Linker to the Linked Open Data Cloud: The Case of Ethiopia

Yemaneberhan Lemma Kebede

Advisor: Solomon Atnafu (PhD)

Co-Advisor: Melkamu Beyene (PhD)

This is to certify that the thesis prepared by Yemaneberhan Lemma Kebede, titled: *Design of Automatic Open Government Data Linker to the Linked Open Data Cloud: The Case of Ethiopia* and submitted in partial fulfillment of the requirements for the Degree of Master of Science in Computer Science complies with regulations of the University and meets the accepted standards with respect to originality and quality.

Signed by the Examination Committee:

Name	Signature	Date
Advisor: Solomon Atnafu (PhD)	_____	
Examiner: Dagmawi Lemma (PhD)	_____	
Examiner: Ayalew Belay (PhD)	_____	

Abstract

The transformation of web of documents into web of data has brought an opportunity to integrate open government data of nations of the world. Linked data- the web of data, is now a hot topic in large number of organizations and countries and they seized this opportunity and are exploiting its functionalities and features. Developing countries like Ethiopia showed some interest in this area but has not yet developed adequate dataset due to capacity limitation in gathering and publishing data. This thesis designs an automatic linker that enables gathering of source data starting from the lower to the higher administrative division of a government and process the data to be ready for Open Data Cloud.

This work enables the upload of different types of data as input and performs removing duplications and domain mapping to ease the task of converting data to Turtle serializations where data is represented in URI format and stores it as RDF dump file. Then the RDF analyzer performs fixing the data from misplacement/duplication and link the RDF data to corresponding data on the LODC by the method of content negotiation. And finally publishing the data.

The implementation of this work has verified the fact that all functional processes in this work are realistic. Evaluation is done for the work and a measured based on performance and it has proven that the larger the data, the more time it will take to process. Data results can be obtained by encoding search key or SPARQL query statement in OG Linked Data Search Interface.

Key words: Linked Data, *Open Government Data, RDF Conversion, RDF Analyzer, SPARQL,*

DEDICATION

To my mother Tsige Lemma

Acknowledgements

Gratitude to the almighty GOD for helping me to never give up on this work after long time of distractions, obstacles and information overload. I can’t thank enough my advisor, Dr. Solomon Atnafu, the one who is the original composer of this work. He helped me settle for this work when I was swinging from one research topic to another and my co-advisor, Dr. Melkamu Beyene, the one who supported me developing knowledge and ambition towards Linked Data.

My family as a whole, my dear wife for all the support. And my little daughter for being a reason to wish for more. And my sister for being on my side all along.

Table of Contents

List of Figures

List of Tables

List of Algorithms

Acronyms and Abbreviations

CSV	Comma Separated Value
DB	Database
HTML	Hyper Text Markup Language
JSON	JavaScript Object Notation
JSON-LD	JavaScript Object Notation – Linked Data
LOD	Linked Open Data
LODC	Linked Open Data Cloud
MCIT	Ministry of Communication and Information Technology
N3	Notable 3
OG	Open Government
OGD	Open Government Data
OWL	Web Ontology Language
PDF	Portable Document Format
RDBMS	Relational Database Management System
RDF	Resource Description Framework
SPARQL	SPARQL Protocol and RDF Query Language
W3C	World Wide Web Consortium
XLS (XLSX)	Spreadsheet file create Microsoft Excel
XML	Extensible Markup Language

Chapter One: Introduction

1.1 Background

As soon as the Open Data Movement is started, there have been a new government initiatives evolving all over the world. Open Government Data (OGD) is used as one of the basic communication means between governments and citizens [1].

In Europe and North America, access to government data, and the possibility to freely use it, is seen as an enabler for Open Government and a goldmine of unrealized economic potential [2]. Open data is data that can be freely used, re-used and redistributed - subject only, at most, to the requirement to attribute and share-alike [3].

OGD has socio-political, economical and operational/technical benefits if used effectively. Effective application of OGD can bring political and social advantages, for instance, more transparency, democratic accountability, more participation and self-empowerment of citizens (users), creation of trust in government, public engagement, close inspection of data, equal access to data, new governmental services for citizens, Improvement of citizen services, improvement of citizen satisfaction, improvement of policy-making processes, more visibility for the data provider, stimulation of knowledge developments, creation of new insights in the public sector, new (innovative) social services and more. Active implementation of OGD can bring serious economic benefits, for instance, economic growth, increase competitiveness, encourage innovation, contribute in the advancement of processes in producing better goods and delivering quality services. Integration of a variety of knowledge or in other words having the access to organized/structured data, as Ethiopian proverb “When spider webs unite, they can tie up a lion.”, well-structured and well-designed linking of Ethiopian Government data can make a difference in all sectors of the country. Since “every” government data is recorded it will create transparency and accountability. OGD has operational and technical benefits, for instance, the ability to reuse data / not having to collect the same data again and eliminating unnecessary duplication and associated costs (also by other public institutions), optimization of administrative processes, improvement of public policies, access to external problem-solving capacity, fair decision-making by enabling comparison, easier access to data and discovery of data, creation of new data based on combining data, external quality checks of data (validation), sustainability of data (no data loss), the ability to merge, integrate and mesh public and private data and more [4].

According to Open Data Barometer Annual Report of 2016, from 115 countries the research is made, key government data is not available as open data, and when published are in non-standard formats. For example, even in the case of public transport, where data standards are well established, just 43% of countries surveyed by Open Data Barometer, have machine-readable data available. Mapping data is also often unavailable in digital forms, or only available for a fee, suggesting that inefficient charging for public data continues to be an issue in many countries [5].

From the distinct download options identified in the technical survey made by Open Data Barometer in 2014 [6], 35% of the files were provided in XLS format, 20% in CSV format, and 8% in XML. Just 2% in JSON files were identified. In general, with the exception of transportation data where the GTFS (General Transit Feed Specification) standard was used in 11 of the cases examined, there was very little evidence of the use of global standards to represent key datasets. The main case is limitation in availability of reference standards to use. The absence of clear standards for representing key datasets has two issues. The first issue is it lacks standard measurement in assessing the adequacy or good quality publication of certain kinds of data. The second issue is users in need of data link up data from different countries, or to transfer an application developed in one context for use in another, have to re-learn and re-code their data uses country-by-country. This indicates that there is a need to have a linking mechanism that can handle common standards developed for open data.

Linked Data refers to a set of best practices for publishing and interlinking structured data on the Web. These best practices were introduced by Tim Berners-Lee [7] in his Web architecture note Linked Data and have since then become known as the Linked Data principles. These principles are use of URIs as names for things, use HTTP URIs so that people can look up those names. When looking up a URI, useful information can be provided, using the standards (RDF, SPARQL) and include links to other URIs, so that they can discover more things. Linked Data enables the flexible virtual integration of government data, through linking, without requiring to redesign information systems and to centralize data in data silos. This will facilitate the collaboration between different public sector agencies in the provision of common services [2].

In the case of Ethiopia, there have been some willingness to share government data to the public. This willingness has been demonstrated by the initiative of government data on the web portal “ethiopia.gov.et”. There was also “data.gov.et” (currently not active). It was owned and prepared by the then Ministry of Communication and Information Technology (now it is Ministry of

Innovation and Technology), this web portal was intended to be the one that enables Ethiopian government data to be accessed openly fulfilling the international standards. The website of Central Statistics Agency (previously "csa.gov.et" and now "statsethiopia.gov.et) is providing multiple statistical government data of Ethiopia. Most of the government offices (ministries, commissions, institutes and agencies) have their own website[1] to disseminate information. federal and on the web portal of regional and federal offices [25].

Based on the 2013 and 2014 global report of Open Data Barometer, Ethiopia ranked 66th (in 2013) and 78th (in 2014) of OGD and 81st (2016) and clustered in the capacity constrained countries [5, 6, 24]. The countries in this cluster all face challenges in establishing sustainable open data initiatives as a result of: limited government, civil society or private sector capacity; limits on affordable widespread Internet access; and weaknesses in digital data collection and management.

From the study of Open Data Barometer, the impact of Ethiopian OGD is 0% [5, 6]. This shows the data provided by the government bodies is not presented in a way that can be easily utilized or the data provided is inefficient. Most of the Ethiopian OGD available on government web portals are presented either in PDF, Excel, Word or Plain HTML format. Since OGD has positive the impact on the socio-economic development of a country, Ethiopia is not achieving growth with regard to OGD because the level of OGD implementation is too small in the country according to Open Data Barometer survey. Countries like UK, Sweden, New Zealand and US have achieved well in implementing OGD and their OGD has impact scaled above 85%. The survey also showed the difference among good achievers and the ones that are low ranked is related to economical level and the willingness of the government bodies. To overcome this challenge of implementing OGD, Low ranked countries like Ethiopia needs two directions. The first direction is the commitment from the government of Ethiopia to set strategies and act upon them in a very effective administrative way. The second direction is developing a mechanism that simplifies the process of gathering Ethiopian Open Government Data and availing the data gathered data in a well-structured and relatively complete data.

This thesis work focuses on the second direction mentioned. This work is proposing to design a means that enables to perform automatical linking of OGD (in our case Ethiopia OGD) to other

[1] https://www.eservices.gov.et, https://www.pmo.gov.et, https://www.mfa.gov.et, https://www.moe.gov.et, https://www.investethiopia.gov.et, http://www.moh.gov.et, http://www.moh.gov.et, http://www.mint.gov, etc.

datasets found locally and outside the country in the Linked Open Data Cloud. The implementation of Linked data will have a big positive impact on Ethiopian OGD. If one data of a small town is placed in its region's linked dataset it can be accessible in the country and global level and it may link to other datasets related to the small town dataset. This will never happen if it is not for linked data.

According to the FDRE government website ethiopia.gov.et[2], Ethiopia's government data is unique in different cases like Calendar, government structure. And this brings the question of how this unique data can be represented in Linked data standard and linked to other datasets. The Ethiopian OGD needs to be modeled which can be suitable for the Linked Open Data Cloud. This work will be designing a model that can represent the Ethiopian OGD in the Linked Open Data Cloud and designing an algorithm that can link the Ethiopian OGD to related datasets which can enhance the content of the data.

The success of this work will bring a significant impact in making OGD accessible and reusable by extending source data through modeling the data to be ready-to-publish on LODC fulfilling all criteria. Researches related to OGD will be easy because a standard and reusable data will be available. Adequate data can be available for investors, tourists, and foreigner in general to have the right image of the country. Local innovations can rise in number since areas open for opportunity can be easily figured out. It is simple to notice that, growing in foreign investment and local innovations can increase employment rate and can have a significant impact in the county's economy. The value, the completeness or the reliability of the OGD will increase because the OGD will be linked to other datasets semantically and other uncountable socio-economic advantages will be recorded. This will also make providers of the data to be cautious and accountable.

1.2 Motivation

The major initiatives to come up with this thesis work are derived from understanding that Linked Data and Open Government Data are hot issues throughout the globe and from noticing that Ethiopia is not benefiting from these approaches. The Web is evolving from a "Web of linked documents" into a "Web of linked data". In Ethiopia data is still accessible manually, information systems and databases and is represented using different, usually not aligned, vocabularies and

[2] http://www.ethiopia.gov.et/about-ethiopia

schemas. This disintegration contributes somehow to corruption, and bad governance. Linked OGD can promote good governance and transparency.

In North America and Europe, the possibility to freely access and utilize government data is becoming a means for Open Government and a positive impact in economic growth [2]. However, currently it is challenging to find well organized data in Ethiopia that is structured and interlinked to related data [5, 6, 24]. Linked Data can respond to this challenge and can facilitate eGovernment transformation, leading to smarter and more efficient government services and applications, and fostering creativity and innovation in the digital economy [2].

1.3 Statement of the Problem

This work is proposing to design an automatic linker of OGD to other datasets found locally and other datasets in the Linked Open Data Cloud. The need to implement Linked Data standards and mechanisms comes from the fact that Linked data appear to be the feasible solution existing now in order to extend the value of data[15]. It will answer the question of data format incompatibility and adding more things/data to the interlinked ones. For example, the "eservices.gov.et" web portal comprises limited amount of data about Ethiopia and hyperlinks to other regional level and federal level web portals. For instance, if a data about Mekdela is searched, it will not be found in the national web portal but it may be found from the Amhara regional web portal. This shows that the Ethiopian and Amhara region web portals are not linked. If using Linked data, not only the data can be located in Ethiopia and Amhara but also the in the UK dataset (remembering the war between Ethiopia and Britain in Mekdela and the reports about it by British) considering the Ethiopian and local data are linked to the Linked Open Data Cloud.

According to the FDRE government website ethiopia.gov.et[3], Ethiopia's government data is unique in different cases such as calendar, governmental structure, budget year, naming and others which brings the question of how this unique data can be represented in Linked data standard and linked to other datasets. The Ethiopian OGD needs to be modeled in a manner that can fit in the Linked Open Data Cloud. This work will be designing a model that can represent the Ethiopian OGD in the Linked Open Data Cloud and designing and implementing an automatic

[3] http://www.ethiopia.gov.et/about-ethiopia

mechanism that can understand the Ethiopian context of the data and link it to related datasets which can enhance the data.

1.4 Objectives

General Objective

Design of an automatic linker Open Government Data to the Linked Open Data Cloud.

Specific Objectives

- Analyze the existing Ethiopian OGD.
- Review related works in linking open government data to LODC.
- Model Linked Open Government Data of Ethiopia.
- Design appropriate algorithm to test the designed model.
- Develop a web application to demonstrate the designed model using sample data.
- Test and evaluate the demo web application and the data generated from the app.

1.5 Methods

The activities in this thesis work are classified under four main phases – Analysis, Design, Implementation and Evaluation.

A. Analysis

- Literature Review: For the purpose of systematically examining the core areas of this thesis work, we will
  - Review of Linked Data Concepts and Theories
  - Review of Existing and Matured Linked Open Government Data
  - Review of Linked Data Cloud

B. Design

- Designing of a model for Ethiopian Open Government Data.
- Formulating an algorithm for linking the designed Ethiopian OGD model to the existing Linked Open Data Cloud.
  - Devise steps to encompass every activity involved in the linking mechanism with clearly identified entry and exit points in each step.

C. Implementation

The devised linking algorithm along with some existing recommended tools will be applied on the sample data sets for demonstration purpose and evaluation of the algorithm.

D. Evaluation

The effectiveness and functionality of the proposed model and the algorithms will be evaluated by measuring the time the processes took and measuring the number of data retrieved by the proposed model.

1.6 Scope and Limitation of the Study

Scope

- This work focuses on designing a model for Ethiopian OGD and linking mechanism to the Linked Open Data cloud.
- It will use a selected set of available government data for testing and demonstration.

Limitation

- It will only consider data in English. It does not consider cross language datasets.
- It will not intend to publish Ethiopian Government data online since it is the duty of the federal and local government bodies to be accountable for it.

1.7 Application of Results

After the full-fledged implementation of this automatic linking mechanism:

- Data regarding Ethiopia government from different sources will have a mechanism to be linked and available for reuse.
- Ethiopian government can provide open government data to all citizens in standard machine readable format.
- Ethiopian citizens or concerned parties can obtain open government data easily.
- Researchers can investigate more possibilities to obtain open government data and reuse the data on their area of interest.

- Programmers will have a bridge to develop and run their Ethiopian OGD related application.

1.8 Organization of the Rest of the Thesis

This thesis is organized as follows. In Chapter 2, relevant literatures will be reviewed mainly on the background of open data, government data, linked open government data and linked open data cloud and how linked data can be published. In Chapter 3 the existing works that are related to this thesis will be presented. In Chapter 4, the design of a model for Ethiopian OGD and linking mechanism to the Linked Open Data cloud architected will be presented and elaborated. In Chapter 5, the implementation and discussion of the thesis will be presented according to the architecture designed in Chapter 4 and also the experiments done with the evaluation results will be presented.

Finally, in Chapter 6, the summary of the thesis will be presented with conclusion made on the thesis result. The contribution of the thesis and recommendation on possible future works related to the thesis will also be presented.

Chapter Two: Literature Review

2.1 Introduction

This chapter presents review of literature to provide adequate background information about the basic building blocks of this thesis. The review emphasizes the basics of open data, open government data, linked data, linked open government data and linked open data cloud. The review also includes how linked data works.

2.2 Open Data

Open data is data that can be used, re-used and redistributed freely by anyone. It is a way of obtaining, using and sharing data. Regarding cost of data must be cost free or on cost that can only cover its cost. Open data must be obtained in a format that is useful and easily modifiable. Furthermore, the data needs to be availed in manner that it can be easily reused, disseminated by creating a link with other datasets. When providing open data, identity of the individual or the company should not be criteria to grant or deny the data since all have equal rights to public data. If data provider needs payment for the data he/she provided, must ask all to pay or must not ask at all, since all have the same privilege to public data.

According to Daniel Dietrich et al, one of the main issues in open data is interoperability. Interoperability represents the ability of heterogeneous systems and organizations to work together towards using, processing and generating interlinked data. Interoperability can be vital since it enables different components to work towards the same or related data. In order to build a large and integrated system, it is mandatory to have interoperability among components. There must exist a common ground where these heterogeneous components or datasets to communicate. Openness not only creates interoperability among different components but also it can create a strength when more data is available. To fly an airplane, the pilots needs not only data about the technical capability of the airplane but also the weather where he/she is flying to, other health related issues about him/her or the current status of the location he/she is flying in case if there is transitive diseases or outbreaks or political uncertainty [12]. The more data available, the more informed the user can get, not to forget the impact of data overload.

2.3 Open Government & Open (Government) Data

In US, there was an Open Government movement in 2009, that was initiated by the then US president Barack who declared and signed "The Memorandum on Transparency and Open Government" (The Transparency Directive). And in the memorandum, it showed the commitment of the administration to create a great level of openness in Government while seeking transparency, public participation, and collaboration and believing Openness can promote democracy and effectiveness with in Government. Establishing a modern collaboration among politicians, public administration, industry and private citizens by enabling more transparency, democracy, participation and collaboration, is the main characteristics of Open Government. In the case European countries, Open Government and e-government are seen as peers collaborating each other.

On September 20, 2011, the Open Government Partnership was launched. And there were only eight founding governments (Brazil, Indonesia, Mexico, Norway, Philippines, South Africa, United Kingdom, United States). And these governments endorsed the Open Government declaration, and prepared their countries' action plans. And other 38 governments joined the partnership. At that time there were only 46 national government showing commitments to Open Government worldwide. Currently (February, 2020), the partnership has 78 nation members. Ethiopia is not one of them.

Free access to information and the possibility to freely use and re-use this information (e.g. data, content, etc.) are the most important enablers for Open Government. Without any transparent and valuable data, it is impossible to establish cooperation among stakeholders. Now a days, Open Government Data (OGD) is seen as the representation of Open Government.

Furthermore, OGD is now an international discussion point into avail open government data, making the data available to be both human and machine-readable, not in proprietary formats so that it can be re-used by any individuals who has the need. Openness of data is needed on government level but not in individual level. For individuals, the right to privacy should not be questioned by any means.

Some of the biggest companies in the world, are currently working on Open Data; the World Bank, the United Nations, REEEP, the New York Times, The Guardian and the Open Knowledge Foundation (OKFN).

In 2007, Sebastopol, California, USA, some Open Government advocates venue a meeting to setup a set of OGD principles that with reasoning OGD is essential for democracy. And a company named- the Sunlight Foundation expanded these principles to 10 principles. However, these principles are not rules that can be enforced rather globally recognized by the global open (government) data community as guidelines.

In order to consider a Government Data as "open", the data must be represented by the following below:

1. Completeness

   Data must be completeness so that it can be easily understandable and usable.

2. Primary Source

   It is recommended data to be published at source without any change or summary.

3. Timed Data

   Data can be valuable if only provided on time.

4. Accessibility

   Data must be available to be accessed by variety of users.

5. Machine-readable

   Data should be structured so that it can be easily reused without exhaustive conversion processes.

6. Non-discriminatory

   Data should be available to anyone who requested.

7. Non-proprietary

   Data should not be proprietary.

8. License-free

   Data should not be tied up by copyright, patent, trademark or trade secrets regulation. Reasonable privacy, security and privilege restrictions may be allowed as governed by other statutes.

9. Durability

   Data must be available for longer time.

10. Usage costs

    Data must be available for free or not greater than the cost spend to produce the data.

The worldwide Open Government Data movement initiated in Australia, New Zealand, Europe and North America. Today, there are solid accomplishments in Open Government Data in Asia, South America and Africa.

The European Commission (EC) has endorsed the Open Government Data issue giving it good priority and aggressively work toward OGD Europe [13].

2.4 Linked Data

The father of the WWW – Sir Tim Berners Lee has stated that the Semantic Web isn't just about putting data on the web, it is about making links. He implies that a person or a dedicated machine can browse the web of data. Regarding linked data, it provides the option to have one can data can have multiple related data [7].

The web of data is constructed with documents on the web just like web of hypertext. The web of data, they are links between subjective things described by RDF while in the web of hypertext, where links are relationships anchors in hypertext documents written in HTML. The URIs identify any kind of object or concept. But for HTML or RDF, the same expectations apply to make the web grow:

1. Use URIs as names for things
2. Use HTTP URIs so that people can look up those names.
3. When someone looks up a URI, provide useful information, using the standards (RDF, SPARQL)
4. Include links to other URIs. so that they can discover more things.

2.5 The Resource Description Framework (RDF)

According to Mike Bergman, RDF is a data model that is expressed as simple subject-predicate-object "triples". He also insisted that substitute verb for predicate and noun for subject and

object. It sounds very simple but it is how data can be easily represented and built up into more complex structures and stories. He also indicated that triple is like as a "statement" and is the basic "fact" or stated unit of knowledge in RDF. Matching the subjects or objects as "nodes" to one another (the predicates act as connectors or "edges") enables combining multiple statements. The aggregation of these node-edge-node triple statements appearing like a network structure is called RDF graph [25].

The most common RDF serialization formats are [14]:

- Turtle, a simple, human-readable format that at the end of each triple there is a full-stop point.
- N-Triples, a very simple, easy to trace or parse, line-based format but not as compact as Turtle.
- N-Quads, a superset of N-Triples, for serializing multiple RDF graphs.
- JSON-LD, a JSON-based serialization.
- N3 or Notation3, a non-standard serialization that is very similar to Turtle, but has some additional features, such as the ability to define inference rules.
- RDF/XML, an XML-based syntax that was the first standard format for serializing RDF.
- RDF/JSON, an alternative syntax for expressing RDF triples using a simple JSON notation.

Querying RDF

The most commonly used query language for RDF graphs is SPARQL. SPARQL is somehow like database SQL and it is the recommended one by W3C.

2.6 Publishing and Interlinking Structured Data

The best methods for publishing and interlinking structured data that can be easily accessed by both humans and machines via the use of the RDF (Resource Description Framework) family of standards for data interchange and SPARQL for query. For the existence of Linked Data, it is mandatory to apply the data model - RDF [14].

Hyland et al. [15] proposed a six steps approach in publishing and interlinking structured/Linked data. The main steps are:

1. Identify the nature the source data
2. Modeling the source data
3. Naming elements of the data
4. Describe the elements with URI
5. Convert the data into RDF
6. Publish the RDF

And maintaining is needed after publishing. This approach focuses on the quality of the data to be published.

Hausenblas et al [16], proposes a different six step Linked OGD publishing approaches that focuses on control over schema, data and data generation. Their steps are as follows:

1. Identifying the nature of the data
2. Modeling the data
3. Publishing the data
4. Searching for related data
5. Creating integration
6. Apply Use cases

This approach prioritizes publishing of data before linking to other datasets. This approach also applies use cases in linking data.

Villazón-Terrazas et al [17], proposed an iterative phases of publishing of Linked Open Government Data:

1. Specify the nature of the data
2. Model the data
3. Generate RDF from the model data

4. Publish RDF
5. Modify the RDF

This approach modifies the RDF data publishing and returns back to specifying the nature of the data to maintain the RDF.

According to Linked Open Data Cloud website[4], in order to publish RDF data one RDF dataset must contain at least 1000 triples and 50 external links to other LODC datasets.

[4] https://lod-cloud.net/

Chapter Three: Related Works

3.1 Introduction

In this Chapter, works that are related this are presented. To obtain the core points from the works they are categorized in to two: Open Government Data in Countries and linking mechanisms or approaches.

3.2 Open Government Data in Countries

3.2.1 Publishing Linked Data - The Pordata Use Case.

Pordata is an initiative of the Francisco Manuel dos Santos Foundation. The Foundation with a target of promoting knowledge about Portugal and Portuguese reality to citizens so that they can participate in the public debate and contribute to the development of the modern democratic society. The Pordata project was established with the goals to collect statistics about Portugal and publish them on the Web. So the paper proposed to extend the use of Pordata. And the authors adopted the existing vocabularies - Data Cube and Timeline Ontology, to develop the Pordata Vocabulary. The authors also indicate that there is ensured better interoperability of Pordata with other Linked Datasets that employed the same vocabularies. Technical realization of Linked Pordata was based on the Virtuoso Universal Server. They also proposed that to keep the existing Pordata data management infrastructure, transferring relational data into RDF is a better option. They generated RDF representation of Pordata in terms of Data Cube and Timeline Ontology using the Virtuoso RDB-to-RDF mappings and to deploy the Linked Pordata, they made Pordata URIs dereferenceable by means of the Virtuoso URL-rewriter [19].

3.2.2 Open Government Data in Brazil

The works focused on securing Open Government Data integration at a global level by promoting the use of standard RDF vocabularies. It urges to have adequate tooling since it is necessary during the triplification process to help users map local concepts to existing RDF vocabularies, and is used by other datasets in the LOD Cloud. As the authors mentioned they have invested in the development of user friendly tools that help users map contextualized information to standard vocabularies. Although a great part of the data is stored in relational databases (OLAP data cubes) to facilitate the manipulation of statistical data, the sources usually require some pre-processing before the conversion process can take place. The authors have maximum value for improving the

overall quality of the provenance provided, so as to increases user's trust and the overall reliability of the published datasets. Finally, the authors recommended that at multinational level, it would be a good strategy to create a centralized repository to capture, store, and annotate the RDF vocabularies used by different countries together with their mappings [20].

3.2.3 Linking UK Government Data

Sheridan and Tennison [11] focused on the case of using Linked Data standards for publishing open government data of UK and indicated how Linked Data standards uniquely allow governments to publish data responsibly and why responsible data publishing is so important to the open government data movement. The paper also explains how the Linked Data world was not quite ready for the large-scale adoption of these standards by a major government, leaving much to be done to develop practical approaches and patterns for the publishing of government data. From URIs, to provenance and versioning, through to statistics and geographic information, much thinking and work has been done. In each case the emphasis has been, not on research, but designing simple repeatable patterns, supported through tools. This work has also involved in building understanding and capability amongst officials from across government departments and agencies. It explains why the government's use of linked data standards was not universally welcomed and was even greeted by antagonism from some. Learning from this feedback the paper describes how we are now using linked data standards to enable government as a platform, commoditizing the process of creating APIs to meet the needs of a wide range of data consumers, from business, academia and the developer communities.

3.3 Linked Data Linking Mechanisms or Approaches

3.3.1 Scalable and Distributed Methods for Entity Matching, Consolidation and Disambiguation over Linked Data Corpora.

Aidan Hogan et al in the paper [21] focused on large-scale, static, Linked Data corpora and used scalable and distributed methods for entity consolidation (smushing, entity resolution, object consolidation, etc.) to locate and process names that signify the same entity. They also applied a baseline approach, which uses explicit owl:sameAs relations to perform consolidation. And also used extended entity consolidation which additionally uses a subset of OWL 2 RL/RDF rules to derive novel owl:sameAs relations through the semantics of inverse-functional properties,

functional-properties and (max-)cardinality restrictions with value one. The authors also applied deriving weighted concurrence measures between entities in the corpus based on shared inlinks/outlinks and attribute values using statistical analyses and disambiguating (initially) consolidated entities based on inconsistency detection using OWL 2 RL/RDF rules. The methods used are based upon distributed sorts and scans of the corpus, where it deliberately avoids the requirement for indexing all data. For evaluation of the algorithm they used a diverse Linked Data corpus consisting of 1.118 billion quadruples derived from a domain-agnostic, open crawl of 3.985 million RDF/XML.

3.3.2 An approach for managing and semantically enriching the publication of Linked Open Governmental Data

The proposed platform allows monitoring the whole process, providing information about the generated triples. Questions like "where are these data from?", "which process generated this data?" or even "which agency is responsible for these data?" can be answered using provenance tracking. The implementation of the platform in a modular approach, supports easier integration with other tools currently in use by government agencies. The development based on open source software is in accordance with the current guidelines of government information infrastructure in Brazil [22].

The semantic data annotation provides knowledge about the published data and the possibility to make reasoning on top of them. The linking phase allows interconnecting data assets from different agencies, providing a partial entity-centric integration across different datasets. Agencies publishing governmental data with the proposed approach can more easily control and monitor the publication process and also increase reuse of data transformation and publication tasks. From the citizen's point of view, data consumer applications can make use of integrated and standardized datasets with richer semantics.

3.3.3 Linked Data Generation for The University Data from Legacy Database

Arup Sarkar et al [23] proposed a framework for linked data generation by dividing the process in five modules namely:

1. Legacy data zone (DB) module
2. RDF dump generation module
3. Provenance handling module

4. Linked data publishing/generating components module
5. Linked Data zone module

The modules RDF dump generation, Provenance handling module, linked data publishing/ generating components hold the important feature of this framework.

Legacy data zone (DB) module preserves the legacy data in the context of our university's database holding the data related to their scope (University of Kalyani) in RDBMS form. The RDF-Dumper component module has two jobs. First it will read the available data of the database from the "Legacy data zone" and will generate the mapping file in N3 format. Its second job is the generation of the RDF dump file from the map file. After that this data will be delivered to the provenance handler that collects the dump file and adds the provenance related data to it. Then the modified dump file will be submitted to the Linked data publisher that will handle the content negotiation related problems and finally it will publish the available data from the modified dump file to the Web-of-Data as Linked Data.

3.4 Chapter Summary

The works mentioned in the Section 3.2 and 3.3, showed how government or any data can be published on the linked data cloud or their server. But they all focused on mainly publishing data on RDF format and how conversion can be done from legacy system to RDF. Most of the works uses variety of tools to perform the transition to Linked Data. From the works, we can witness that there doesn't exist an automatic method that handles all from source data gathering to readying publishable linked data. This thesis work is a full-fledged automatic linker that enables to automatically link a government data to be processed via one system that integrates variety of components to work together and perform their task without the involvement of users starting from imported source data to retrieving an extended data.

Chapter Four: Automatic OGD Linker to The LODC

In this chapter, the design of the proposed work is presented. It elaborates how the proposed design works in two sections. The chapter starts with the general overview of the proposed work. The second section elaborates how all the components of the proposed automatic OGD linker works.

4.1 Overview

As stated in section 1.4, the proposed work enables to gather Open Government Data (OGD) from different accountable governmental sources. And also processes the obtained OGD to make it ready for publishing on the Linked Open Data Cloud. The proposed work comprises various steps in order to ready the Open Government Data to the LODC. As one of the advantage of having this proposed work is to provide a simple interface for uploading OGD from the lower governmental administrative division to the highest administrative division (country level). The initial stage of the proposed work is master data configuration. The master data configuration includes assigning domains into domain specific ontologies, assigning of domain specific attributes into common attribute names. And also adding unique government data elements that can be possible RDF subject element into the domain ontology DB. For instance, Kebele is a unique data element that exists only in Ethiopia government data. According to Wikipedia[5], Kebele is the smallest administrative unit of Ethiopia, similar to a ward, a neighborhood or a localized and delimited group of people. Therefore, any unique government data must be added in the domain ontology DB so that it can easily be identified and linked to external datasets. And also the administrative division of Ethiopian Government must be included as part of configuration. These configurations ease the domain mapping process by providing meaning and directions.

The proposed work accepts different types of source data as input and from the data obtained removes duplicated data, invalid data (for instance, presenting data in wrong data type). Then forwards the cleansed data for domain mapping. The domain mapping process includes multiple sub processes starting from identifying the source and domain of the data based on the provenance data obtained from the input form. The other sub process in the domain mapping interpreting the data based on the vocabularies assigned to specific domains from the Domain Ontology database. For example, if the provided data is population of the country based on sex and region, and Region X has 2,000,000 males and 3,000,000 females and 5,000,000 both sexes. The both sexes data

[5] https://en.wikipedia.org/wiki/Kebele

element can also be interpreted as total population of the region without considering sex as a parameter. For this example, both sexes data can be interpreted as total population of Region X. This shows how the proposed works performs modeling of data automatically by referring to the domain ontology database.

The other vital process in the proposed work is triplification (converting data into subject-predicate-object format, the way data represented in linked data or RDF). It is mandatory to have a converter that converts the source data into RDF since most of the source data are represented in tabular data. Converting a tabular data into RDF triples does not have a generic formula since the conversion depends on the data itself. As mentioned on Section 1.3, in our case study, Ethiopian government offices do not have a uniform template or structure to present their data on the web. The nature of the data these government offices provide may vary but directly or indirectly the data mentions core data units like people and place. To perform the RDF conversion, the triplifier needs domain-mapped data, so that it can automatically decide on subject, predicate and object. After deciding on the triples, it generates RDF file having only triple per line. In this thesis, turtle is the selected RDF serialization. Replacing the triples into URI is the next task. To fulfill the criteria of being linked data is representing the data in URI format.

One of the main principles of Linked data is representing in URI format. In order to generate URI from the source data, the proposed work performs classification of data types of each data element in the source data. The classification eases the process of URI generation by isolating the data elements that can be represented in URI and the ones that cannot be represented in URI (for instance, literals) and also assigning data types to the literals. In the URI generation process, based on the domain mapping done contributes in deciding URI of a specific data element. For the example provided above, both sexes can be represented as region (http://data.ethiopia.gov/region[6]). After URI generated for all possible data elements of the source data, the source data file will be replaced with a file having a data represented in URI format.

After having a triple file where each possible data element is represented in URI format, data validation will be done as per the syntax of the selected RDF serialization (turtle-in our case). And the validated data will be stored in RDF dump file.

[6] The URI http://data.ethiopia.gov/region doesn't exist, it is just for example.

Converting the source data to RDF data is not enough to link to the Linked Open Data Cloud. The RDF dataset must have to link to external datasets as mentioned in section 2.1. Dataset file must fulfill the minimum requirement of LODC as stated in section 2.6 (1000 triples and 50 external links to other datasets). Data reconciliation among the RDF dataset generated and existing data in the LODC. To perform the data reconciliation, SPARQL query is used to extract similar data from all sources and when found, an owl:sameAs statements will be created. Since domain mapping is done for the dataset when querying for datasets in Linked Open Data Cloud, it simplifies the task of looking up for the external dataset, the owl:sameAs statements will be added to the RDF dump file. After the data reconciliation is done, the consolidated RDF dataset will be tested by substituting the URIs referring the same thing (it can be either subject, predicate or object). Then the tested/validated RDF dataset will be published locally. This dataset can only be accessed the server it is loaded. For globally accessing via other interfaces, the dataset must be uploaded on the Linked Open Data Cloud. The publishing of Linked Dataset on the Linked Open Data Cloud cannot be automatically done by the proposed work having in mind that the available Linked Open Data Cloud doesn't allow server side (FTP) access instead it provides a client side uploading option.

Figure 4.1 shows how the proposed performs the process flow from the initial stage of accepting source OGD to the target of publishing on the Linked Open Data Cloud.

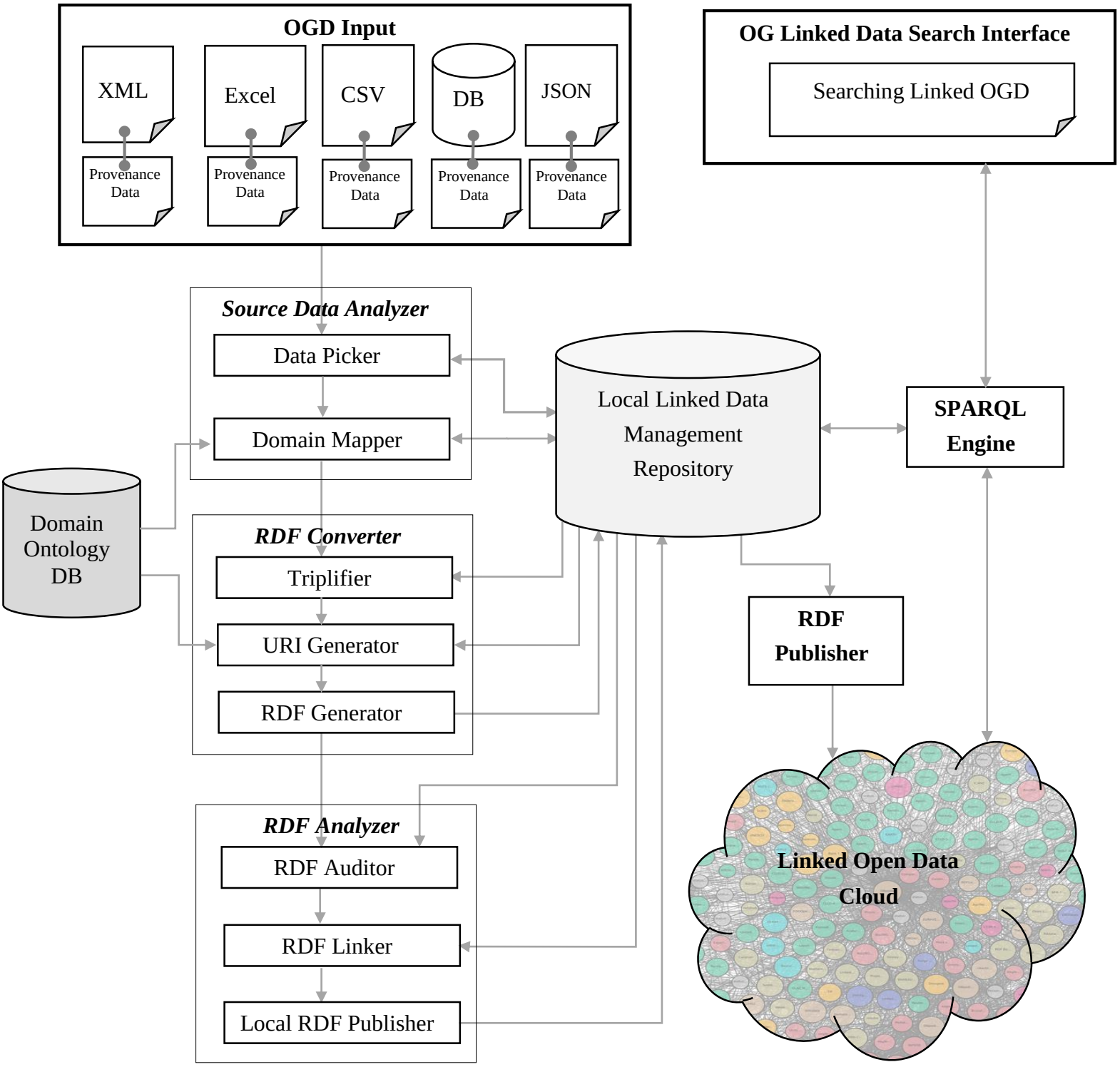

Figure 4.1 Design of Automatic OGD Linker to the LODC.

4.2 Components of The Proposed Work

The proposed work comprises seven main components:

1. OGD Input
2. Source Data Analyzer
3. RDF Converter
4. RDF Analyzer
5. RDF Publisher
6. Local Linked Data Repository
7. OG Linked Data Search Interface

Furthermore, the proposed work utilizes additional internal and external auxiliary components.

In our test case, the Ethiopian Open Government Data is presented in different formats on the web mainly on the willingness of the government offices handling the websites. There is no centralized website or portal that contains adequately large collection of open government data. It is therefore vital for our work to gather some of the available open government data in one box in order to model the Ethiopian open government data.

4.2.1 OGD Input

The proposed work accepts government data with different file types. The proposed work only allows tabular data which can easily be extracted and transformed into at least 3-star linked data. As stated in Section 1.1, even though, there is no big collection of government data (in our case Ethiopia) available on the web. So, the proposed work provides an interface to upload, transform and link OGD on the web. The proposed work only accepts:

- Spreadsheet files (.xlsx)
- Comma Separated Values file (.csv)
- XML file (.xml)
- SQLite Database file (.db)
- JSON file (.json)

While accepting these source data, it is mandatory to enter their provenance data in parallel to increase the authenticity of the data and to simplify the identification of the domain of the data.

The provenance data includes information about the OGD itself, the provider of the data, domain of the data, and description of the data.

4.2.2 Source Data Analyzer

Source Data Analyzer receives OGD in different formats and passes it through three processes. The three major tasks of Source Data Analyzer are namely, identify the file format and parts of the data, picking only the correct data, identifying the domain of the data.

Identifying the File Format of the Dataset File

Identifying the file type or format enables the proposed work to decide on how to the triplification process can be done. Identifying the file format can easily be done at the submitting the dataset file by only taking the file name extension.

Picking Only the Correct Data

One of the main task of Source Data Analyzer is to remove duplication from the source data as stated in the Algorithm 4.1 and it is mandatory for the uploader of the source data to know that the source data must have uniformity or consistency. For instance, if the data is in tabular format, the data must be presented in uniform number of columns – rows structure. Including title, source or date information of the data may trick this work's effort in processes like triplification process.

Algorithm 4.1: Algorithm for Picking Correct Data

```
Input: Source File
Task: Pick Correct Data Only
   Start
   if Source File Name Extension is "xlsx" or "xls" then
      Declare string array data[][]
      Declare integer variable rowCount with initial value 0
      Declare integer variable i with initial value 0
      // Assigning file content to multi-dimensional array data[][]
      for each row in the rows do
           j← 0
```

```
            for each cell in row do
                  data[i][j]← cell.value
                  j ← j+1
            end for
            i ← i +1
            rowCount ← rowCount +1
      end for
      // Removing duplicate rows
      for i ← 0 to rowCount do
            for j ← i do
                  if row data[i][*] is identical with data[j][*] then
                        rowCount ← rowCount -1
                        for k ← 0 to rowCount do
                              data[k][*] ← data[k+1][*]
                        end for
                        i ← i – 1
            j ← j + 1
            end for
      i ← i + 1
      end for
      Replace file content with values data[][]
   else if Source File Name Extension is “csv” then
      Declare string array data[]
      Declare integer variable rowCount with initial value 0
```

```
      Declare integer variable i with initial value 0
      for each line in the lines do
            data[i] ← line string
            i ← i+1
            rowCount ← rowCount +1
      end for
      // Removing duplicate rows
      for i ← 0 to rowCount do
            for j ← i do
                  if row data[i]is identical to row data[j] then
                        rowCount ← rowCount -1
                        for k ← 0 to rowCount do
                              data[k] ← data[k+1]
                        end for
                        i ← i – 1
            j ← j + 1
            end for
      i ← i + 1
      end for
      Replace file content with values data []
save file
End
```

If the source file is either XML or JSON, the duplication check will be done in the triplication process since both XML and JSON are very close to RDF triple format.

Domain Mapping

To avoid linking one specific data to a wrong one can be avoided primarily if domain mapping is done before any linking processes. For instance, the word "fine" has several meanings, including two different adjectives. First, it can be used to describe something as high quality and second, it can describe something especially thin. As a noun, "fine" means a payment for a violation.

The possible domains a government can provide data on are related to either one or more of these (National Security, Foreign Affairs, Finance, Justice, Agriculture, Trade, Industry, Technology, Transportation, Urban Development, Construction, Natural Resources, Energy, Irrigation, Education, Health, Women, Children, Youth, Labor, Social Affairs, Culture, Tourism, Revenues, Population, Sport, Election, Investment, Migration, Media, etc.). Therefore, the Source Data Analyzer performs checking on the provenance data to identify the domain of the data as seen Algorithm 4.2.

Algorithm 4.2: Algorithm for Domain Mapping

```
Input: Source_File, Provenance_Data[]
Task: Identifying Domain of Data
   Start
   Declare string variable origin
   Declare string variable domain
   Declare string array ontology[]
   Declare string array ontologyKeywords
   origin ← Provenance Data [origin]
   domain ← Provenance Data [domain]
   ontology []← standard domain ontologies // availed by the proposed work
   domainKeywords[][] ← domain with their keywords // availed by the proposed work
   if domain is known then // when uploader specified domain
```

```
    for each dom in ontology do
        if domain is similar to dom then
            return dom
            // if domain found easily return domain
    end for
  else
    Declare integer variable foundCount with initial value 0
    Declare integer variable foundMax with initial value 0
    Declare integer variable foundDomain with initial value -1
    for each domn in domainKeywords do
        foundCount ← 0
        for each keyword in domn do
            Read File as Text
            for each word in File to end of file
                if domainKeywords[domn][keyword] is similar
                with word then
                    foundCount ← foundCount + 1
            end for
        end for
        if foundCount > foundMax then
            foundDomain ← domn
            foundMax ← foundCount + 1
    end for
    if foundMax is 0 then
        return "unknown"
```

```
      else
            return foundDomain
End
```

At the time of uploading the source data specifying the domain is expected from the uploader but if not the proposed work uses its dedicated domain-based ontology library to compare the data content of the source data with keywords in domain-based ontology library.

Knowing the origin of the data and finally settling its domain can minimize the effort to integrate to another sources.

4.2.3 RDF Converter

In order to join the web of data and extend the value of an open government data, it is mandatory to present the data in a standard format of RDF. There are different serializations of RDF and for this work "Turtle" serialization is selected.

To accelerate the RDF conversion of the existing OGD the following processes will be done:

- Triplifier: To model the data in RDF triple (Subject, Predicate, and Object) format without losing meaning or context the original data.
- URI Generator: To name every part of the triple in URIs except literal values and null valued ones.
- RDF Generator: To put everything triple produced from the source data and dump to one RDF Dump file.

Triplifier

Whenever a source data is received, it is kept in its original format with its provenance data and having minor issues like duplication are being fixed in the source data analyzer component. After having a duplicate free data, the triplier performs modeling of the data based on the original format of the source data.

Triplification of data in Excel or CSV or db format:

Both excel and csv format are tabular data representation. In order to convert tabular data into RDF format the source and structure of the data must be analyzed and the following questions must be answered.

- Which part of the tabular data can be either subject, predicate, or object?
- Is the data selected as object a literal or blank or an object representing another resource?

For instance, we have the following example tabular data. Let's say X is a region in Country Y and the data below is the population of the cities in region X.

Table 4.1: Sample Population Data

City	Male	Female	Total
City1	1,000,000	1,200,000	2,200,000
City2	2,500,000	2,458,000	4,958,000
City3	1,750,000	1,900,000	3,650,000

The steps in converting the above data in Table 4.1, into RDF triple (triplification) is started by stating the relationship between the source data owner (i.e. Region X) and first column data (i.e. City). And that gives the context of City names in Column 1 are cities found in region X when putted in RDF triple format having in mind that Region X is also stated as a region in Country Y initially.

Subject	**Predicate**	**Object/Literal**
Country_Y	Is_Kind_Of	Country
Region_X	Is_Kind_Of	Region
Region_X	Country	Country_Y
City1	Is_Kind_Of	City
City1	Region	Region_X

The next step is confirming the first column except the first cell of the column as subject for the remaining triples to be generated. And also confirming the first row as predicate for the remaining triples to be generated. We can identify the first column data as Row Headers and the first Row Data as Column Headers.

Row Headers: City1, City2, and City3

Column Headers: Male, Female, Total

At the time of setting a tabular data value as a predicate, the proposed work checks for a better naming scheme for the predicate by the learning the domain and origin of the data. This can reduce ambiguity at the time of linking the data to the Linked Open Data Cloud. For the example above: male, female and total are the main predicates but they are not descriptive enough to be a predicate for the data, so they can be replaced by male_population, female_population, and total_population.

The next step will be joining Row Headers (Subjects) and Column Headers(Predicates) and assigning them corresponding Object data from the tabular source data.

Subject	**Predicate**	**Object/Literal**
Country_Y	Is_Kind_Of	Country
Region_X	Is_Kind_Of	Region
Region_X	Country	Country_Y
City1	Is_Kind_Of	City
City1	Region	Region_X
City1	Male_Population	1,000,000
City1	Female_Population	1,200,000
City1	Total_Population	2,200,000

`And the same will be done for the remaining rows of data.

Subject	**Predicate**	**Object/Literal**
City2	Is_Kind_Of	City
City2	Region	Region_X
City2	Male_Population	2,500,000
City2	Female_Population	2,458,000

City2	Total_Population	4,958,000
City3	Is_Kind_Of	City
City3	City	Region_X
City3	Male_Population	1,750,000
City3	Female_Population	1,900,000
City3	Total_Population	3,650,000

From the above example and steps, we can understand that identifying the best data to be subject, predicate or object from the whole data is the main task. In the above example the first column data and the first data are selected as subjects and predicates respectively. But that may not work for other tabular data. For others the process is the same but the subjects and predicates may not appear on the first column or first row. For this case, let's see an example tabular data about Tourism Offices information of Country Y.

Table 4.2: Sample Government Data

Region	City	Tour Office	Email	Telephone
X	City1	City 1 Tour Office	c1_tour_office@y.gov	1xxxxxxxxx
X	City2	City 2 Tour Office	c2_tour_office@y.gov	2xxxxxxxxx
Z	City5	City 5 Tour Office	c5_tour_office@y.gov	5xxxxxxxxx
G	City4	City 4 Tour Office	c4_tour_office@y.gov	4xxxxxxxxx

For the sample data in Table 4.2, the best fit to be a subject is not only first column but also the second and third column.

Subject	**Predicate**	**Object/Literal**
Region_X	Country	Country_Y
City1	Region	Region_X
City1	Tour_Office_name	City 1 Tour Office
City1_Tour_office	Email	c1_tour_office@y.gov

```
City1_Tour_office   Telephone          1xxxxxxxxxx
```

To entertain both examples seen above, this work provides an approach as indicated in Algorithm 4.3 of selecting the right subject, predicate and object combinations by referring existing domain libraries. These libraries reside in the Local LD Repository and contains possible subjects, predicates, and objects. And also based on the data type of the data given.

Algorithm 4.3: Algorithm for Triplification of Tabular Data into RDF

```
Input: Source Tabular Data (CSV, Excel or DB File), Provenance_Data
Task: Convert Tabular Data to RDF Triples
   Start
   Declare string array table[][]
   Declare integer variable column_count with initial value 0
   Declare integer variable row_count with initial value 0
   Declare string array subjects[]
   Declare string array predicates[]
   Declare string array objects[]
   Declare string variable source
   Declare string variable source_type
   source ← Provenance_Data[source]
   source_type ← Provenance_Data[source_type]
   Read Tabular File to End of File do // row based reading
      col ← 0
      Read Row to Last Column do
           table[rows][col]← Current Cell Value
           col ← col + 1
```

```
    row_count ← row_count + 1
    if(col>column_count) then
       column_count ← col
    Declare integer variable k with initial value 0
    for i=1 to row_count do
       subject[k]← table[i][0]
       predicate[k] ← source_type
       object[k] ← source
       k ← k + 1
       for j=1 to column_count do
            subject[k]← table[i][0]
            predicate[k] ← table[0][j]
            object[k] ← table[i][j]
            k ← k + 1
       end for
    end for
Return subject[], predicate[],object[]
End
```

Triplification of JSON and XML file

Since JSON and XML are taken as RDF serializations it only requires context data to be added to them. So that the JSON data to be transformed to JSON-LD serialization and the XML file to RDF/XML as seen in Algorithm 4.4. Since our selected serialization format is turtle, this proposed work uses existing JSON-LD to Turtle and RDF/XML to Turtle serialization conversion tool. Domain information of the source data will be used to easily add context to the source files.

Algorithm 4.4: Algorithm for Triplification of JSON/XML Files

```
Input: Source JSON/XML File, Provenance_Data

Task: Convert JSON/XML file to Turtle

   Start

   Read JSON/XML File

   Declare integer variable domain

   Declare string array triples[]

   domain ← Provenance_Data[domain]

   Read Source_File to End of File

      triples[]← Convert_to_Triples(Source_File,domain)
   Return Triples

End
```

URI Generator

As mentioned in section 2.4, naming things in URI is the first principle of linked data. In this work, after performing triplification of the source data, there will be triples identified as subject, predicate and object. As seen in Algorithm 4.6, in order to assign URI for this triple elements, the following tasks will be done.

- Use hosting server http address as URI to provide prefix mainly for the subject and object elements.
- For predicates, based on the domain of the source data, URI can be obtained from Ontology libraries reside on the Local LD Repository that are pre-asssigned for specific domain of data.

Algorithm 4.5: Algorithm for URI Generator

```
Input:  Subjects[],  Predicates[],  Objects[],  Provenance_Data,
Ontology

Task: Assign URI for triples

   Start

   Declare integer variable triple_count
```

```
   Declare string variable domain
   Declare string owner_URI
   triple_count ← length of Subjects[]
   domain ← Provenance_Data[domain]
   owner_URI ← host address of the server
   for i=0 to triple_count do
      if domain then
           for k=0 to length of Ontology[domain][] do
                if Predicates[i]== Ontology[domain][k] then
                     Predicates[i]← Ontology[domain][k].URI
                     break
           end for
   end for
   Return Subjects[], Predicates[], Objects[]
End
```

RDF Generator

After naming things in URI and having a triple data, the next task will be putting all the triple data in RDF file with turtle serialization format. In this subcomponent RDF Generator as specified in Algorithm 4.7, RDF dump file will be generated after concatenating subject, predicate and object by using separator characters like space (separating subject, predicate and object) and full-stop (separating one triple from another).

Algorithm 4.6: Algorithm for RDF Generator

```
Input: Subjects[], Predicates[], Objects[]
Task: Create RDF Dump File
   Start
   Declare integer variable triple_count
```

```
   triple_count ← length of Subjects[]
   Create an RDF Dump file: RDF_DUMP_FILE


   for i=0 to triple_count do
      Write on RDF_DUMP_FILE (Subjects[i], " ", Predicates[i], "
", Objects[i], "."
      New Line
   end for
   Save and Close RDF_DUMP_FILE
End
```

4.2.4 RDF Analyzer

The basic paradigm of linked data is to transform the use of "web of documents" to "web of data". So far we have only RDF dump file that contains different triples of OGD but not linked to existing external sources in LODC.

RDF Analyzer has three sub components to perform the transition from RDF Dump file into ready to publish Linked Data.

RDF Auditor

During dumping all triple data into RDF dump file, duplication of data may happen from different causes. But duplication of data cannot happen from one origin cannot happen since it is resolved in the Source Data Analyzer. For instance, in our case study, Ethiopia, we have observed that duplication can happen from misspelling of data like Addis Abeba and Addis Ababa. Other cause for duplication describing one instance into multiple ways like Ethiopia and Federal Democratic Republic of Ethiopia. This kind of issues can be resolved by configuration where this allows users to provide multiple naming for a specific thing to enable auto-correcting while processing of data. And also these wrongfully encoded data can be captured by RDF Auditor when it checks the domain of the data and the source of the data and the governmental administrative division.

Algorithm 4.7: Algorithm for RDF Auditor

```
Input: RDF_DUMP_File, Subjects[], Predicates[], Objects[]
Task: Auditing RDF_DUMP_File
   Start
   Declare integer variable triple_count
   triple_count ← length of Subjects[]
   for i=0 to triple_count-1 do
      for j=i+1 to triple_count do
           if Similarity(Subjects[i], Subjects[j]) then
                Subjects[i]← Similarity(Subjects[i], Subjects[j])
                Subjects[j]← Similarity(Subjects[i], Subjects[j])
           if Similarity(Subjects[i], Objects[j]) then
                Subjects[i] ← Similarity(Subjects[i], Objects[j])
                Objects[j] ← Similarity(Subjects[i], Objects[j])
      end for
   end for
   for i=0 to triple_count-1 do
      for j=i+1 to triple_count do
           if Similarity(Objects[i], Objects[j]) then
                Objects[i]← Similarity(Objects[i], Objects[j])
                Objects[j]← Similarity(Objects[i], Objects[j])
           if Similarity(Objects[i], Subjects[j]) then
                Objects[i] ← Similarity(Objects[i], Objects[j])
                Subjects[j] ← Similarity(Objects[i], Subjects[j])
      end for
```

```
   end for


   for i=0 to triple_count-1 do
      for j=i+1 to triple_count do
           if Similarity(Subjects[i], Subjects[j]) and
           Similarity(Objects[i], Objects[j]) then
                remove_triple(j)
      end for
   end for
   Open RDF_DUMP_File
      for i=0 to triple_count do
           Write on RDF_DUMP_FILE (Subjects[i], " ",
           Predicates[i], " ", Objects[i], "."
      New Line
   end for
   Save and Close RDF_DUMP_FILE
End
```

RDF Linker

The RDF Linker performs searching of similar data element from by using SPARQL query. The query can easily be done since the domain, the vocabulary used in the predicate part and the RDF triples are already settled. For instance, to find a similar dataset about the Addis Ababa City Population in the Linked Open Data Cloud, the term City and the domain population plays key roles in finding the correct URI or linking to that other datasets in the cloud.

Whenever a result is obtained from the SPARQL query, selecting the best result is another task. When selecting the best result, major data value comparison appears one option. The other one is

to find best result for one part of the RDF triple by the remaining ones (for instance, finding external link for the Subject data element by the help of corresponding data elements both Predicate and Object).

After selecting the best result, how the result and the data element from RDF dump file can be linked is the biggest question to answer. Since both the result and the data element from RDF dump file have similar value, they can be connected by the owl property that is known as owl:sameAs. For example, linking the resource Ethiopia with another resource named Abyssinia as shown below.

```
<rdf:Description rdf:about="#Ethiopia">
        <owl:sameAs rdf:resource="#Abyssinia"/>
</rdf:Description>
```

For every similarity results found, additional RDF triples will be created in order to link the source data and the data in the Link Open Data Cloud. And those triples will be added to the RDF dump file and saved with different file name for differentiation purposes.

Algorithm 4.8: Algorithm for RDF Linker

```
Input: RDF Dump File, Provenance_Data[]
Task: Link RDF_DUMP_File
   Start
   Read RDF_DUMP_File
   Declare string variable domain
   Declare string variable subject
   Declare string variable predicate
   Declare string variable object
   Declare text variable linked_data
   Read RDF_DUMP_File from Start to End of Line
      //Perform SPARQL Query based
      subject← extract_string_separated_by_space(0)
```

```
      subject2←subject
      predicate← extract_string_separated_by_space(1)
      predicate2←predicate
      object← extract_string_separated_by_space(2)
      object2←object
      if SPARQL_LODC_search(Subject) then
           /*  SPARQL_LODC_search(Subject)returns  URI  for  best
         match from LODC */
           subject2←SPARQL_LODC_search(Subject)
      if SPARQL_LODC_search(Predicate) then
           /*  SPARQL_LODC_search(Predicate)returns  URI  for  best
         match from LODC */
           predicate2←SPARQL_LODC_search(Predicate)
      if SPARQL_LODC_search(Object) then
           /* SPARQL_LODC_search(Object)returns URI for best match
         from LODC */
           object2←SPARQL_LODC_search(Object)
      linked_data ← concatenate(linked_data,subject2," ",
      predicate2," ",object2
      Next Line
   Write linked_data at End of RDF_DUMP_File
   Save and Close RDF_DUMP_File
End
```

Local RDF Publisher

Since uploading the source data in a specific format to having an RDF dump file with link to external datasets in the LODC, all progress files are available and can be used. To enable access for the RDF file and source and progress files, it is vital avail these data via website. When said Local RDF Publisher, it is an option to publish and access RDF files with or without external servers' involvement. The local RDF publishing can be done on the server of the OGD owner.

It is the recommendation of this work to verify the quality of the data locally before publishing it on the LODC. This also can speed up the searching process since the repository holds its own copy of open government data.

4.2.5 RDF Publisher

After having a clean and reconciled final RDF file, the next main task that needs to be done is publishing the RDF file on the Linked Open Data Cloud.

In order to publish the OGD in the Linked Open Data Cloud, the following will be checked:

- The RDF file must have at least 1000 triples.
- The RDF file must have at least 50 links to external sources found in the LODC.
- The RDF file must be in standard RDF serializations format (in this work, turtle serialization).

After the above requirements are fulfilled, the RDF publisher component ready the RDF file for publishing. In this work, publishing the RDF file on the LODC is handled manually by the owner of the OGD since there is no option to have a server side to the LODC.

Algorithm 4.9: Algorithm for RDF Publisher

```
Input: Subjects[], Predicates[], Objects[]

Task: Verify RDF Dump File for LODC Publisher

   Start

   Declare integer variable triple_count

   Declare integer variable link_count

   triple_count ← length of Subjects[]

   link_count ← 0
```

```
   if triple_count < 1000 then
      Display "Dataset Not Adequate to Publish"
   else
      for i=0 to triple_count do
           if Subjects[i].prefix Not Local_Server then
                link_count ← link_count + 1
      end for
      if link_count < 50 then
           Display "Dataset Not Adequate to Publish"
      else
           Display "Dataset Ready for Publishing!"
End
```

4.2.6 Local Linked Data Repository

For this work, there is a Local Linked Data Repository is a server that can store variety of files starting from the source file to the final ready-to-publish RDF dataset. The main purpose of the repository is to store:

- Configuration Data: stores configuration data that represents the administrative organization of the specific government data. Setup of administrative organization of the government on the proposed system works as a guideline for uploading and managing open government data.
- Source Data: stores the source data file uploaded by the user.
- Provenance Data: stores the supporting information about the data like domain of the data, uploader information, etc.
- Progress Data: Throughout the processing of the source data file to become ready-to-publish dataset, different files generated in every phase of the process. Those files are also stored in Local LD Repository.

- Status Log of entered OGD: Every progress status of the process is recorded in Local LD Repository.
- Libraries: Ontology Libraries and references used.
- Source Code/System: Source Code of the system used to process the data.

The list things above are stored in the Local LD Repository in a form of file and database.

4.2.6 OG Linked Data Search Interface

This work provides a user interface to search open government data from published OGD. The user interface enables users to provide a search key and it provides list of available URIs that are related to the search key. The searching can be done into ways:

- Using Search Key

  It considers the search key as Subject or Object and put it in SPARQL query. And return found results from both Linked Open Data Cloud and Local Linked Data Repository.

- Using SPARQL Query Statement:

  It takes the SPARQL query directly. And return found results from both Linked Open Data Cloud and Local Linked Data Repository.

Chapter Five: Implementation and Evaluation

In this Chapter, the implementation and evaluation of the proposed work for linking Open Government Data is elaborated. In the first Section, we will discuss the implementation detail by describing each component of the proposed work and how they are implemented. Different Scenarios are presented in order to validate the functionalities of the work in the second section. The evaluation of the approach introduced in this work is presented in the final section of the chapter.

5.1 Overview

The implementation process and the demonstration of the proposed work – this work enables full-fledged management of Linked Open Government Data. Starting from capturing OGD in different file formats, it cleans the data from duplication and groups the data in its specific domain. Then it converts the source file into Turtle (RDF). It puts everything in RDF Dump. This work invokes EasyRdf functions to perform the serialization process. This work cleans the data by editing/removing wrongly represented data URI. Linking the processed RDF dataset with external dataset is other major task of the work. It fetches URI representation of each element in the RDF dump from externals sources (Linked Open Data Cloud). Multiple representation of one data can help the data to elaborated more. But these multiple representations need reconciliation.

5.2 Development Environment

Several tools are used for the purpose of implementing the proposed work. The following are list of programming languages, libraries, database management tools, Operating System and others.

- **PHP**

PHP programming language is used to write the implementation of the proposed work.

PHP[7] (recursive acronym for PHP: Hypertext Preprocessor) is a widely-used open source general-purpose scripting language that is especially suited for web development and can be embedded into HTML. PHP generally runs on a web server. Any PHP code in a requested file is executed by the PHP runtime, usually to create dynamic web page content or dynamic images used on websites or elsewhere,

[7] https://www.php.net

In this work, PHP is selected because it can also be used for command-line scripting and client-side graphical user interface (GUI) applications. PHP can be deployed on most web servers, many operating systems and platforms, and can be used with many relational database management systems (RDBMS) and mainly in linked data case it can used with all types RDF serializations. And also because Most web hosting providers support PHP for use by their clients. It is available free of charge, and having the spirit of openness even the PHP Group (the owner of PHP) provides the complete source code for users to build, customize and extend for their own use. In this work, latest PHP version (7.4) is used.

- **EasyRdf**

EasyRdf[8] is a PHP library designed to make it easy to consume and produce RDF. It is an Object Oriented PHP Library. After parsing EasyRdf builds up a graph of PHP objects that can then be walked around to get the data to be placed on the page.

For implementing this work, some functionalities of EasyRdf are utilized. For instance:

- RDF Serialization
- Searching Data via SPARQL Query

- **MySQL**

MySQL[9] is the world's most popular open source database. With its proven performance, reliability, and ease-of-use, MySQL has become the leading database choice for web-based applications, used by high profile web properties including Facebook, Twitter, YouTube, and all five of the top five websites.

In this work, MySQL is used in order to keep the status log of data uploaded in the system.

- **PHPUnit**

  PHPUnit[10] is a unit testing framework for the PHP programming language. In this work, PHPUnit is used to debug the sub modules of the proposed work.

- **WampServer**

[8] https:///www.easyrdf.org
[9] https://www.oracle.com/mysql/
[10] https://phpunit.de/

WampServer[11] is a Web development platform on Windows that allows to create dynamic Web applications with Apache2, PHP, MySQL and MariaDB. WampServer automatically installs everything that are needed for Web applications.

WampServer is used as the main webserver for the proposed work.

- **Windows**

Windows is one of the leading Operating System in the world. The implement of this work is done on a PC running Windows 10 operating system.

5.3 Configuration of The Proposed Work

To make this proposed work function effectively it has the following configurations

- Master Data of the Government Data is needed.
    - It is recorded in the MySQL database – OGD and it includes:
        - Administrative Division Setup
        - Common government data templates
        - Data type of each data
        - Validation of each data
- OGD Domain and Related Vocabularies Assignment is needed.
- Operating System is not an issue for the proposed work since it is works as web application. For testing, it is currently works on Windows 10.
- Apache Server Setup is needed.
- PHP Server Setup is needed.
- MySQL Server Setup is needed.

5.4 Implementation Details

In the previous section, Section 5.3, the general setup used is elaborated. In order to test this application, the implementation detail of the components is presented below in Table 3.1.

[11] https://wampserver.com

Table 5.1 Implementation Details of Components

Component Name	Implementation Detail
Source Data Collector	▪ Provide interface for uploading source data ▪ Accept source data from user via POST method and Forward to Source Data Analyzer
Source Data Recorder	• Accepts the imported data as it is and • Save the data file on the Server and record the status in source data log table by making the date and file name as unique identifier of the data.
Data Picker	• Always check clean – 0 (Not clean) flagged data from source data log. • Open the file • Put the data on temporary data table. • Remove duplicates from the data. • Replace the file after cleaning • Remove data from temporary data table.
Domain Mapper	• Always check Domain – 0 (No Domain) flagged data from source data log. • Read provenance data • Open the source file • Compare the data on the source file with Standard Vocabularies based on the provenance data. • Assign domain for the source file on the source data log table.
URI Master	• Always check URI – 0 (No URI) flagged data from source data log. • Accepts Non-RDF data

Component Name	Implementation Detail
	• Based on the assigned domain, it compared each element of the data with vocabularies except literal values. Assign URI for each element and save it in a separate triple file. • Assign the triple file name to the source data.
Triplifier	• Always check RDF – 0 (No RDF) flagged data from source data log. • Accepts Non-RDF file. • Create RDF Triples based on the assigned domain vocabularies. • Save as Turtle files. • Assign the Turtle file names to the source data.
RDF Cleaner	• Always check RDFCln– 0 (No RDF Clean) flagged data from source data log. • Accepts RDF file. • Remove duplicate triples. • Save Cleaned RDF file.
RDF Auditor	• Always check RDFAdt– 0 (No RDF Audit) flagged data from source data log. • Accepts RDF file. • Checks every part of the triples from the RDF file for duplication and misrepresentation errors. • Fix errors • Save Audited RDF File
RDF Linker	• Always check RDFLnk-0 (No RDF Link) flagged from source data log. • Compares every part of the triples from the RDF file with LODC using SPARQL query.

Component Name	Implementation Detail
	• If found similarity, pick the link from LODC and add a new triple using the link from LODC. • Save Reconciled RDF

Component Name	Implementation Detail
RDF Publisher	• Always check RDFPub– 0 (Not Published RDF) flagged data from source data log. • Check whether the Data is ready for publishing with criteria of 1000 or more triples and 50 or more link to external datasets in the LODC. • Ready the file for publishing.

OG Linked Data Finder	• Provide an interface to Search data
	• Accept Search Query from User either
	  ○ Search Key
	  ○ SPARQL Query
	• Processes both types of Queries using SPAQL and both from LODC

5.5 Demonstration

For demonstration purposes, we have uploaded Ethiopian population census data, that is obtained from the website CSA (Central Statistics Agency) of Ethiopia. The uploaded data is in excel (.xlsx) format. And processed the data uploaded as per the implementation details mentioned in Section 5.4. To verify whether the uploaded data is linked to other external datasets or not, searching data can be done in the interface available for searching. And the system delivered four directly connected URIs. One result is from the Local LD Repository and the other are from the Linked Open Data Cloud.

- **OGD Input Dataset**

  In this interface (Figure 5.1), the proposed work enables to upload government data files. It only accepts XML, JSON, XLSX (XLS), CSV, and DB file formats. It rejects non tabular text data. And also it demands the user to provide header information with the file that is named as provenance data.

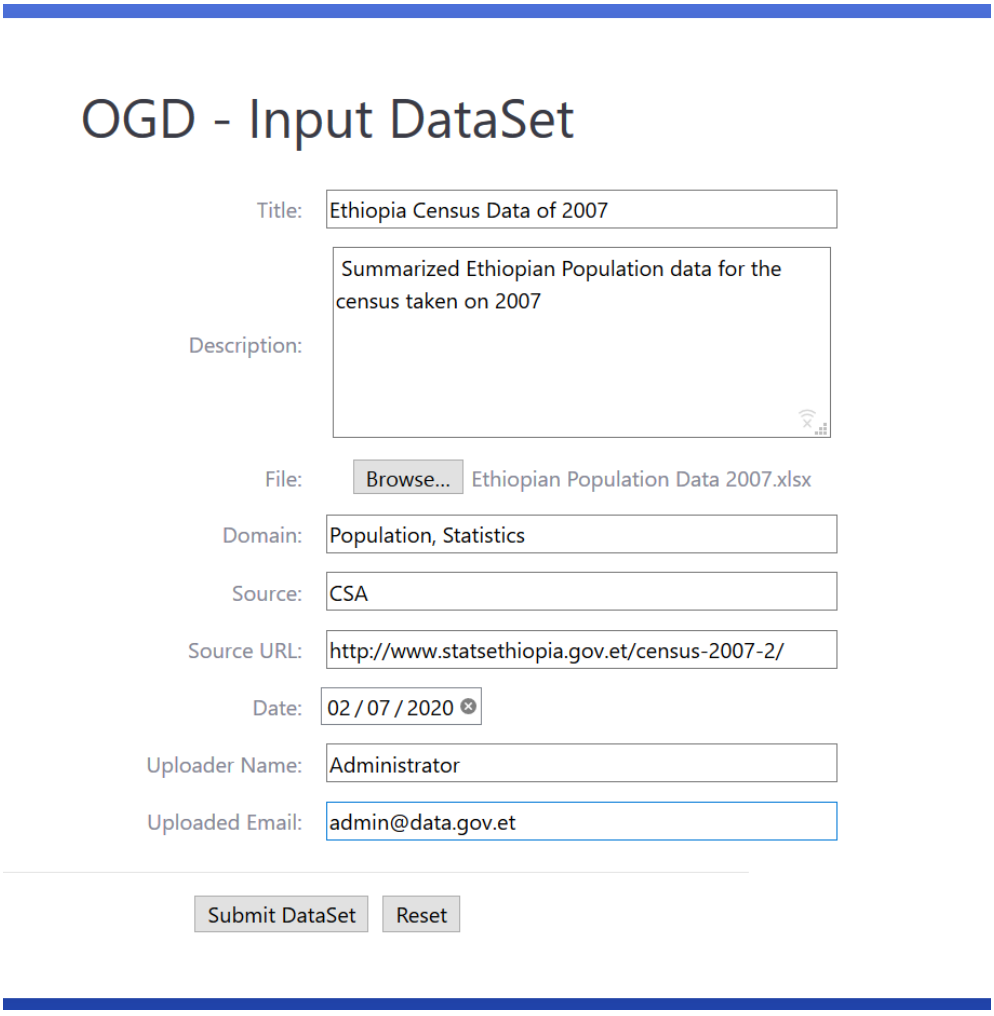

Figure 5.1: OG Data Input Interface

- **Search Interface**

  In this interface (Figure 5.2), the proposed work enables to search data from Linked data sources via two options:

  - Searching by writing keywords
  - Searching by writing a SPARQL SELECT Query

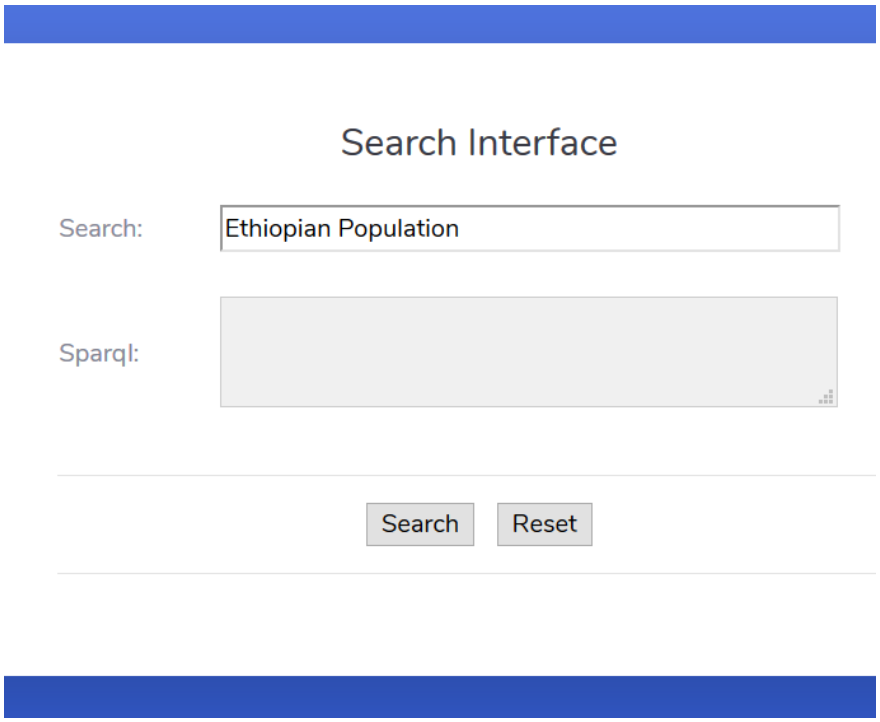

Figure 5.2 OG Linked Data Search Interface

After the Search button is selected, the proposed work performs searching and generates the results like shown in Figure 5.3.

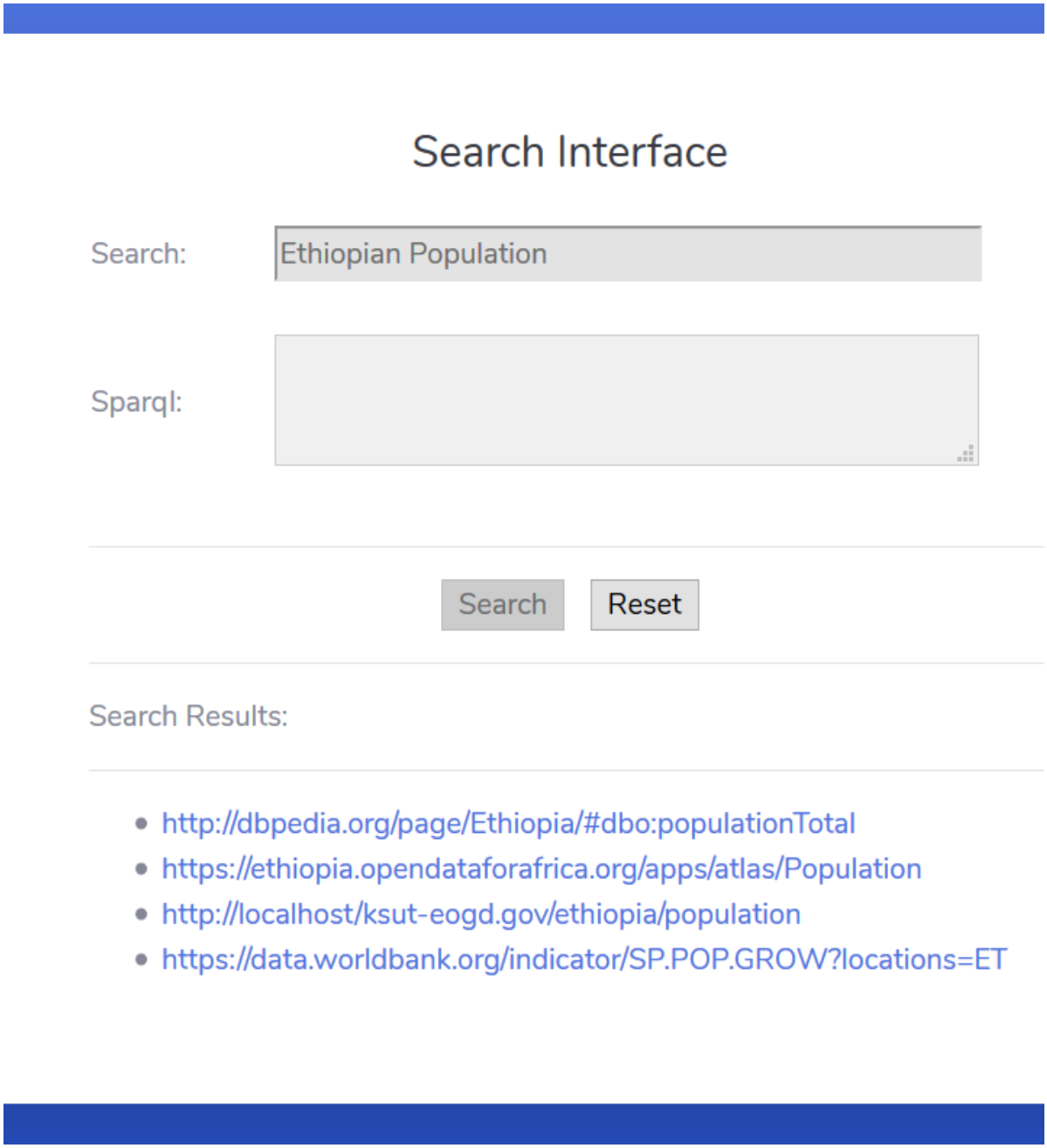

Figure 5.3 OG Linked Data Search Result

Figure 5.3, shows the result for a Search done via Searching in Words. The way the result displayed is similar to the search done via SPARQL Query.

5.6 Evaluation

In order to evaluate the system, sample data is taken for testing. Data with different nature and domain were taken for testing. When searching data using just search key or SPARQL, the result is different. Searching in SPARQL generates more data and the accuracy is much better. When searching using Search Key, the proposed work is forced to generate a SPARQL query from the text search key and/or just process the searching from LOD, that is an additional effort for this work. In processing the Source data, the proposed work performs the transformation to Linked Data in the background to avoid user interference before completion of its life cycle.

5.6.1 Performance Evaluation in Data Processing

The performance of the proposed work is evaluated by measuring the speed of the data processing. Three different types files are taken as sample data and processed and the amount of time it took to process the data is presented in Table 5.1.

NB: All these evaluations are done in Personal Computer having a capacity of 8GB RAM, Core i7 Processor, Windows 10 Operation System.

Table 5.2: Performance Evaluation of the Model

Process	Time Taken (in Seconds)	Data Size	Remark
Upload Source Data	1	434 KB	
Domain Mapping	2	434 KB	
Source Data Picking	2.7	434 KB	
CSV to RDF Triple Conversion	0.9	434 KB	
XLSX to RDF Triple Conversion	1.2	420 KB	
JSON to RDF Triple Conversion	0.5	455 KB	
XML to RDF Triple Conversion	0.5	460 KB	
Convert Triples to Turtle/RDF	15	434 KB	
RDF Dumping	4	434 KB	
RDF Auditing	16	434 KB	
RDF Linking to LOD	40	434 KB	Depends on Internet Connection Speed

5.6.2 Data Retrieval Evaluation

NB: All these evaluations are done in Personal Computer having a capacity of 8GB RAM, Core i7 Windows Processor.

In searching data using the OG Linked Data Search Interface of the proposed work, has two faces. One that can easily talk to the Linked Open Data Cloud using SPAQRQL query. And the other one that uses search key which needs further conversion to SPARQL query. In this work, when searching population data of Ethiopia two types of data were retrieved.

- Using SPARQL query for searching is quicker than using Search Key. And the SPARQL query generate 15 results while 5 of them were directly connected to the question “Ethiopian Population” and while the rest are indirectly connected to the question. The quality of the query has significant impact on the result.
- Using Search Key searching, the proposed work, only generated 4 resources from local and global dataset.

NB: All these evaluations are done in Personal Computer having a capacity of 8GB RAM, Core i7 Processor, Windows 10 Operation System.

In summary, this work took small sized data sources to test the functional and non-functional requirements of the proposed work. And it can be said most of the performance issues are issues of Internet Speed, Computer Processing Capacity and not to forget the quality of the SPARQL query the user is using.

Chapter Six: Conclusion and Future Work

6.1 Conclusion

In this thesis, we described the shortcomings of open government data in the case of Ethiopia and tried to design an automatic linker that can make a difference in transforming the process of dissemination of Ethiopian government data to the tax paying citizens and also others who are interested via LODC.

The proposed work, starts from capturing non-RDF data and ends up with linking open governmental data to the Linked Open Data Cloud. The implementation of the proposed work is developed to maintain the general and specific objectives of this work. The proposed work has seven main components (OGD Input, Source Data Analyzer, RDF Convertor, RDF Analyzer, RDF Publisher, Searching Linked Data and Local Linked Data Repository). It has two end user interfaces: the first one enables users to upload Ethiopian open governmental data and the other one enables users to search for open government data from published Open Government Data and Linked Open Data Cloud. The Source Data Analyzer is the component that performs cleaning the source data, mapping data to its domain, and converting the cleaned source data into triples (subject, predicate and object). The RDF Converter (Triplifier) is the component that performs the transformation of triple data into the user friendly RDF Serialization-Turtle and after conversion, it dumps the data into two RDF Dump files (Turtle). The RDF Analyzer is one of the most important components that handles cleaning the RDF dump files from duplication and misplaced URIs and inspection of attached provenance data in order to identify the correct global representation of the data elements in the RDF dump files. The RDF Analyzer, by matching and mapping of URIs, it performs consolidation of data to ready a publishable RDF data. Online Linked data publishing cannot be done in this work at this level since it is the government who is responsible for this task. Local server named Local Linked Data Repository is prepared for testing purposes. The proposed work has been tested with few sample data in different domains and prompted a good result.

Having a technology or a tool that can handle the automatic process of linking Open Government Data to the Linked Open Data Cloud is advantageous but needs the attention of the government body to avail updated data in standard format (in the case of Ethiopia).

6.2 Contribution

The contributions of this thesis work are as follows,

- Identify the status and shortcomings in the availability and presentation of Open Government Data and proposing the way forward.
- Proposed an automatic processing of source open government data. And this can be a stepping stone for future web applications involving Open Government Data.
- Ignoring the fact that this work performs variety of tasks automatically, the main components and subcomponents of this work can be used independently for particular purposes. For instance, this work can be used independently for domain mapping, converting any tabular data to triples (subject, predicate and object), converting both tabular and triple data to RDF, and data cleaning.
- Minimize the effort in the production of Open Government Linked Data.
- Provide a centralized tool that can track all the phases in processing Government data till the stage of publishing on the Linked Open Data Cloud.

6.3 Future Works

Since the standing of Open Government Data in many countries is in initial level, most the areas are not discovered. If this proposed work is implemented, there are future works needs to be addressed. This work can extend it functionalities to:

- Implementation of Cross-Language Automatic Linking of Federal and Regional Datasets to the Linked Open Data Cloud.
- Using web services to capture datasets from every government units starting from the bottom to top.
- Domain based web applications for Open Government Data.

References

[1] Ubaldi, B., "Open Government Data: Towards Empirical Analysis of Open Government Data Initiatives", Organization for Economic Co-operation and Development (OECD) Working Papers on Public Governance, No. 22, OECD Publishing, retrieved from http://dx.doi.org/10.1787/5k46bj4f03s7-en, Last accessed on July 19, 2019.

[2] Nikos Loutas, "How Linked Data is Transforming eGovernment", ISA Programme of the European Commission, 2013.

[3] Tim Davies, Fernando Perini, and José M Alonso, "Researching the Emerging Impacts of Open Data", Open Data in Development Countries (ODDC) Working Papers #1, 2013.

[4] M. Janssen, Y. Charalabidis and A. Zuiderwijk, "Benefits, Adoption Barriers and Myths of Open Data and Open Government", Information Systems Management (ISM), Vol. 29, No.4, 2012, pp. 258-268.

[5] Tim Davies, Open Data Barometer 2013 Global Report, Open Data Barometer (ODB), 2013, retrieved from http://www.opendatabarometer.org, Last accessed on September 27, 2019.

[6] Tim Davies, Raed M. Sharif, and Jose M. Alonso, Open Data Barometer 2014 Global Report, Open Data Barometer (ODB), 2015, retrieved from http://www.opendatabarometer.org, Last accessed on September 27, 2019.

[7] Tim Berners-Lee. Linked Data - Design Issues, retrieved from http://www.w3.org/DesignIssues/LinkedData.html, Last accessed on June 19, 2019.

[8] Georgi Kobilarov, Tom Scott, Yves Raimond, Silver Oliver, Chris Sizemore, Michael Smethurst, Christian Bizer, and Robert Lee, Media Meets Semantic Web – How the BBC uses DBpedia and Linked Data, Springer-Verlag Berlin Heidelberg, pp. 723–737, 2009.

[9] Dove Winer, "Jewish Studies knowledge grid in the Linked Open Data Cloud", The 11th Jerusalem International Conference on the Digitisation of Cultural Heritage, 2014.

[10] Pedro Szekely et al, "Publishing the Data of the Smithsonian American Art Museum to the Linked Data Cloud", The Semantic Web: Semantics and Big Data, Springer Berlin Heidelberg Vol. 10, 2013, pp. 593-607.

[11] John Sheridan and Jeni Tennison, “Linking UK Government Data”, in Proceedings of the WWW2010 Workshop on Linked Data on the Web, Raleigh, USA, April 2010.

[12] Daniel Dietrich, Jonathan Gray, Tim McNamara, Antti Poikola, Rufus Pollock, Julian Tait and Ton Ziljkstra, Open Data Handbook Documentation, retrieved from http://opendatahandbook.org/guide/en/what-is-open-data/, Last accessed on February 3, 2020.

[13] Florian Bauer, Martin Kaltenböck, Linked Open Data: The Essentials, retrieved from http://www. semantic-web.at/LOD-TheEssentials.pdf, Last accessed on February 3, 2020.

[14] Bernadette Hyland, Ghislain Atemezing, and Boris Villazón-Terrazas, Best Practices for Publishing Linked Data, retrieved from https://www.w3.org/TR/ld-bp, Last accessed on February 3, 2020.

[15] Bernadette Hyland and David Wood, The Joy of Data - Cookbook for Publishing Linked Government Data on the Web. URL: http://www.w3.org/2011/gld/wiki/Linked_Data_Cookbook, Last accessed on February 3, 2020.

[16] Michael Hausenblas and Richard Cygankiak. Linked Data Life Cycles, retrieved from http://linked-data-life-cycles.info, Last accessed on February 3, 2020.

[17] Boris Villazón-Terrazas et al, Methodological Guidelines for Publishing Government Linked Data, retrieved from http://link.springer.com/chapter/10.1007/978-1-4614-1767-5_2, Last accessed on February 3, 2020.

[18] Christian Bizer. Richard Cyganiak and Tom Heath, How to Publish Linked Data on the Web, retrieved from http://linkeddata.org/docs/how-to-publish, Last accessed on February 3, 2020.

[19] Tatiana Tarasova, Publishing Linked Data: The Pordata Use Case, https://www.emcl-study.eu/fileadmin/master_theses/thesis_tarasova.pdf, Last accessed on February 3, 2020.

[20] Karin Breitman, Percy Salas, Daniel Saraiva, Vinícius Gama and Marco A. Casanova, Open Government Data in Brazil, retrieved from http://www.dados.gov.br, Last accessed on February 28, 2017.

[21] Aidan Hogan, Antoine Zimmermann, Jürgen Umbrich and Axel Polleres, Stefan Decker Scalable and Distributed Methods for Entity Matching, Consolidation and Disambiguation

over Linked Data Corpora, retrieved from https://www.emse.fr/~zimmermann/Papers/jws-si2012.pdf, Last accessed on February 28, 2019.

[22] Kelli de Faria Cordeiro et al, An approach for managing and semantically enriching the publication of Linked Open Governmental Data, retrieved from https://www.researchgate.net/publication/284686112, Last accessed on February 28, 2019.

[23] Arup Sarkar, Ujjal Marjit and Utpal Biswas, "Linked Data Generation for The University Data from Legacy Database", International Journal of Web & Semantic Technology (IJWesT) Vol.2, No.3, July 2011.

[24] Ana Brandusescu et al, Open Data Barometer 2016 — Global Report, Open Data Barometer (ODB), 2017, retrieved from http://www.opendatabarometer.org, Last accessed on February 28, 2019.

[25] Mike Bergman, Advantages and Myths of RDF, retrieved from https://www.researchgate.net/publication/256476560, Last accessed on February 7, 2019.

Declaration

I, the undersigned, declare that this research is my original work and has not been presented for degree in any other university, and that all sources of materials used for the research have been acknowledged.

Declared by:

Name: Yemaneberhan Lemma Kebede

Signature: _____

Date: _____

Confirmed by advisor:

Name: Dr. Solomon Atnafu

Signature: _____

Date: _____