Mappings Management for Ontology-based Integration

Nikos Anyfantis

Thesis submitted in partial fulfillment of the requirements for the Masters’ of Science degree in Computer Science

University of Crete
School of Sciences and Engineering
Computer Science Department
Voutes University Campus, GR-70013 Heraklion, Crete, Greece

Thesis Advisor: Assistant Prof. Yannis Tzitzikas

---

This work has been performed at the University of Crete, School of Sciences and Engineering, Computer Science Department.

The work has been supported by the Foundation for Research and Technology – Hellas (FORTH), Institute of Computer Science (ICS).
Mappings Management for Ontology-based Integration

Thesis submitted by
Nikos Anyfantis
in partial fulfillment of the requirements for the
Masters’ of Science degree in Computer Science

THESIS APPROVAL

Author: ________________________________
Nikos Anyfantis

Committee approvals: ________________________________
Yannis Tzitzikas
Assistant Professor, Thesis Supervisor

________________________
Martin Doerr
Researcher, Committee Member

________________________
Dimitris Plexousakis
Professor, Committee Member

Departmental approval: ________________________________
Antonis Argyros
Professor, Director of Graduate Studies

Heraklion, February 2016
Mappings Management for Ontology-based Integration

Abstract

The Semantic Web (SW) is an evolving extension of the World Wide Web in which the content can be expressed not only in natural language, but also in formal languages (e.g. RDF/S) that can be read and used by software agents, permitting them to find, share and integrate information more easily. Semantic Web technologies form the foundation for publishing data according to the principles of Linked Data. Unfortunately, most web content is based on technologies that do not meet the principles of the Semantic Web. The majority of the data is available in relational, object-oriented and NoSQL databases. For this reason several systems have been developed for converting and publishing content as linked data. Each system follows its own method, while it is worth noting the R2RML mapping language proposed by W3C as an official language for expressing mappings from relational databases in RDF datasets. In this work we focus on X3ML language; a mapping formalism designed by the Institute of Computer Science of FORTH, which is human readable, generic enough to cover most of the data models used nowadays and more explicit with the URI generating process.

The main objective of this MSc thesis is to provide a complete analysis of mappings expressed in X3ML language. Firstly, we identify the main business use cases, a) Create a Mapping in which the actor wants to generate a mapping from scratch, b) Improve a Mapping in which the actor needs to improve it, check its quality, compare with others and monitor its evolution. Based on these, we defined the system requirements and then designed and implemented 11 basic functionalities. These functionalities refer to the analysis of the schemas involved in a particular mapping, the calculation of coverage metrics, the graphical representation and comparison of mappings, and the monitoring of their evolution. More specifically, the analysis of schemas offers the opportunity of navigating and better understanding the content. The coverage metrics give an overview of the mapped data and allow excluding elements that are considered undesirable for a mapping. For the graphical representation, we introduce functionalities that visualize a mapping, focusing on syntactic elements of the language, and provide a three-dimensional interactive representation. As regards comparisons, several functionalities are offered such as textual comparison of mappings in XML format, representation of differences through graph and comparison of generated instances. Finally, the functionality of tracking the evolution of a mapping helps the user to identify the basic points that have changed over time. All in all, the main novelty of this work is the interactivity of all functionalities and 3D visualization of mappings.
Διαχείριση Κανόνων Αντιστοιχίσεων για Ολοκλήρωση Πληροφοριών με Οντολογίες

Περίληψη

Ο Σημασιολογικός Ιστός (ΣΙ) είναι μια εξελισσόμενη επέκταση του Παγκόσμιου Ιστού στην οποία το περιεχόμενο μπορεί να εκφραστεί όχι μόνο με φυσική γλώσσα αλλά και με τυπικές γλώσσες (π.χ. RDF/S) που επιτρέπουν την παροχή προηγμένων υπηρεσιών αναζήτησης, διαμορφωμάτων και αναλυτικών πληροφοριών. Στις τεχνολογίες αυτές εδράζεται η τάση για δημοσίευση περιεχομένου βάσει των αρχών των Διασυνδεδεμένων Δεδομένων. Δυστυχώς, το μεγαλύτερο μέρος του περιεχομένου που βρίσκεται στο διαδίκτυο βασίζεται σε τεχνολογίες που δεν ανταποκρίνονται στις αρχές του Σημασιολογικού Ιστού. Η πλειοψηφία των δεδομένων που είναι διαθέσιμα είναι σε μορφή σχεσιακών, object-oriented και NoSQL βάσεων δεδομένων. Για το λόγο αυτό έχουν αναπτυχθεί διάφορα εργαλεία και συστήματα τα οποία αναλαμβάνουν την μετατροπή αυτού του περιεχομένου σε συνδεδεμένα δεδομένα. Κάθε σύστημα ακολουθεί μια διαφορετική μέθοδο, καθώς αξίζει να σημειωθεί και η γλώσσα συσχετίσεων R2RML που προτείνεται από το W3C ως επίσημη γλώσσα για την έκφραση αντιστοιχίσεων από σχεσιακές βάσεις δεδομένων σε RDF σύνολα δεδομένων. Στην εργασία αυτή επικεντρωνόμαστε στην X3ML, μια γλώσσα αντιστοιχίσεων που σχεδιάστηκε από το Ινστιτούτο Πληροφορικής του ΙΤΕ, η οποία είναι αναγνώσιμη από τον άνθρωπο, αρκετά γενικής έκφρασης και εκφραστικής ώστε να καλύπτει τα περισσότερα μοντέλα δεδομένων που χρησιμοποιούνται στις μέρες μας και πιο εκφραστική ως προς τη διαδικασία δημιουργίας URI.

Ο κύριος στόχος αυτής της μεταπτυχιακής εργασίας είναι η παροχή λειτουργιών για τη διαχείριση αντιστοιχίσεων εκφρασμένες στη γλώσσα X3ML. Αρχικά εντοπίσαμε τις κύριες περιπτώσεις (business use cases), συγκεκριμένα τις α) Δημιουργία μιας αντιστοίχισης, κατά την οποία ο χρήστης θέλει να δημιουργήσει μια αντιστοίχιση από την αρχή, και β) Βελτίωση μιας αντιστοίχισης κατά την οποία ο χρήστης επιθυμεί να βελτιώσει μια υπάρχουσα αντιστοίχιση, να ελέγξει την ποιότητά της, να τη συγκρίνει με άλλες και να παρακολουθεί την εξέλιξή της. Βάσει αυτών ορίσαμε τις απαιτήσεις του συστήματος και εν συνεχεία σχεδιάσαμε και υλοποιήσαμε 11 βασικές λειτουργίες. Οι λειτουργίες αυτές αφορούν στην ανάλυση των σχημάτων που εμπλέκονται σε μία αντιστοίχιση, στην αναδιάρθρωση πεσοστών κάλυψης, στη γραφική απεικόνιση και σύγκριση αντιστοιχίσεων, και στην παρακολούθηση της εξέλιξης τους. Συγκεκριμένα, η ανάλυση σχημάτων προσφέρει την δυνατότητα να περιηγηθούμε και να καταλάβουμε καλύτερα το περιεχόμενο τους. Τα ποσοστά κάλυψης παρέχονται μια συνολική εικόνα των στοιχείων που έχουν χρησιμοποιηθεί ενώ υπάρχει και η δυνατότητα εξαίρεσης των στοιχείων που δεν ισχύουν αντιστοίχιση. Λειτουργίες όπως κειμενική σύγκριση αντίστοιχων αναπαραστάσεων, υπολογισμός επιπλέον στίγματος, αναλυτικές λειτουργίες όπως κειμενική σύγκριση αντίστοιχων XML αναπαραστάσεων, οπτικοποίηση της διαφοράς με γράφο και σύγκριση παραγόμενων στιγμιότυπων είναι εφικτές. Τέλος,
Η λειτουργία της παρακολούθησης της εξέλιξης βοηθάει το χρήστη να εντοπίσει τα σημεία που έχουν αλλάξει κατά την πάροδο του χρόνου. Συνοψίζοντας, η καινοτομία έγκειται στον διαδραστικό χαρακτήρα όλων των λειτουργιών και στην τρισδιάστατη απεικόνιση των αντιστοιχίσεων, καθιστώντας την παρούσα εργασία την πρώτη μελέτη εκτεταμένης διαχείρισης αντιστοιχίσεων στο συγκεκριμένο τομέα.
Ευχαριστίες

Στο σημείο αυτό θα ήθελα να ευχαριστήσω τον επόπτη καθηγητή μου κ. Γιάννη Τζίτζικα, τόσο για την εμπιστοσύνη που μου έδειξε στο πρόσωπό μου, ανακηρύσσοντάς μου τη συγκεκριμένη διπλωματική εργασία, όσο και για την ορθή καθοδήγησή και ουσιαστική συμβολή του στην ολοκλήρωση της παρούσας εργασίας. Επιπλέον, θα ήθελα να εκφράσω τις ευχαριστίες μου στον κ. Martin Doerr και στον κ. Δημήτρη Πλεξουσάκη για την προθυμία τους να συμμετέχουν στην τριμελή επιτροπή, καθώς και στην κ. Μαρία Θεοδωρίδου για την άψογη συνεργασία και την πολύτιμη βοήθειά της.

Ακόμα θα ήθελα να ευχαριστήσω το Ινστιτούτο Πληροφορικής του Ιδρύματος Τεχνολογίας και Έρευνας για την υποτροφία που μου προσέφερε καθ’ όλη τη διάρκεια της μεταπτυχιακής μου εργασίας, καθώς και για την πολύτιμη υποστήριξη σε υλικοτεχνική υποδομή και τεχνογνωσία.

Πολλές ευχαριστίες θα ήθελα να εκφράσω επίσης στην Αργυρώ, η οποία με αμέριστη υπομονή και κατανόηση ήταν εκεί να με υποστηρίζει και να με ενθαρρύνει για το καλύτερο.

Τέλος, θα ήθελα να ευχαριστήσω ιδιαίτερα τους γονείς μου Σπύρο και Ευγενία, και την αδερφή μου Ιωάννα, για την συμπαράστασή και την εμπιστοσύνη που μου έδειξαν όλα αυτά τα χρόνια των σπουδών μου, αλλά και για την αγάπη τους σε κάθε βήμα την ζωής μου. Σας ευχαριστώ πολύ για όλα!
στους γονείς μου
4 Proposed Approach

4.1 Business Use Cases
4.1.1 Create a Mapping
4.1.2 Improve a Mapping

4.2 Actors

4.3 Requirements Analysis

4.4 System Use Cases

4.5 Proposed Functionality
4.5.1 Preliminaries of Mappings
4.5.2 Explore Schemas
4.5.2.1 Explore Source Schema
4.5.2.2 Explore Target Schema
4.5.3 Percentage Coverage of Schemas
4.5.3.1 "Direct" and "Ancestors - descendants" Metrics
4.5.3.2 Percentage Coverage of Source Schema
4.5.3.3 Percentage Coverage of Target Schema
4.5.4 Mapping Representations
4.5.4.1 X3ML-centered Representation
4.5.4.2 Mapping-centered Representation
4.5.5 Comparison of Mappings
4.5.5.1 Textual Comparison
4.5.5.2 Graphical Comparison
4.5.5.3 Instance Comparison
4.5.6 More Functionalities
4.5.6.1 Instance Representation
4.5.6.2 Tracking Changes Over Time

5 Implementation

5.1 Description

5.2 Technical Information and Architecture
5.2.1 Server-side Environment
5.2.2 Client-side Environment

5.3 Maze User's Manual
5.3.1 Single Mapping Analysis
5.3.2 Comparison of Mappings

6 Conclusions and Future Work

Appendices

Appendix A
A X3ML Mapping

Bibliography
List of Figures

2.1 Basic Layers of Semantic Web ........................................... 8
2.2 R2RML Example ............................................................... 13
2.3 The OpenLink Virtuoso Universal Server architecture ............... 15

3.1 Data Provisioning Flow ..................................................... 21
3.2 Structure of X3ML ............................................................ 24
3.3 Coins - An Archaeological Source ....................................... 25
3.4 Coins Database ............................................................... 25
3.5 Mappings of coins ............................................................ 26
3.6 The main components of X3ML Engine .................................. 28
3.7 3M Editor - Info page ......................................................... 29
3.8 3M Editor - Mapping table .................................................. 30

4.1 Source Schema Transformation ............................................ 44
4.2 Tree structure ................................................................. 46
4.3 X3ML-centered Representation .......................................... 48
4.4 Mapping-centered Representation ....................................... 50
4.5 Comparison of Mappings based on X3ML ............................... 51
4.6 Textual Comparison .......................................................... 53
4.7 Definition-Comparison of Mappings .................................... 54

5.1 The Maze Global Architecture ............................................ 58
5.2 Server-side Environment .................................................... 60
5.3 Client-side Environment .................................................... 61
5.4 Configuration File - Client .................................................. 62
5.5 Configuration File - Server ................................................ 63
5.6 Single Mapping Analysis Screen-shot ................................... 68
5.7 Source Schema details ....................................................... 68
5.8 Sample data ................................................................. 69
5.9 Define undesired data source schema ................................... 69
5.10 Source Schema ER .......................................................... 69
5.11 Source Schema Tree ......................................................... 69
5.12 Selected Target Schema .................................................... 70
5.13 Target Schema Statistics .................................................. 70
# List of Tables

2.1 R2RML Example .......................................................... 13  
2.2 Comparative table summarizing the main characteristics of the existing mapping approaches. .............................................. 17  
3.1 Sample of X3ML .......................................................... 23  
3.2 X3ML mapping of Coins .................................................. 26  
4.1 Source schema Analysis .................................................... 38  
4.2 Target Schema Analysis ................................................... 38  
4.3 Ignore Private Data or data for internal usage .......................... 38  
4.4 Percentage coverage of source schema ................................. 39  
4.5 Percentage coverage of target schema .................................. 39  
4.6 Mapping Overview of X3ML ............................................. 39  
4.7 Mapping assertions between mapping and source schema .......... 40  
4.8 Mapping assertions between mapping and target schema .......... 40  
4.9 View mapping assertions involving a given concept .................. 40  
4.10 Compare two mappings ................................................. 41  
4.11 Tracking changes over time ............................................. 41  
4.12 Analysis of generated instances ....................................... 41
## List of Algorithms

<table>
<thead>
<tr>
<th></th>
<th>Algorithm</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Mapping rules generation procedure</td>
<td>50</td>
</tr>
<tr>
<td>2</td>
<td>Order X3ML elements for Textual Comparison</td>
<td>52</td>
</tr>
<tr>
<td>3</td>
<td>Main procedure for Graphical Comparison</td>
<td>54</td>
</tr>
</tbody>
</table>
Chapter 1

Introduction

1.1 Motivation

The World Wide Web (www) has radically changed the world by connecting users from different places and proposing a new era of sharing knowledge and information. However, most users could only navigate to websites and not contribute to their content. The information was static and could be updated only by experts. The second wave of the Web (Web 2.0) provided the user with the capability of interaction and collaboration among others, creating dynamic web content with user-centered information. This was closer to the original vision of Tim Berners-Lee of a “collaborative medium, a place where we could all meet, read and write” [37].

Nowadays, it is easy to create web pages and publish them to the web. Now anyone, anywhere can access them using a Web Browser and benefit from their content. Nevertheless, this information can be exploited only by people rather than machines. Users can understand the content of websites and navigate through links. In contrast with humans, machines cannot process data and use the available information with a meaningful purpose.

However, it is very difficult to develop software that collects information from the web to perform a specific function, for example organize a business trip according to specific preferences. This problem is a major challenge in research for more than a decade. If the available information was available in a form understandable by machines then tools could take advantage of it and use it. Regarding this perspective, data from different sources could be connected together and it would be possible to create new knowledge derived from a combination heterogeneous data sources. This is the main purpose and vision of the Semantic Web.

The main goal of the Semantic Web is to publish the available data on the Internet. Semantic technologies allow users to focus on application development rather than on data processing. The first attempt to do this was to define with semantic annotations the existing web pages. The use of XML technology and the separation of content from the configuration were techniques which expected to
fulfill this goal. However, the forces that shape the evolution of the web help the
dynamic representation based on users rather than the semantic content. There-
fore, today it is difficult to add semantic notes in applications and services that
were static web pages.

According to the current state, the most information is not available in se-
monic form. The most frequent forms are relational, object-oriented and NoSQL
databases. So instead of trying to retrieve information from web content, it is eas-
ier and more effective to exploit directly databases. One of the current challenges
of Linked Data [17] is the effective and correct conversion of existing data. These
data can be published in a particular form that is readable and can be processed
by any software. Also, heterogeneous data from different sources can be intercon-
ected together to create a common web content. This achievement will give us
great opportunities in order to access and process the available resources.

One of the biggest challenges is the interoperability of various existing data
sources. A typical system based on relational databases can export and import
data in various formats (e.g. XML, CSV). Also, the current databases are defined
based on standards, such as SQL (Structured Query Language). Unfortunately, all
these technologies are not enough to meet the objective of the Semantic Web.

Below, we describe the main disadvantages of existing resources regarding the
conversion process. We focus on relational databases which is the most common
case of stored data.

1. The structure of relational databases is not flexible. The current software
that processes a database is designed and implemented for a specific database
schema and needs to be renewed in order to reflect changes in that schema.
One application in order to process relational data independently of its schema
would be very complicated to implement.

2. The semantics of the data in a relational database is not explicit. The development
of applications that process various databases with different semantics
and correlations with different data is quite difficult. For humans, it is easy
to understand that two tables with names "teacher" and "lecturer" contain
similar information, but this would be extremely difficult to be achieved by
any software. There are many approaches which try to implement machine
learning techniques and algorithms, but they demand programming effort,
time and results are not always reliable.

3. A typical relational database may contain private information or data for
internal use only mixed with data that should be published. Therefore, software
must be developed in order to use the private data for internal use only. Also,
publishable data may need to be preprocessed to normalize its structure.

Taking into account the main differences between Semantic Web and Object-
Oriented (O-O) structures, we would say that the biggest advantage of RDF/OWL
structures is the properties between resources. More specifically, in O-O languages
1.2. GOALS AND APPROACH

The properties are defined locally to a class (and its subclasses through inheritance) in contrast with RDF where properties are stand-alone entities that can exist without specific classes. Furthermore, instances can have values only for the attached properties in O-O structures; while in RDF, instances can have arbitrary values for any property (range and domain constraints can be used for type checking and type inference).

In order to allow data transformation and aggregation, it is required to produce mappings to correlate equivalent concepts or relationships from the source schemata to the aggregation schema, i.e. the target schema, in a way that facts described in terms of the source schema can automatically be translated into descriptions in terms of the target schema (also known as the "enterprise model" [18]). This is the mapping definition process and the output of this task is the mapping, for instance a collection of mapping rules. Until now, there is a big variety of systems that has been developed in order to convert existing data to linked data (see § 2.2 and § 2.3). These systems have implemented several manual or automatic techniques in order to achieve an effective conversion. However, these approaches have a common target: create mappings that link source to target schemas. In order to express these mappings, there is a big variety of implemented mapping languages which is used to describe them. Also, these approaches can be classified into three general categories: Direct Mapping, Augmented Direct Mapping and Domain Semantics-Driven Mapping.

Unfortunately, the majority of the current mapping approaches do not take into consideration the analysis of the mappings. They target to convert source to target schemas creating mapping rules, but there are several considerations and challenges during the mapping process. The mapping rules express the way that the source data will be translated to another representation. So instead of trying to create simple rules for shortening the mapping process, it is required to study their behavior and analyze the final result. Thus, the main purpose of this work is the complete analysis of mappings and their dependencies such as source and target schema. Overall, we distinguish two main business use cases: a) create a mapping from scratch, b) improve an existing mapping, check its quality, compliance with others, evolution etc. The next section presents a more extend description of our goals and approach.

1.2 Goals and Approach

The purpose of this work is the management and the analysis of mappings to convert properly a source schema to a target schema through mappings. Firstly, it would be useful to understand the forces which led us to this study. Then, we will focus on real problems describing business use cases and providing an overview of our approach.

Studying the different approaches, we have identified three common goals of these systems: (a) access to data from the deep web, (b) interconnection of data
with other relevant parts, (c) integration of multiple heterogeneous data sources.

The web contains information which is at a lower level, in most cases it is difficult for search engines to consume and utilize this data. We refer to data sources that are hidden in unstructured documents (photographs, scans), semi-structured documents (CSV files, PDF files, etc.), or structured data (relational databases, databases, XML, NoSQL databases, LDAP directories etc.) where standard web tools cannot consume them, but they are accessible only through questions. For example, a few years ago, the 70% of the websites were supported by RDBs, which contained 500 times more data than the available resources [10]. Converting all this vast amount of information and make it available in a form understandable by machines, we expect to create opportunities for innovative applications and services. Regarding this achievement, the representation in RDF/S constitutes a strong data format specification for this purpose. Also, ontologies provide several advantages such as the possibility to use automatic processes to reason on knowledge thanks to its formal and explicit nature and the availability of rich semantics to represent knowledge (richer than databases due to its sophisticated semantic concepts [38]). However, in order to ensure the viability of existing applications, data should remain in the form of relational databases; hence the great need for mapping techniques can access source schemas and convert them into linked data.

The interconnection of data with other relevant resources increases dramatically their value. Tim BernersLee formulated the main principle of linked data [12] proposing practices about publishing and sharing information and knowledge on the Semantic Web using URIs and RDF. Guided by these recommendations, the society of the Linked Open Data aims to a new trend extending the existing web, publishing several sets of open data in RDF format and creating connections between them. In that regard, the success of this vision depends heavily on the accessibility of data and the availability of mapping tools to assist in conversion and publication of existing relational data to RDF/S.

The integration of heterogeneous data is a major challenge in many areas (i.e. earth sciences [50], biodiversity [47, 48], clinical studies and e-Health [34]). For example, the correlation and search in heterogeneous data sources help the Neuroscience in order to describe different forms and scales of proteins, genes, cells and behavior. The most important step towards the integration of heterogeneous data sources is to define the semantics. The existing source schemas usually have no or very poor semantics. In most relational schemas, it is very difficult to distinguish relationships between entities. Although in cases of structured formats, the direct relations are distinct and the indirect links are impossible to be obtained. Also, many of the source schemas serve better the performance than the semantics requirements. As a result, addressing the challenges of data integration, several techniques have to collect and distinguish the semantics in order to be an understandable format for machines. Using RDF as a form of representation of relational data seems to be a strong and promising way in order to face all these problems and challenges.

Taking into account the mapping process, most of the systems have as main
objective to convert existing databases into linked data. This can be achieved by creating rules which express mappings between a source and a target schema. However, many of these tools create rules which require: (a) domain experts’ knowledge in order to define mapping since the user shall write code in a specific programming language or define rules in a specific mapping description language manually, (b) time consuming processes in order to complete effective mapping rules, (c) manual way in order to analyze the total progress and the final result.

Thus, in most cases it is not enough to create mapping rules. There is a need for a more expressive and effective mapping analysis in order to evaluate and compare several approaches.

1. **Source and target schema analysis.** The analysis of schemas consists one of the most important requirements in order to achieve an effective mapping. Thus, we have to understand the content and be able to identify correct transformations between these schemas.

2. **An important part is the percentage coverages of mappings.** In many cases several approaches convert data to an ontology schema. Therefore we must know the coverage of the schemas regarding a specific mapping.

3. **In most cases, the mapping rules have an incomprehensible structure.** The tools use specific forms to express mappings which are difficult to navigate and understand the rule. So there is a need for visualization and description through virtual rules which are more understandable to users who are not experts.

4. **Different ways of representation.** There are cases where a specific representation is not enough. Thus, we need to apply different ways of representation which are easier to understand the mappings and facilitate the navigation of the mapping.

5. **Need for mapping comparison.** The conversion of a source schema into linked data can be done in different ways depending on the purposes of use, the actor, etc. So, we need to be able to compare two mapping together to observe common and different points of interest.

Considering all the above-mentioned cases, we define the main purposes of our approach providing the business use cases regarding the management of mappings. In brief, these are:

- **Create a Mapping:** it describes the process in which actors want to create a mapping from scratch.

- **Improve a Mapping:** cases which actors have already implemented a mapping and try to improve it, check its quality, compare with others and monitor its evolution.
CHAPTER 1. INTRODUCTION

As regards these business use cases, we define system requirements and propose system use cases analyzing more extensively the users’ capabilities during the mapping process. Then we design and implement eleven functionalities referring to analysis of source and target schema, percentage coverage metrics, mapping representations, comparison of mappings and tracking changes over time.

More specifically, the analysis of schemas offers the opportunity to explore and understand better their content providing multiple capabilities. The percentage coverages of schemas include two different metric methods that allow us to have an overview of the used data. Also, the user may exclude data which are considered undesirable for a mapping and calculate new coverage metrics. The graphical representation techniques are divided into two functionalities. The first represents a graph based on language X3ML focusing on syntactic elements of language. The second functionality transforms the schemas in a three-dimensional space creating relationships and illustrating the correlations regarding the selected mapping. As regards the comparison of mappings, we implemented comparisons about the syntax of X3ML language, the graphical representation of correlations and comparison of the generated instances. Taking into consideration the scope and application parameters of mapping, instance representation offers exploring abilities to users concerning derived result. Finally, the functionality of tracking the evolution of mapping helps the user understand the basic points which have changed over time.

1.3 Thesis Overview

The body of this work is organized in six main chapters. The next chapter provides the relevant background and describes the main prerequisites. The following chapter describes the mapping context, the X3ML language and relative tools. Then, the next chapters provide the proposed approach and the implementation. The remainder of this section introduces each of these chapters.

Chapter 2 introduces the Semantic Web and its most important principles. It describes several mapping approaches and their general classification, while the final section provides a list of related tools and their approach.

Chapter 3 provides an introduction of the mapping context by referring the Synergy Reference Model. The next section presents the structure of the X3ML language and its components. Finally, we provide the repository and other tools, such as the X3ML Engine and the 3M Editor.

Chapter 4 presents the proposed approach by describing business use cases of mapping process, involved actors, requirements and system use cases. Also, it draws the proposed approach describing the purposes and their effectiveness regarding mapping management.

Chapter 5 describes in detail the implementation part providing the structure of our system. Also, it presents in detail the user manual of implemented application.

The last chapter draws some conclusions about this work and the future directions of our research.
Chapter 2

Background and Related Work

2.1 Semantic Web

The Semantic Web (Web 3.0) [15, 7] is an extension of the current Web that offers structure to the meaningful content of Web pages. The strategy behind this vision is that the published information will contain metadata, which will be public; they can be "understood" by machines, which will contribute to better usage and processing. The Semantic Web is based on technologies that already exist (URI and XML), but it also combines new technologies (RDF/S\textsuperscript{1}, OWL\textsuperscript{2}, etc.), which were developed in order to meet the requirements of its principle. The new trend of the web intends to be a great base where data from different fields will be linked together.

Some of the areas that will have the biggest impact are the health, education and business. There are already many efforts by companies, researchers and non-profit organizations to produce ontologies, especially for the above fields in order to have common languages and more combined data for better results.

As regards the field of health, there is an effort to create a unified medical language and terminology services to help medical staff and to guide consumers to trustworthy health information depending their symptoms. In education, the Semantic Web will significantly contribute to learning and mainly the searching process for information, by organizing the results and creating special learning programs. In the business field, we will have a proper structure of each company, better experiences for users in online shopping and better coordination between different companies. In everyday life there will be effects of Web 3.0 in social networks and virtual communities. There will be applications that will give more and more trusted information and they will greatly facilitate the online activities.

The main goal of the Semantic Web is to evolve the current web. This permits users to find, share, and combine information more easily. People are able to use the opportunities of the web, such as to find translations for other languages, to

\textsuperscript{1}https://www.w3.org/TR/rdf-schema/
\textsuperscript{2}https://www.w3.org/2001/sw/wiki/OWL
bind a book from the library, searching the lowest price for a DVD, etc. Nevertheless the machines cannot use these features without the involvement of users because websites are currently designed to be readable by people, not by machines. The semantic web is a form of information which can be used by machines. Thus, computers undertake to do processes including search, union and information processing which is available on the Internet.

The Semantic Web provides a common framework that allows data to be shared and reused across applications. Its components are deployed in the layers of Web technologies and specifications as represented in Figure 2.1.

![Figure 2.1: Basic Layers of Semantic Web](image)

There are five main components of the Semantic Web:

- **URI** - Uniform Resource Identifier: is a format for web identifiers that is widely used on the World Wide Web. The Semantic Web uses URIs to represent most kinds of data.

- **RDF** - Resource Description Framework [35, 32] is used by Semantic Web to describe data uniformly, allowing it to be shared. It is a general metadata format used to represent information about Internet resources. It extends the expressive capability of Web augmenting human-readable web pages with machine-processable information.

- **RDF Schema** is a language used by the Semantic Web to describe the data properties used in RDF. It provides mechanisms for describing resources and relationships between these resources. The RDFS vocabulary descriptions are also RDF [5].

- **Ontologies** are used to represent the structure of knowledge domain [29]. The Semantic Web uses OWL, Web Ontology Language. Applications need that language in order to process data rather than just display it. OWL adds the possibility of reasoning to data by identifying and describing relationships between data items. Ontologies are defined independently from the actual
2.2. MAPPING APPROACHES

data and reflect a common understanding of the semantics of the domain. It provides definitions of classes, relations, functions, constraints and other objects.

- Logic - Inference is useful to derive new data from data. A common example is the property transitivity. If an element has a type A and the type A is a subtype of type B then the element has also the type B.

2.2 Mapping Approaches

This section describes the general approaches and methods of existing mapping systems. It analyzes the mapping processes and distinguishes them in three main categories. Finally, we present the R2RML mapping language proposed by W3C as an official language for expressing mapping from relational database in RDF datasets.

2.2.1 General Approaches

Considering different approaches, it is important to study the description of mappings between relational databases and representation in RDF format [43, 40, 4, 42]. More generally, the description is based on the fact that the final mappings should either express an ontology resulting from the relational database schema, or fulfill descriptions defined by existing ontology. The second case is more complicated process regarding the mappings.

In this specific field of science the term Direct mapping is considered as a synonym of automated transformation of data and the term Manual mapping is synonym of transformations regarding a target schema. Nevertheless there is confusion about these terms because the direct mapping is created automatically but then customized manually. Respectively the manual mapping in the early stage creates a basis on which we can build the rest transformation rule. In addition some approaches follow augment direct mapping to construct a rule. Overall, the different approaches can be classified into three general categories: Direct Mapping, Augmented Direct Mapping and Domain Semantics-Driven Mapping.

2.2.1.1 Direct Mapping

The first approach is the Direct Mapping [8, 14] between relational database and Linked Data. In this approach, the semantics of the data are generated by a predefined way translating the relational schema to ontology. More specifically, it provides automatic creation of URIs following these simple rules, as formulated by Tim Berners-Lee [13]:

- table-to-class: a table is translated into an ontological class identified by a URI whose construction follows the pattern “namespace/database/table”;
• **column-to-property**: each column of a table is translated into an ontological property whose URI follows the pattern “namespace/database/table/column”;

• **row-to-resource**: each row of a table is translated into a resource whose type is the class representing the table and whose URI is formed by using the table’s primary key: “namespace/database/table/primaryKey” or “namespace/database/table/primaryKey”;

• **cell-to-literal-value**: each cell with a literal value is translated into the value of a data property;

• **cell-to-resource-URI**: each cell with a foreign key constraint is translated into a URI which is the value of an object property.

Applying this set of rules a vocabulary is created automatically to RDFS or OWL form, reflecting the structure of the relational schema. In order to avoid unnecessary or sensitive data like passwords, most approaches automatically generate a first mapping that can be adapted to manual mode to some extent. Some specific cases, such as columns primary keys, and many tables without primary key not treated in a straightforward way.

The Direct Mapping method according to the relational schema is typically applied when there is not a specific ontology that describes properly the contents of the database, or when we want the whole transformation process to be very fast. Although the result of this method is readable by machines, there is a high possibility of error on the semantic interoperability of data.

### 2.2.1.2 Augmented Direct Mapping

Another mapping approach is the Augmented Direct Mapping. This approach targets to improve the quality of the previous approach by the automatic detection of common database design patterns that can contain domain semantics. For instance, many-to-many relations are suggested by tables in which all non-primary key columns are foreign keys to other tables; nullable/not nullable columns can be converted into OWL cardinality constraints; implicit subclass relationships are often suggested by a primary key used as a foreign key [21]. In the latter case though, the literature argues that, in the context of databases not in the third normal form, this pattern may reveal a vertical partitioning (splitting of a table into several smaller tables for performance concerns) rather than a subsumption relationship [44, 46].

Some semi-automatic approaches propose an iterative process in which a domain expert validates or dismisses proposed mappings. Classes may be refined based on the detection of lexical clues in the column names or data redundancy suggesting categorization patterns [19]. Additionally, some research questions remain open with regards to the possibility of translating relational database triggers into additional knowledge in the form of semantic rules.
2.2. MAPPING APPROACHES

2.2.1.3 Domain Semantics-Driven Mapping

The direct mapping approach is hardly sufficient in real world applications, in which the data semantics lies outside the RDB schema, in domain-specific rules encoded in the application that exploits the database. This is outlined in a short but enlightening feedback from the Ordnance Survey of Great Britain: "databases are rarely good descriptions of a domain, being the result of both performance optimization processes and contingent maintenance history. And, in any case, the schema itself will not support a full description of the domain, other relevant relationships often being buried in code or in the encoding of various attributes" [25].

To overcome these limitations, the domain Semantics-driven mapping approach applies when the relational database must be translated using classes and properties of existing ontologies. The database and the ontology may have been designed separately, and their similarity level may be low. A typical use case is the alignment of a legacy database with an existing ontology describing or referring to the same domain of interest.

Domain semantics-driven systems rely on mapping description languages to allow the description of expressive mappings, able to bridge the conceptual gap. In any case, those languages generally implement two different strategies:

- The mapping description essentially relies on SQL queries to present the data as RDF triples. The expressiveness of mappings is therefore constrained by that of the SQL flavour used; in other words, complex cases that would require a richer expressiveness cannot be addressed, unless they are supported by extensions of a specific RDBMS. On the other hand, the rewriting of a query on the target RDF data (generally a SPARQL query [30]) into SQL is almost straightforward, and the query execution can benefit from the native database optimizer. Besides, the great popularity of SQL facilitates the adoption of the mapping language by data providers who do not need to learn a new mapping language.

- The mapping description uses a specific dedicated language. In this approach, the query rewriting process is more complex as it does not rely on SQL queries written by the person who wrote the mapping. The mapping is not constrained by the expressiveness of SQL. As a result it can be extended in order to meet specific complex needs such as keyword search, regular expression matching, natural language processing, data mining, etc. Nevertheless, it must be underlined that most existing projects hardly reach the expressiveness of SQL (for instance, aggregation and grouping are not always possible although they are natively supported by SQL).

Some mapping languages such as R2RML and D2RQ [22] use both strategies simultaneously: they are able to complement SQL snippets with specific mapping descriptors.
2.2.2 R2RML Approach

2.2.2.1 Description and Specifications

R2RML [23, 20] is a generic language to describe a set of mappings that translate data from a relational database into RDF. It is the result of preliminary works held by the W3C. Also, R2RML is a language for expressing customized mappings from relational databases to RDF datasets. Such mappings provide the ability to view existing relational data in the RDF data model, expressed in a structure and target vocabulary of the mapping author's choice. R2RML mappings are themselves RDF graphs and written down in Turtle syntax. R2RML enables different types of mapping implementations. Processors could, for example, offer a virtual SPARQL endpoint [6] over the mapped relational data, or generate RDF dumps, or offer a Linked Data interface. Below, we report some of the main advantages and drawbacks of R2RML.

R2RML provides a very expressive mapping language. Together with SQL, it is possible to map the database in any way one wants. R2RML relies quite heavily on SQL. This can be both an advantage and a disadvantage. It is an advantage, because it allows R2RML to reuse the power of SQL to produce mappings. This will push some of the mapping work from the R2RML processor to the DBMS. However, using SQL can hide much of the mapping semantics inside the SQL query, where it is not as easily accessible.

R2RML does not support write access to the generated RDF views. However, this was explicitly listed as out of scope by the RDB2RDF Working Group. Adding support for write access to the mapping language creates several problems. The main problem is the view update problem, which makes it hard to update the database based on a view. It would therefore be necessary to add restrictions to the mapping language to get general write support. This is impractical for R2RML.

2.2.2.2 R2RML by Example

In order to understand the R2RML language, we will create a mapping from scratch. This example regards the conversion of coins. First step is to understand the tables of the source schema. We suppose that there is a COIN table with three columns: ID, COINID and WEIGHT (see Figure 2.2).

A mapping rule in this language is referred to as a TriplesMap. Example 1 illustrates an R2RML mapping rule in Turtle syntax (see Table 2.1). The R2RML mapping in this example has three components: The predicate rr : sourceTable points to a structure selecting a table, view, or SQL query from the relational database that is mapped into RDF; in this example, a custom SQL query is used to retrieve the columns ID, COINID and WEIGHT from the table COIN. The predicate rr : subjectMap points to a structure defining how rows from the logical table query results are transformed into possibly typed, subjects; in the example, we create subjects by instantiating the specified rr : template with the values from the COINID column, and specify mo : coin as the type of these subjects. The
2.3. MAPPING TOOLS

2.3.1 D2R Server

D2R Server is a project developed at the Freie University of Berlin. We will see that a current standard adopted by the W3C is based on a mapping language of D2R Server. The goal of D2R server is publishing of relational databases content on the semantic web, more precisely in RDF form. The base of functioning of D2R Server relies on a mapping mechanism to translate relational data (SQL, for instance) into resources and properties. Moreover, D2R server provides functionalities to Web agents to query RDF data with SPARQL and retrieve XHTML and RDF representation of those data [16].

D2R Server uses the language D2RQ Mapping Language, addressing the specific needs of the application. The role of a D2RQ Mapping is to specify how RDF triples have to be created from a relational data source. The main component of a D2RQ mapping, is the \texttt{d2rq-ClassMap}. It represents the set of relational elements that will be translated to a certain class of resources. It also defines how the instances of those classes are identified. A \texttt{d2rq:-classMap} is linked to a \texttt{d2rq:-Database}.
element. The D2R Mapping Language provides four mechanisms for identifying instances of classes.

1. URI patterns and relative URI patterns. A URI pattern is a generic string which is instantiated with values contained in the database.

2. URI columns. In some cases, database columns already contain URIs (for example, a column containing webpage links or email addresses). Values can therefore be used directly as resources in the RDF graph.

3. Blank Nodes. Those qualifiers can be used to describe resources that cannot be named and that has certain properties. They are created from one or more columns and a distinct blank node is generated for each distinct set of values.

4. URI expressions. One can generate URIs with a valid SQL expression. The only constraint is that the result of this expression must produce a valid URI.

In order to have functional mapping, a data source must be given. The element \texttt{d2rq:-Database} gathers the several necessary information, such as the type of database, connection, driver, username, password, location, etc. It is important to note that the connection to the data source takes part in the core of the mapping language, while in other mapping languages, the data source becomes abstract and the connection not directly implemented.

2.3.2 Openlink Virtuoso Universal Server

OpenLink Virtuoso Universal Server is a cross-platform universal server. It acts as a virtual database engine managing a large number of databases types: DB2, SQL Server, Oracle, Sybase, etc. It is completely transparent to the user, who can easily access data distributed among a heterogeneous set of server. The roles and possibilities of OpenLink Virtuoso Universal Server are summarized on the diagram below (see Figure 2.3).

A particularity of Virtuoso is that it does not store triples, but quads, which use a graph field indicating the belonging of a triple to an application or a resource. Quads are formed thanks to quad map patterns. Quad map patterns are the rules managing the translation process. They define the transformation of a column (or a set of columns) into triples. The quad map pattern is composed of four parts, specifying how to derive triple field from the SQL data [3].

A virtuoso mapping is divided in three main parts. First, the \texttt{sparql} tag has to be given and can be followed by a series of prefixes that will be used later in the document. In our case, we use \texttt{rdf} (for the \texttt{rdf:type} predicate), \texttt{foaf} (for researchers names) and our own namespace \texttt{ulb}, addressing the description of all the other concepts. Second, the different IRI classes have to be created. Our example requires to create two IRI classes, namely for laboratories and researchers.
The IRI creation is made by replacing ad-hoc parts in a formatted string by values in columns. The concerned columns are specified in parenthesis alongside the string. Third and final, a quad storage can be created. It will contain the different quads generated by quad patterns (two in our case). A quad map storage is very useful when dealing with a huge amount of quad map patterns, so they can be manipulated as sets. A quad storage is related to relational tables, which are the data sources for all the quad map patterns defined in this very quad storage.

2.3.3 Triplify

In order to publish Linked Data from relational sources, a light-weight application called Triplify has been created [9, 31]. The main purpose of triplify is to promote Semantic Web usage by ease the publishing of RDF triples from existing Web applications. It aims also to manage large amount of data with a flexible and scalable approach. In a lesser degree, Triplify provides pre-existing mapping for popular Web applications (Drupal, WordPress).
Triplify appears to be a good system for translating relational data into linked data. It presents a lot of serious advantages, such as the maintenance of expansive computation to the database, the use of the mature SQL language and its high scalability.

However, there are several drawbacks to use Triplify. First, the standard is maintained by very few people and documentation is hence really poor, unlikely standards developed by big groups such as W3C. Second, the intention of developers is to not implement extensions such as SPARQL support or other improvements. Finally, even if it is an express wish of the developers, it is sad that a mapping for RDF is not an RDF graph itself. This would have allowed some extra possibilities of data processing and reuse with a mapping suited for Web 3.0.

2.3.4 db2triples

db2triples is an free open-source Java implementation of the Direct Mapping mechanism and the R2R Mapping Language (R2RML). This component has been developed and released by Antidot\(^3\), an enterprise developing tools for linked data, among others.

db2triples reaches the goals established by the W3C\(^4\), namely integrate data from SQL databases into the linked data, give the possibility to manipulate them with SPARQL and gather them with other data sets. This implementation is maintained by the community on GIT repository\(^5\).

The main feature of this tool is reflected in the use of data from complex databases, where it can quickly and easily find the links between the data scattered across multiple tables, and enrich them through the power of SPARQL 1.1. Moreover, this tool supports WordPress CMS and e-commerce sites based on Magento solution.

The use of db2triples is quite easy. It offers a command-line interface. Two entry-points for the application are provided: one for using Direct Mapping and the other to use R2RML. Both translation processes require the specification of a database in input (with eventual parameters for the connection). When using R2RML, a mapping file must be specified as well. db2triples dumps the result of the translation into user’s file.

2.3.5 Comparative Analysis

In this section, we have discussed the (non-exhaustive) different standards and implementations that define methods to translate relational data into RDF. Each of those has their advantages and drawbacks. Therefore, we propose a comparative analysis retaining the aforementioned existing mapping approaches. Table 2.2 summarizes their main characteristics.

\(^3\)http://www.antidot.net/
\(^4\)http://antidot.net/fr/Actualites/Produit/Antidot-fournit-db2triples-en-Open-Source
\(^5\)https://github.com/antidot/db2triples
### 2.3. MAPPING TOOLS

<table>
<thead>
<tr>
<th>Mapping Tools</th>
<th>Advantages</th>
<th>Drawbacks</th>
</tr>
</thead>
</table>
| **R2RML**     | - W3C recommendation  
- Customizable mapping  
- Concise mapping vocabulary | - No support for XML sources |
| **D2R Server** | - RDF Views and RDF Dumps  
- Customizable mappings  
- Complete mapping language | - D2RQ Mapping language hard to implement  
- No support for XML sources  
- Connection to datasources implemented in the mapping language |
| **Openlink Virtuoso Universal Server** | - All-in-one server  
- Supports plenty of connection types  
- Integration with other services  
- RDF Views | - Heavy-weight  
- Commercial  
- No support for XML sources  
- Unmodifiable implementation |
| **Triplify** | - Fast execution (SQL level)  
- Easy to learn and use  
- Customizable mapping | - Impossible to use with XML  
- Mapping language in SQL (not RDF-graph) |
| **db2triples** | - Light-weight  
- Free and open-source  
- Direct mapping and R2RML implementation | - R2RML version outdated  
- No support for XML sources |

Table 2.2: Comparative table summarizing the main characteristics of the existing mapping approaches.
Chapter 3

Mapping Context

3.1 The Synergy Reference Model

This section presents the Synergy Reference Model (SRM) [26] for a better practice of data provisioning and aggregation processes for e-science. Today, these processes have become a complex challenge since the increasing amount of heterogeneous data is becoming available on the Internet. Thus, there is the need for data aggregation and integration in order to create rich resources useful for a range of different purposes. This model defines a set of business processes, user roles, generic software components and open interfaces that will form a harmonious whole.

The main principle behind this model is the following: Managing heterogeneous cultural heritage data is a complex challenge. Member institutions like galleries, libraries, archives and museums curate different types of collections that, even between similar types of institutions, are documented in different ways using different languages; influenced by different disciplines, objectives and geography, and are encoded using different metadata schemas. However, handling these metadata as a unified whole is vital for progressing new fields of humanities research and discovery, providing more knowledgeable information retrieval and (meta) data exchange, and advancing the field of digital humanities in its various aspects. Despite the fact that this model is referred to cultural data, it is obvious that it can be applied to other domains, too.

In general, data aggregation and integration seem to be a difficult but inevitable task. In recent years, this problem concerns the expert users of this field, because it is a powerful weapon to create useful resources for many fields and make a move towards the Semantic web. There are now significant numbers of projects that aggregate data with these purposes in mind. However, aggregators face two problems.

Firstly, the process of transferring data from source institutions to a central repository can result in a form of data representation stripped of essential information and institutional perspectives. This occurs when mandating target models into which all data sources must fit, regardless of their range, individuality and richness.
The generalizations used in these models, designed to facilitate data integration, are too abstract to support the meaningful connections that undoubtedly exist in the data and thus significantly reduce the value of such aggregation initiatives.

The second problem, addressed by this document, relates to the lack of sustainability in the mechanisms and processes through which data is mapped, and the weakness of the partnership between data providers and aggregators inherent in these flawed processes. The mechanisms used for transferring data do not include the full set of necessary processes and tools to create a consistent and good quality outcome and furthermore cannot practically respond to changes in schema and systems on either side of the data provisioning relationship. In order for systems to be sustainable a broader approach is needed that incorporates the experience and knowledge of provider institutions into the infrastructure, in an accessible, beneficial and cost effective manner.

Therefore we refer a new data provisioning model, the "Synergy Reference Model" (specifically the provision of data between providers and aggregators) including associated data mapping components. The intention is to address the design flaws in current models and crucially incorporate, through additional processes and components, the required knowledge and input from providers to create good quality, sustainable aggregations. The funding allocated to humanities aggregation projects over the last two decades has not generated the benefits and progress enjoyed in other sectors who have taken better advantage of digital innovation by using solid and inclusive infrastructures. Unless the value of these infrastructures is clearly demonstrated in the cultural heritage sector, resources for building and developing them will become even more scarce. Unfortunately, numerous systems initiated by projects in both Europe and the United States have failed to understand and identify the relationships and activities necessary to operate collaborative aggregation systems properly and instead have relied on one-sided and centralized approaches using top down modeling and technology led solutions.

The goal of SRM is to:

- describe the provision of data between providers and aggregators including associated data mapping components
- address the lack of functionality in current models (i.e., OAIS [36]) and practice (all European research infrastructures and some Digital Libraries)
- incorporate the necessary knowledge and input needed from providers to create quality sustainable aggregations
- define a modular architecture that can be developed and optimized by different developers with minimal inter-dependencies and without hindering integrated UI development for the different user roles involved.

The Synergy Reference Model will be described in terms of a formal process model. The process model requires the definition of the individual roles, data objects and processes necessary for designing a controlled and managed mapping
system. Figure 3.1 presents a high level view of the data provisioning process. The Provider Institution and the Aggregator Institution agree on the data provisioning and related activities. A "Mapping Manager" is nominated by both parties to oversee the actual data transfer process and forms the third primary role in the Synergy Model. The Provider Institutions own the Provider Records which are transformed to the Aggregator Record Format and are transferred to the Aggregator Institution. The data provisioning process is regarded as an open-ended and on-going task. Throughout the transformation and transfer processes, consistency checks and updates are necessary between all partners and will be supported by the model. The model foresees a series of distinct update processes at all partner sites which trigger each new data transfer.

The main steps of the data provisioning workflow are:

- **Schema matching**: Source and target schema experts (a.k.a the domain experts) define a schema matching which is documented in a schema matching definition file. This file should be human and machine readable and it is the ultimate communication mean on the semantic correctness of the mapping.

- **Instance generation specification**: In this step the URI generation and datatype conversion policies are defined for each instance of a target schema class referred to in the matching. In this step only IT experts are involved and domain experts have no interest or knowledge about it.

- **Terminology mapping**: Finally, the terminology mappings between source and target data/terms are defined. Providers may use anything from intuitive lists of uncontrolled terms up to highly structured third party thesauri.

- **Transformation**: Once the mapping definition has been finalized (and all syntax errors are resolved) the data needs to be transformed. The transfor-
mation process itself may run completely automatically. In the case where any issues arise, the aggregator can resolve them on a temporary or permanent basis but it is also possible that these records are send back to the provider for further analysis and resolution. Finally, the result is a set of valid target records.

- **Ingestion**: Once records are transformed, an automated translation for source terms using a terminology map follows. The transformed records will then, be ingested into the target system.

- **Change detection**: After the ingestion of the records all changes that may affect the consistency of provider and aggregator data are monitored. SRM describes 18-20 different updating and transformation reasons and is the only framework at the moment which takes the maintenance into account.

### 3.2 X3ML Language

The X3ML is an XML based language that describes mappings which can be created and discussed collaboratively by experts. Until now, the mappings were created manually. For this reason, the X3ML emphasizes on a standardized description of a mapping which aims to a better cooperation and creation of mapping memory in order to accumulate knowledge and experience.

The X3ML language\(^1\) designed and implemented by the Foundation for Research and Technology (FORTH) in 2006 [33], which is human readable and generic enough to cover most of the data models used nowadays. This model carefully distinguishes between mapping information from the domain experts who knows and provide the data and that created by the IT technicians, who actually implement data translation and integration solutions, and serves as an interface between both. Primarily, it was designed to follow the principle DRY (avoiding repetition) [45] and be more clear about the URI generating process.

Moreover, X3ML separates schema mapping from the concern of generating proper URIs so that different expertise can be applied these two very different responsibilities. The URI expert must ensure that the generated URIs match certain criteria such as consistency and uniqueness, while the Schema experts only need to concern themselves with the proper interpretation of the source.

#### 3.2.1 Global Structure

By studying the global structure of the X3ML language, we could say that it is quite understandable and properly structured. The general format is consisted of a specific shape and it provides a series of relations between the source and target schemata.

\(^1\)https://github.com/delving/x3ml/blob/master/docs/x3ml-language.md
3.2. X3ML LANGUAGE

Below we can see the general form of X3ML. Each X3ML mapping contains a root element with name "x3ml". It contains a "namespaces" and a "mappings" element. The first includes all namespaces that we can use in the mapping. The second element includes one or more "children" elements which express the transformation rules. The X3ML language uses the XML language in order to keep a structured form and XPath to express selectors. These are the main technologies in order to create a mapping from scratch, while Table 3.1 depicts the main structure of X3ML language.

```
<x3ml>
  <namespaces/>
  <mappings>
    <mapping>
      <domain>
        <source_node/>
        <target_node/>
      </domain>
      <link>
        <path>
          <source_relation/>
          <target_relation/>
        </path>
        <range>
          <source_node/>
          <target_node/>
        </range>
      </link>
      <!---- more links ---->
    </mapping>
    <!---- more mappings ---->
  </mappings>
</x3ml>
```

Table 3.1: Sample of X3ML

3.2.2 Mapping Structure

Each mapping rule is expressed in a "mapping" element where each of them contains the elements domain and range. These elements include target nodes which can either include or create URIs or represent literals to specify the related details of the ontology. The target blocks can also include criteria which restrict a mapping in order to meet the requirements of the transformation and may permit the direct creation of nodes and relationships.

Each class-property-class of the source schema is mapped individually to the global schema and can be seen as self-explanatory, context independent proposition. An x3ml structure consists of:
• the mapping between the source domain and the target domain
• the mapping between the source range and the target range
• the proper source and target path
• the mapping between source path and target path

In Figure 3.2 we can see the general form of a mapping and its features.

3.2.3 X3ML by Example

In this sub-section we will implement a simple example in order to understand the syntax of X3ML. This example concerns the Coin Example trying to convert a database and express the mappings through X3ML.

Let us first explain the source schema (relational database). We suppose that the database contains 10 different tables with primary and foreign keys which link to each other. In our example, we will use the main table $COIN$ in order to create direct links to the target schema. Also, we define as target schema the $CIDOC-CRM$ [49, 24] which provides definitions and a formal structure for describing the implicit and explicit concepts and relationships used in cultural heritage documentation [28].

The first step is to translate the database schema as an XML document. For example, the $COIN$ table is translated to an element which contains all the records expressed in elements regarding the column name. The Figure 3.4 shows the database’s schema and its conversion to XML.

Then, we try to map the $ID$ column of the $COIN$ table. More specifically, this table has a primary key which identifies each record. Also, the target
schema contains descriptions about a class $E22\text{Man} – \text{MadeObject}$ and a property $P1\text{isIdentifiedby}$. We map each coin record as an instance of the $E22\text{Man} – \text{MadeObject}$ class, the relation of the record and the ID as $P1\text{isIdentifiedby}$ and the column ID as $E41\text{Appellation}$. The Figure 3.5 illustrates the above mentioned transformation.

Taking into account the above-mentioned transformation the Table 3.2 shows the X3ML syntax which describes these rules. The "crm:" namespace is defined in element namespaces. There is a mapping element which contains a domain and a link element. The domain converts each record of $\text{COIN}$ table as "crm:$E22\text{Man-Made Object}$", the link element defines that the column "ID" is translated as property "crm:P1 is identified by" and finally the value of this column is type of "crm:E42 Identifier".
Figure 3.5: Mappings of coins

Table 3.2: X3ML mapping of Coins
3.3. THE REPOSITORY AND OTHER TOOLS

3.3 The Repository and Other Tools

3.3.1 The X3ML Engine

The X3ML engine [41] takes as input the source data (currently in the form of an XML document), the description of the mappings and the URI generation policy file and is responsible for transforming the XML document into a valid RDF document which is equivalent with the XML input, with respect to the given mappings and policy. The engine has been originally implemented in the context of the CultureBrokers project co-funded by the Swedish Arts Council and the British Museum. It has been developed by Delving BV in close collaboration with FORTH and is currently being maintained by FORTH.

The X3ML Engine has been designed and developed with respect to the following design principles:

- **Simplicity.** It is easier to create complicated things than it is to find the simplicity in something that would otherwise be complex. One important way to achieve simplicity and clarity is by carefully naming things so that their meaning is as obvious as possible to the naked eye.

- **Transparency.** The most important feature of X3ML is its general application to mapping creation and execution and hopefully its longevity. People must be able to easily understand how it works. The core design of this engine and X3ML language, and the clearer its documentation, the more readily it will get traction and become the basis for future mappings.

- **Collaborative Mapping Memory.** The X3ML mapping descriptions must lend themselves to being stored and handled by collaborative tools, as well as potentially written by hand. This was the motivation for choosing a simple syntax in XML, and one which does not depend on implicit knowledge.

- **Re-use of Standards and Technologies.** The best way to build a new software module is to carefully choose its dependencies, and keeping them as small as possible. Building on top of proven technologies is the quickest way to a dependable result.

- **Facilitating Instance Matching.** An application of X3ML which came up during discussions at the beginning of this project involved extracting semantic information with the intent of finding correct instance URIs. This implies a relatively small extension to the original idea of the X3ML engine because it will have to provide modified source records as well as RDF in its output.

Having these principles in our minds we implemented the X3ML engine. The X3ML engine framework has been implemented in Java, producing a single artifact in the form of a JAR file which contains the engine software. Figure 3.6 depicts the main components of the framework. The Input Reader component is responsible
for reading the input data (currently we support XML documents, however more formats will be supported using proper extensions). The X3ML Parser component is responsible for reading and manipulating the mappings which are described in the form of an XML document. The component RDF Writer outputs the transformed data into RDF format. The Instance Generator component produces the URIs and the labels based on the descriptions that exist in the mappings and finally the Controller component coordinates the entire process.

The X3ML Engine takes as input source XML records and generates RDF triples consisting of subject, predicate, and object. The subject and the object are “values”, generally consisting of URIs, but objects can also be labels or literal values.

The generation of values (URIs, or literals) is being handled by the Instance Generator component. The following block shows two configurations; for generating (a) URIs and (b) label values.

![Figure 3.6: The main components of X3ML Engine](image)

### 3.3.2 3M Editor

The 3M Editor combines the X3ML language and the X3ML Engine, providing a Graphical User Interface (GUI) in order to facilitate the total mapping process. It provides an easier way to create mappings graphically and it is connected with the engine to generate instantly triples from the source to the target schema.

As we described in previous section (§ 3.2), the initial form of X3ML is very difficult to be processed by non-experts. Therefore, FORTH has developed this platform in order to manipulate and create mappings graphically. The new form of mappings is displayed through tables.

Also, the X3ML Engine requires an IT expert to import the source, the target schema and the mappings in order to generate the RDF format. The 3M Editor can facilitate this process by connecting the mappings with the engine automatically.

Overall, the main functionalities of the 3M Editor provide the user with the capability to:

- Provide info about the mappings
3.3. THE REPOSITORY AND OTHER TOOLS

- Import Source Schema in XML format
- Import Target Schema (e.g. OWL format)
- Be informed about suggestions during the mapping process based on the schemata
- Create mapping graphically
- See and edit the initial form of X3ML
- Define policies and generate URIs
- Use the X3ML Engine in order to generate the final result

Figure 3.7: 3M Editor - Info page
CHAPTER 3. MAPPING CONTEXT

Figure 3.8: 3M Editor - Mapping table
Chapter 4

Proposed Approach

This section describes the proposed approach of this work. At first we provide Business Use Cases regarding the main purposes of mapping process (§ 4.1), then we describe the basic roles and actors of system (§ 4.2), we present the requirements (§ 4.3), we describe the lower level of mapping process by defining system use cases (§ 4.4), and finally we present the proposed functionality of system (§ 4.5).

4.1 Business Use Cases

A Business Use Case aims to describe the way in which the actors or some other interested party can make use of the system to get the desired outcome. This is the main purpose of the system’s usage. Therefore, we define the main business use cases taking into account the main purposes of mapping process. Overall, we define two cases: (a) Create a Mapping, which describes the process in which the actor wants to generate mappings from scratch, (b) Improve a Mapping, in this case the actor has already a mapping and tries to improve it, check its quality, compare it with others, monitor its evolution etc.

4.1.1 Create a Mapping

The first Business Use Case is to create a mapping from scratch. This case constitutes the initial purpose of the mapping process. As we already have mentioned in previous chapter, the current Web data is in a format understood only by humans and not by machines. For this reason we need to convert this data into a well-structured format in accordance with the terms and principles of Semantic Web. So we define a process to make this conversion. When we refer to this process, we will name it “mapping process”. The initial scenario requires an available resource in order to process the data and create mapping rules for proper conversion.

Since there are many produced vocabularies (ontologies) in order to achieve an effective homogenization of heterogeneous data, the initial intention would be to understand the source and target schemas. Therefore, the participation of a Source
Schema Expert user would be necessary, as the later would be able to understand the source contents and determine the semantics. Moreover, in an ideal scenario would be involved a Target Schema Expert, who would be able to create correlations between the two schemata. Finally the process would become more efficient and less time consuming if there was IT Expert’s involvement, in order to translate the correlations to mapping rules of a specific language format.

As regards this business use case, there are several sub-cases:

1. **Explore Source Schema.** The first step of a mapping process is to understand the source schema correctly. We need to have a quick overview and a global structure in order to be familiar with the source data and their semantics.

2. **Explore Target Schema.** Another sub-case is the exploration of target schema. In order to create proper associations and an effective mapping, we need to understand the target schema. The main purpose of this case is to identify if the target schema meets the requirements of the source schema and if it is capable to cover all entities.

3. **Create mappings properly.** Finally, the main sub-case is to create mapping rules step by step to ensure a correct transformation. The actors should identify the high-level entities in order to start the mapping and continue with nested elements.

### 4.1.2 Improve a Mapping

Studying Business Use Cases and taking into account the mapping process, there is a case to have implemented an initial mapping between source and target schema. In this scenario, additional requirements are created to improve the existing mapping rules. This case does not concern direct mapping methods, because they target in short time consuming conversions and do not concern optimization. Thus, some of the derived requirements are to improve mappings, check its quality, compare with others, monitor its evolution etc.

A more detailed description of sub-cases follows:

1. **Optimization.** An existing mapping needs optimization in order to modify its rules to make some aspect of it work more efficiently and keep the semantics. In general, a mapping may be optimized as such it includes more concepts from the source schema, or is capable of supporting a correct conversion of data to avoid wrong derivations, or consume the maximum number of the target schema’s capabilities.

2. **Percentage Coverage.** Since a mapping has been created, the user needs to check its consistency and quality. The percentage coverage provides an effective way on confirmation and validation of mapping rules regarding the schemas.
3. **Comparison.** It is a common case that there are various approaches regarding the mapping process. Although source and target schema may be the same, it is observed that users can translate source schema to different correlations. Thus, we have to compare different approaches.

4. **Evolution Monitoring.** Another case is the monitoring of mapping’s evolution. During the time, there are various versions regarding the same mapping. Versioning is based on the current scope and purpose of the mapping. For instance, the derived data may target to different purposes between a time span in order to serve users’ requirements. The system provides user with the capability of evolution monitoring of a mapping over time.

4.2 **Actors**

An actor is a person or other entity being specified externally to the software system who interacts with the system and performs use cases to accomplish tasks. Different actors often correspond to different user classes, or roles, identified from the users’ community. We define three main actors who interact with the mapping process: (a) **Source Schema Expert**, (b) **Target Schema Expert**, and (c) **IT Expert**. An extended description of these actors follows.

(a) **Source Schema Expert.** The Source Schema Expert is the actor who is responsible for the source schema and he has little or no programming-level knowledge. He knows the describing concept, the dataset and the structure in general of the source schema. Also, he is able to understand the semantics of the concept and the existing links by knowing correlations of existing elements. He knows the variables of the elements and their content (e.g. a numeric field for weight can be in kilograms or pounds). Furthermore, the Source Schema Experts are able to explain easily a given cell but it is very difficult to create a mapping. Usually, these particular actors are the creators or managers of the source datasets.

(b) **Target Schema Expert.** The Target Schema Expert is the actor responsible for the target schema (ontology) and has little or no programming-level knowledge. The majority of target schema experts use a more widely known standard schema, but there is also a growing trend in the linked data world towards custom ontologies. Also, these experts are aware of the principles of the Semantic Web and technologies such as RDF/S and OWL. He is an expert regarding the identification of classes, properties and their specifications. He knows the semantics of ontologies and he is able to map correctly a given concept to the target schema.

(c) **IT Expert.** The IT Expert has high programming-level knowledge but he has no experience regarding the target and source schemata. Usually, he is aware of used technologies about the schemata but he is not able to create a
mapping since he lacks of experience. However, he is more effective in URI generation process since he knows the mapping language. In our case, we assume that the main mapping language is X3ML and for this reason the IT Expert can be named X3ML Expert.

Typically but not exclusively, an identical scenario would be the presence of all above-mentioned actors in order to create a correct and efficient conversion from a source to a target schema. The combination of these actors could create a mapping which can describe all concepts effectively. Source schema expert can easily describe the meaning of the source and provide information about concepts. Target schema expert can understand the ontology schema and define the correlations between source and target schema. Finally, the IT expert can translate these correlations to a mapping language and define URIs.

However, total experts’ participation is quite difficult to be achieved in most cases. If we assume that we have two different actors, the second best combination could be the participation of a target and source schema expert, because the most difficult tasks are to understand the source and target schema, since it is considered easier to understand and learn a mapping language.

4.3 Requirements Analysis

This section provides a detailed requirements analysis which are based on business use cases and describes the decomposition of user needs into clear, achievable, and verifiable high-level requirements. We define capabilities that link the users' needs to the system, system elements, and enabling system elements to be designed and developed. Also, the objective is to define a system that meets users' operational requirements.

In brief, we separate requirements into two categories based on business use cases. Overall, we define 9 basic requirements and 21 sub-requirements which provide a complete analysis of our system.

Requirements of “Create Mapping” Business Use Case

Req1. **Explore / Navigate Source Schema**

The system should provide means for an actor to explore and navigate through the source schema. Also, the user can have a quick overview and the global structure of the source schema.

Req1.1. **Source schema Overview**

The system should provide a source schema overview, providing details about the total number of tables, total entities per tables, etc.

Req1.2. **Source Schema structure and associations**

The system should illustrate the source schema’s structure drawing a graph of tables and associations among primary and foreign keys.
4.3. REQUIREMENTS ANALYSIS

Req2. **Explore / Navigate Target Schema**
The system should provide the actors with the capability of exploring the target schema offering an effective way to understand it and its features.

Req2.1. **Target Schema Overview**
The system offers a detailed overview of the target schema. It provides information and statistics about the total number of classes/subclasses and properties/subproperties.

Req2.2. **Target Schema structure**
The system should draw a detailed graph which includes the global structure of the target schema.

Req2.3. **Hierarchy (Classes-subclasses, Properties-subproperties)**
In most cases, actors have to understand the target schema's hierarchy, since the semantic web includes this capability. Therefore, the system should clarify the hierarchy of classes and properties providing their correlations between subclasses and subproperties, respectively.

Req2.4. **Triangles of Classes**
Users can focus on a particular class in order to identify triangles between other classes. A triangle of class is the involvement of three classes and three properties in a row where the beginning and the end are the same class.

Req3. **Propose mapping rules**
The system should suggest mapping rules providing an initial mapping in order to facilitate users.

Req3.1. **Map high level Entities**
Starting the mapping process, the system should suggest mapping rules in order to map high level entities of the source to the target schema.

Req4. **Depth of Current mapping state**
During the process, experts have to map many instances which include child elements thereby making the process more difficult. Therefore, system should inform users about the current state and the depth of mapping process.

**Requirements of “Improve Mapping” Business Use Case**

Req5. **Mapping Overview**
Since a mapping is already defined; the system should provide an overview of the rules and its general characteristics.

Req5.1. **X3ML-based overview**
System should provide user with the capability of monitoring the consistency and hierarchy of the existing mapping. It could provide a graph
with all mapping rules focusing on syntactic elements of the X3ML language.

Req6. **Optimization / Check quality**
Another important system’s capability is the optimization and quality check. The actors can navigate through correlations and see the percentage coverage about specific concepts.

Req6.1. **Correlations between source and target schemata**
The system should illustrate how the mapping is linked with the source and target schema. More precisely, it would provide correlations between them.

Req6.2. **Percentage coverage of schemas**
The system should provide information about current percentage coverage of both source and target schemata. As regards the source schema, it would draw the total number of used entities and instances concerning the source schema. Also, not only total number of used classes would be shown, but also the properties of target schema regarding current state of mapping.

Req6.3. **Multiple coverage metrics**
Regarding the percentage coverages, the system will provide different metrics of coverages in order to capture different use cases of actors.

Req6.4. **View mapping assertions involving a given concept**
The system shall represent all the mapping assertions involving a given concept from source and target schema. The representation must have filters in order to search concepts in mappings.

Req6.5. **Track Private Data or data for internal usage**
Users may have the capability of specifying data of the source schema as private. In this case system can verify if these data is included in a mapping or not in order to avoid their transformation.

Req7. **Mapping Comparison**
The comparison of several mappings constitutes an important method to define differences and track several approached regarding the mapping method. Thus, the system shall compare source schema, target schema and mapping rules.

Req7.1. **Textual comparison of two mappings based on mapping language**
The initial need of mapping comparison is a textual comparison based on the X3ML language. The system should provide a side-by-side projection marking the differences of the mapping.

Req7.2. **Graphical comparison of two or more mappings based on mapping rules**
4.3. REQUIREMENTS ANALYSIS

Another requirement is the comparison of two or more mappings graphically providing the different mapping approaches over same source and target schemata. The system should provide an overview how the mappings translate different concepts from the source to the target schema.

Req7.3. **Comparison of generated instances**

Despite of mapping-centered comparisons, there is the need of comparison regarding the generated results of the mapping. The system should compare generated instances between different mappings.

Req8. **Evolution monitoring**

Furthermore, there are various versions regarding the same mapping process. The versioning is based on the current scope and purpose of the mapping. Therefore, the system provides the user with the capability of monitoring the evolution of a mapping over time regarding the transformation rules and the evolution of the source and target schema.

Req8.1. **Tracking differences over time based on mapping language**

Firstly, the actors should have the capability of monitoring syntactic changes of the mapping.

Req8.2. **Tracking differences over time based on mapping rules**

Also, the system shall track the changes about how an actor translates the source schema to the target schema during a time period.

Req8.3. **Tracking differences over time based on schemata**

The system should track the changes based on the source and target schema selecting a specific time period.

Req9. **Explore / Navigate generated Instances**

Despite the analysis of pre-proceedings, the system allows the opportunity of exploring the generated instances regarding.

Req9.1. **Search instances**

The system shall provide search of generated instances in order to focus on specific generated concepts. This functionality includes both direct search of instances and instances based on their class.

Req9.2. **View generated correlations of instances**

Finally, the users would be able to explore a specific instance by displaying incoming and outgoing properties.
4.4 System Use Cases

The proposed functionality would be better understandable, if it was described all the possible ways, in which users act during the mapping process, in order to achieve an effective transformation rule. Below, all possible system use cases and links with mentioned requirements are described, with a view to study the mapping process step by step.

<table>
<thead>
<tr>
<th>Use Case Title:</th>
<th>Source schema Analysis</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description:</td>
<td>User has the ability to select or to import a source schema as well as to explore its structure, its elements and the relations between them. The form of the source file is based on the XML language.</td>
</tr>
<tr>
<td>Actors:</td>
<td>Source Schema Expert</td>
</tr>
<tr>
<td>Requirements:</td>
<td>Req1., Req1.1., Req1.2.</td>
</tr>
<tr>
<td>Proposed functionality:</td>
<td>Explore Source Schema</td>
</tr>
</tbody>
</table>

Table 4.1: Source schema Analysis

<table>
<thead>
<tr>
<th>Use Case Title:</th>
<th>Target Schema Analysis</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description:</td>
<td>User can import or select a mapping and explore its structure, its classes/subclasses and properties/sub-properties. Also, he can view the connections between elements.</td>
</tr>
<tr>
<td>Actors:</td>
<td>Target Schema Expert</td>
</tr>
<tr>
<td>Requirements:</td>
<td>Req2., Req2.1., Req2.2., Req2.3., Req2.4.</td>
</tr>
<tr>
<td>Proposed functionality:</td>
<td>Explore Target Schema</td>
</tr>
</tbody>
</table>

Table 4.2: Target Schema Analysis

<table>
<thead>
<tr>
<th>Use Case Title:</th>
<th>Ignore Private Data or data for internal usage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description:</td>
<td>An expert has to be careful with data which might be private or to be used for internal usage only. Thus, we have to be sure that a part of source schema is not mapped (e.g. private data). User selects the private data from source schema and system returns/responds if it is involved in mapping or not.</td>
</tr>
<tr>
<td>Actors:</td>
<td>Source Schema Expert, Target Schema Expert</td>
</tr>
<tr>
<td>Requirements:</td>
<td>Req6.1., Req6.5.</td>
</tr>
<tr>
<td>Proposed functionality:</td>
<td>Define undesired data for mapping</td>
</tr>
</tbody>
</table>

Table 4.3: Ignore Private Data or data for internal usage
### 4.4. SYSTEM USE CASES

<table>
<thead>
<tr>
<th>Use Case Title</th>
<th>Percentage coverage of source schema</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Description:</strong></td>
<td>The system includes information about the current percentage coverage of source schema. It draws the total number of used entities and instances concerning the source schema. It indicates the total usage of source schema from the mapping.</td>
</tr>
<tr>
<td><strong>Actors:</strong></td>
<td>Source Schema Expert</td>
</tr>
<tr>
<td><strong>Requirements:</strong></td>
<td>Req6., Req6.2., Req6.3.</td>
</tr>
<tr>
<td><strong>Proposed functionality:</strong></td>
<td>Percentage coverage of source schema</td>
</tr>
</tbody>
</table>

Table 4.4: Percentage coverage of source schema

<table>
<thead>
<tr>
<th>Use Case Title</th>
<th>Percentage coverage of target schema</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Description:</strong></td>
<td>The system shows information about the current percentage coverage of target schema. It draws the total number of used entities and instances concerning the target schema. Also, a graph depicts the total number of used classes and properties of the target schema regarding the current state of mapping. It indicates the total usage of target schema from mapping.</td>
</tr>
<tr>
<td><strong>Actors:</strong></td>
<td>Target Schema Expert</td>
</tr>
<tr>
<td><strong>Requirements:</strong></td>
<td>Req6., Req6.2., Req6.3.</td>
</tr>
<tr>
<td><strong>Proposed functionality:</strong></td>
<td>Percentage coverage of target schema</td>
</tr>
</tbody>
</table>

Table 4.5: Percentage coverage of target schema

<table>
<thead>
<tr>
<th>Use Case Title</th>
<th>Mapping Overview of X3ML</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Description:</strong></td>
<td>An IT Expert needs to understand the transformation logic regarding an existing X3ML mapping over a specific concept and get a general overview of the mapping. The system returns a graph displaying the consistency and hierarchy of the existing mapping.</td>
</tr>
<tr>
<td><strong>Actors:</strong></td>
<td>IT Expert</td>
</tr>
<tr>
<td><strong>Requirements:</strong></td>
<td>Req5., Req5.1.</td>
</tr>
<tr>
<td><strong>Proposed functionality:</strong></td>
<td>X3ML-based Overview Graph</td>
</tr>
</tbody>
</table>

Table 4.6: Mapping Overview of X3ML
### Use Case Title: Mapping assertions between mapping and source schema

**Description:** User can get informed about the way a mapping is linked with the source data by selecting a mapping and the related source schema. It is also included a graph with the above information which are connected with edges describing theirs correlations.

**Actors:** Source Schema Expert

**Requirements:** Req6.1., Req6.4.

**Proposed functionality:** Mapping-based Representation

<table>
<thead>
<tr>
<th>Use Case Title</th>
<th>Mapping assertions between mapping and source schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description</td>
<td>User can get informed about the way a mapping is linked with the source data by selecting a mapping and the related source schema. It is also included a graph with the above information which are connected with edges describing theirs correlations.</td>
</tr>
<tr>
<td>Actors</td>
<td>Source Schema Expert</td>
</tr>
<tr>
<td>Requirements</td>
<td>Req6.1., Req6.4.</td>
</tr>
<tr>
<td>Proposed function</td>
<td>Mapping-based Representation</td>
</tr>
</tbody>
</table>

Table 4.7: Mapping assertions between mapping and source schema

### Use Case Title: Mapping assertions between mapping and target schema

**Description:** User can select a mapping and the related target schema and get informed about how the mapping is linked with the target data. It is also included a graph of the mapping and target schema which are connected with edges describing theirs correlations.

**Actors:** Target Schema Expert

**Requirements:** Req6.1., Req6.4.

**Proposed functionality:** Mapping-based Representation

<table>
<thead>
<tr>
<th>Use Case Title</th>
<th>Mapping assertions between mapping and target schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description</td>
<td>User can select a mapping and the related target schema and get informed about how the mapping is linked with the target data. It is also included a graph of the mapping and target schema which are connected with edges describing theirs correlations.</td>
</tr>
<tr>
<td>Actors</td>
<td>Target Schema Expert</td>
</tr>
<tr>
<td>Requirements</td>
<td>Req6.1., Req6.4.</td>
</tr>
<tr>
<td>Proposed function</td>
<td>Mapping-based Representation</td>
</tr>
</tbody>
</table>

Table 4.8: Mapping assertions between mapping and target schema

### Use Case Title: View mapping assertions involving a given concept

**Description:** A user might be interested in specific concept of the source or target schema. Thus, system provides the user with the capability of selecting tables of the source schema or classes of the target schema and a graphical view focuses on them. Also, the user can filter concepts focusing on them.

**Actors:** Source Schema Expert, Target Schema Expert

**Requirements:** Req6.1., Req6.4.

**Proposed functionality:** Mapping-based Representation

<table>
<thead>
<tr>
<th>Use Case Title</th>
<th>View mapping assertions involving a given concept</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description</td>
<td>A user might be interested in specific concept of the source or target schema. Thus, system provides the user with the capability of selecting tables of the source schema or classes of the target schema and a graphical view focuses on them. Also, the user can filter concepts focusing on them.</td>
</tr>
<tr>
<td>Actors</td>
<td>Source Schema Expert, Target Schema Expert</td>
</tr>
<tr>
<td>Requirements</td>
<td>Req6.1., Req6.4.</td>
</tr>
<tr>
<td>Proposed function</td>
<td>Mapping-based Representation</td>
</tr>
</tbody>
</table>

Table 4.9: View mapping assertions involving a given concept
### 4.4. SYSTEM USE CASES

<table>
<thead>
<tr>
<th>Use Case Title</th>
<th>Description</th>
<th>Actors</th>
<th>Requirements</th>
<th>Proposed functionality</th>
</tr>
</thead>
<tbody>
<tr>
<td>Compare two mappings</td>
<td>There is the powerful capability of comparing two mappings with same source and target schema in order to identify similarities or differences. User can select either textual or graphical comparison in order to compare the mapping language of mapping (X3ML) or track changes. Also, he is able to identify differences between the generated instances of mappings.</td>
<td>Source Schema Expert, Target Schema Expert, IT Expert</td>
<td>Req7., Req7.1., Req7.2., Req7.3.</td>
<td>Textual, Graphical and Instance Comparison</td>
</tr>
<tr>
<td>Tracking changes over time</td>
<td>There are various versions regarding the same mapping over a specific time period. Therefore, if the system has a log of mappings, the system provides the capability of monitoring the evolution of a mapping over time regarding transformation rules and the evolution of the source and target schema. The user can drag a bar and explore the changes.</td>
<td>Source Schema Expert, Target Schema Expert, IT Expert</td>
<td>Req8., Req8.1., Req8.2., Req8.3.</td>
<td>Textual, Graphical and Instance Comparison</td>
</tr>
<tr>
<td>Analysis of generated instances</td>
<td>A significant capability is the exploration of generated instances. The user searches direct instances or instances depending on their classes. Moreover, he is able to explore incoming and outgoing properties via a graph and navigate among them.</td>
<td>Source Schema Expert, Target Schema Expert, IT Expert</td>
<td>Req9., Req9.1., Req9.2.</td>
<td>Instance representation</td>
</tr>
</tbody>
</table>

Table 4.10: Compare two mappings

Table 4.11: Tracking changes over time

Table 4.12: Analysis of generated instances
4.5 Proposed Functionality

Since system requirements and use cases have been defined, in this section it will be presented the proposed functionality of this work regarding the management of mappings. The main core and structure is described below in brief:

- Firstly, we try to formalize and express the mapping definitions providing the preliminaries of the proposed work.
- The following part refers to several techniques of schema analysis. We propose detailed exploration of source and target schema in order to understand the available resources.
- Furthermore, we introduce coverage metrics of schemas contributing to the understanding of the existing mapping.
- Another important point during the mapping process is the representation of mappings. The fourth subsection describes two efficient representations in order to depict the mapping rules.
- The next part proposes an effective way to compare mapping focusing on textual, graphical and instance comparison.
- Finally, we suggest other useful functionalities in order to cover several rising issues, such as the representation of generated instances and the evolution of the mapping over time.

4.5.1 Preliminaries of Mappings

Since the requirement analysis and the proposed approach are based on the components of Semantic Web and Mappings, below we provide the preliminaries regarding definitions which we will use throughout this work. Overall, it is given the definition of mapping identity as a combination of source, target, application, person and time.

**Definition 4.5.1.** DEF1 The first formalization regards a triple $<S, M, T>$ which expresses the mapping process, where $S$ is a source schema, $T$ is a target schema (ontology) and $M$ is a mapping between $S$ and $T$. More precisely, $S$ typically presents knowledge and is expressed in a language $L(S)$, whereas $M$ is specified in a language $L(M)$. In our case, we assume that $L(S)$ is in XML format while $L(M)$ is the X3ML mapping language. Finally, $M$ consists of a mapping set rules $M(R1,\ldots,Rn)$ where $Rn$ is a rule. It describes the correlations between $S$ and $T$ regarding a specific concept. A mapping rule is an expression of the form $R: \phi(\chi) \rightarrow \psi(\chi)$, where $\phi(\chi)$ is a query over the $S$ and $\psi(\chi)$ is a query over the $T$. An X3ML mapping file is a set of X3ML mappings $M$.

**Definition 4.5.2.** DEF2 Since the X3ML file has a standard format, we can define its elements. Each X3ML file consists of a set of info, namespaces and mappings.
4.5. PROPOSED FUNCTIONALITY

- The info node contains information about title, mapping, source schema, target schema and a sample of data.
- The namespaces node contains information about the available namespaces.
- The mappings node is a set of $M$ which contains:
  - A domain node specifying the concept from the target schema
  - A set of link specifying the “children” nodes of the domain

4.5.2 Explore Schemas

4.5.2.1 Explore Source Schema

According to the system use cases which were mentioned in previous section (§ 4.4), it is important to understand the source that we want to create the mapping. We need to understand the content, the information it contains and distinguish useful or undesirable fields according to mapping requirements. These features are not easy to be achieved by editing source in initial form of XML. Thus, the system provides user with the above-mentioned functionality in a friendly environment.

In detail, the current functionality has the following features. Firstly, user is able to select a mapping or to import the source schema before starting the mapping process. The system detects the structure of the schema and organizes it. Finally, system illustrates the structure and gives an overview of the source schema.

System provides the following capabilities:

1. **General Info.** System displays general information about the source schema such as name, version and comments. The type is identified providing different representations whether it is a relational schema or an XML based form.

2. **General Statistics.** System calculates the total number of parent and child elements according to the source schema providing an overview.

3. **Representation of schema.** Different kinds of representations are available in order to display the schema’s structure.

   (a) **Relational database.** In case of relational databases, the representation illustrates the available tables. For each table, user can see its attributes in order to navigate and understand the containing information (Figure 5.10). Also in most cases, the relational databases contain foreign keys. This functionality identifies these keys and connects them from one table to another.

   (b) **XML forms.** The most common case is XML based schemas. This representation illustrates the schema as tree displaying parent and child
elements. The user has the ability to focus on specific element expanding its contents (Figure 5.11).

4. **Mark tables as private data or undesired information for mapping.** In most cases, source contains data that cannot be transformed either because they are private data, or because they are undesirable concerning the mapping. For this reason, user is able to specify these elements with regard to avoid mapping and also to identify whether it is already mapped.

![Figure 4.1: Source Schema Transformation](image)

### 4.5.2.2 Explore Target Schema

As already discussed, it is very important the fact to know the target schema and its content in which we intend to match our source data. In most cases, ontologies are described in OWL or RDF/S format which is difficult to be understood by non IT Experts. For this reason, system provides a friendlier environment illustrating all the features of the target schema and providing a summary of its contents. Moreover, system can contribute to a total exploration in classes, properties and extra features.

Users can select a mapping in order to navigate and learn its target schema. It is also common to have more than one target schema involved in a mapping. In this case user selects a mapping and system creates several graphs depicting all the available ontologies. Furthermore, it is possible to isolate one or more target schemata for decreasing the range.

System provides the following capabilities:

1. **Statistics.** System offers a detailed overview of the target schema (Figure 5.13). It provides information and statistics about the total number of classes/subclasses and properties/subproperties.

2. **Hierarchy (Classes-subclasses, Properties-subproperties).** The system clarifies the hierarchy of classes and properties providing their correlations between subclasses and subproperties, respectively. These graphs are based on ISA structure depicting a tree based on parent-child relations.
4.5. PROPOSED FUNCTIONALITY

3. Classes. The system draws a detailed graph which includes the global structure of classes of the target schema. We can focus on a specific class and get informed about the label, URI, type, comments and subclasses. Also, an extended analysis of involved properties is available (Figure 5.19 and 5.20).

(a) Incoming Properties. This graph illustrates properties which their range concerns the focused class. Also, it gives a view of all classes which are the domain of these properties.

(b) Outgoing Properties. Furthermore, we are capable to identify properties which their domain is the focused class and all the classes which are the range of these properties.

(c) Triangles. We define as triangle three classes and properties in a row, where the domain class of first property is the same with range of the third property.

4. Properties. With reference to the properties, system draws a detailed graph which includes the global structure regarding the target schema. User can search and get informed about the label, URI, type, comments, domain properties and range properties.

5. Mark classes and properties as undesired information for mapping. User is able to specify classes in order to avoid mapping and identify if they are already mapped (Figure 5.14).

4.5.3 Percentage Coverage of Schemas

In most cases, during the transformation process from a source schema to linked data, typically users follow a straightforward technique. Although source and target experts are well aware of the source and target schema respectively, they do not know the percentage coverage of current mapping. Thus, our system provides the user with the capability of knowing the percentage coverage of source and target schema calculating the involved concepts and providing two different methods of metrics. Firstly, we describe the proposed metrics studying nodes inside trees and then how they can be integrated and applied in percentage coverages of schemas.

4.5.3.1 "Direct" and "Ancestors – descendants" Metrics

In order to capture several purposes of percentage coverages of schemas, we define two metrics a) "Direct Metric" and b) "Ancestors-descendants Metric". The first method illustrates the percentage coverage based on direct involvement of concepts in mapping, while the second is based on parent-child relations manipulating the concepts as trees.

Direct Metric

More precisely, the main purpose of Direct Metric is to handle each node as a distinct element ignoring relations between them. An element is considered as
covered if it is referred in the mapping directly. The percentage is the number of covered elements divided by the total number of elements.

\[
\text{Perc}_{DM} = \left[\frac{\text{total}(\text{Cov}(x_n))}{n}\right] \times 100
\]

\(x_n\) : element, \(\text{Cov}(x_n)\) : covered element

For instance, according to Figure 4.2 the total number of covered elements is 2 \((x_3, x_4)\), while the total number is 6.

\[
\text{Perc}_{DM} = \left(\frac{2}{6}\right) \times 100 = 33.3\%
\]

Ancestors – descendants Metric

Studying cases of mapping coverages, we concluded that it is not enough to count the direct covered elements. For example, if we map an element, we consider its children as covered. Therefore, the Ancestors – descendants Metric introduces a different method of coverage calculation. An element is considered as covered if there is a direct reference in the mapping or there is at least an ancestor which is referred. The percentage is the number of covered elements and their descendants divided by the total number of elements.

\[
\text{Perc}_{ADM} = \left[\frac{\text{total}(\text{Cov}(x_n) + \text{Desc}(x_n))}{n}\right] \times 100
\]

\(x_n\) : element, \(\text{Cov}(x_n)\) : covered element, \(\text{Desc}(x_n)\) : descendants count of \(x_n\)

An indicative example as regards the Figure 4.2, the total number of considered covered elements is 2 \((x_3, x_4)\), their descendants are 2 \((x_5, x_6)\) and the total number is 6.

\[
\text{Perc}_{ADM} = \left[\frac{(2 + 2)}{6}\right] \times 100 = 66.6\%
\]

![Figure 4.2: Tree structure](image)

4.5.3.2 Percentage Coverage of Source Schema

It is a common case that we need to know the percentage coverage of source schema during the creation of a mapping and at the end of the process. The user should
4.5. PROPOSED FUNCTIONALITY

be informed about the coverage of the entire source schema or be sure that he has covered the elements with high interest. Thus, system enables users to provide a summary regarding covered and uncovered concepts as well as an overview of coverage metrics. The system draws a graph depending on the source schema. The overview of the graph will be like a tree which each branch will show the descendant elements.

In order to specify the metrics in source schema coverages, we apply the “Direct” metric considering an element $x_n$ as covered if $\exists R: \phi(x_n) \rightarrow \psi(\chi) \in M(R1,\ldots,Rn)$ [Definition 4.5.1].

System provides the following capabilities:

1. **Metrics of Sample Data.** System displays the total number of both parent and child elements according to the source schema providing an overview.

2. **Coverage Metrics.** We calculate three different percentages which are related to source schema. The first percentage displays the coverage of parent elements, while the second counts only the source schema leaves existing in the mapping (Figure 5.21). Finally, the third percentage calculates the total coverage of the source schema.

3. **Graph of Covered Elements.** Conclusively, the system depicts the source schema as tree highlighting the covered elements (Figure 5.22).

4.5.3.3 Percentage Coverage of Target Schema

Usage of target schema facilitates mapping evaluation, providing feedback about covered resources of the ontology. The main purpose of this functionality is to get informed about the covered classes and properties regarding a specific mapping. This can be applied by integrating both “Direct” and “Ancestors-Descendants” Metric and illustrating a graph of classes and properties, by highlighting each used concept.

More precisely, according to Definition 4.5.1 we consider a resource $x_n$ as covered if $\exists R: \phi(\chi) \rightarrow \psi(x_n) \in M(R1,\ldots,Rn)$. The percentage coverage of target schemas applies both metrics for classes, properties and resources, calculating three different percentages for each metric.

System provides the following capabilities:

1. **Metrics of Sample Data.** System displays the total number of classes and properties according to the target schema providing an overview.

2. **Coverage Metrics.** As regards metrics, we provide three overview percentages (classes, properties and resources) for each metric, combining all the related target schemas. Figure 5.23 depicts an indicative example. Also, there are distinct metrics for each target schema in order to restrict the point of interest (Figure 5.24).
3. **Graph of Covered Classes and Properties.** Finally, the system depicts each target schema as two graphs (classes and properties) highlighting the covered resources. For instance, Figure 5.25 illustrates covered classes and properties of specific target schema.

### 4.5.4 Mapping Representations

Concerning the mapping overview, it provides the ability to understand the structure of an X3ML file and navigate through the assertions between source and target schema. Users can choose two different representations by selecting an X3ML file.

#### 4.5.4.1 X3ML-centered Representation

The first proposed approach tries to represent a single X3ML mapping as a graph. Regarding the Definition 4.5.2, each X3ML file consists of several mapping elements and each of them contains a domain element and a number of links.

This particular representation depicts each mapping as a node. As regards mapping nodes, graph shows the sourceNode and targetNode which are connected with the main mapping. Finally, we are able to navigate through links which are linked to each mapping. The Figure 4.3 illustrates an indicative example of this particular representation.

![Figure 4.3: X3ML-centered Representation](image)

Although this form of representation helps to a better understanding of a transformation rule, there are several problems raising through different mapping cases. This graph is an effective representative way in cases when the mapping does not contain many nested elements. Thus, we can navigate and understand easily the structure and the syntax of the X3ML rule.
Conversely, in cases which the rule has a large amount of elements (e.g. mappings or links), the graph becomes quite large and there is confusion. Also, in case which the size of the mapping is increased, users are not able to explore the graph easily, and they may are unable to follow the elements structure and hierarchy as it was defined by X3ML. These kinds of problems are handled by the following representation method which is suggested.

4.5.4.2 Mapping-centered Representation

Since X3ML intends to separate users into three different expert groups (Source, Target and IT expert) regarding the total mapping workflow, we have to support a Mapping-centered representation in order to provide a comprehensive view for all user groups. Taking into account the aforementioned requirement, we facilitate users by creating a three-dimensional interactive representation which illustrates the mappings between selected concepts from source and target schema.

This modeling scenario may contribute to both Schema and IT Experts. On the one hand, Schema experts can easily distinguish and manage concepts on the source and target schema. They can also distinguish correlations between specific concepts. The most important feature is that user has an abstract overview about the coverage of mapping rules. On the other hand, IT experts can easily find elements for conversion which are in structured form as described in source schema. Overall, in most cases this approach is the most efficient representation method regarding a single mapping.

As regards the procedure, we depict a three-dimensional scene which represents mapping rules based on source schema. This process uses an X3ML mapping as input, it creates the structure of source schema (tree or relational) and it illustrates assertions based on the X3ML language. In order to be more precise, Algorithm [1] describes in detail the generation of mapping rules regarding mapping elements according to Definitions 4.5.1 and 4.5.2. Also, we follow the same pattern about link elements.

System provides the following capabilities:

1. Interactive representation. Creating a 3D world, we implement an interactive representation which allows users to select concepts of schemas and explore relations between them.

2. Source schema structure. As regards the source, the system depicts two representations: XML forms (tree) and relational schema (Figure 4.4). We are able to explore source schemas, understanding their structure through elements and view "ancestor-descendants" relations between them.

3. Coverage. Despite the source schema structure, users are capable to view covered elements directly. If an element is not covered by the mapping, it has transparent substance in scene. Otherwise, it is colored with random colors.
Algorithm 1 Mapping rules generation procedure

1: Input: $X3ML, SourceSchema$
2: Output: $MappingRules, SourceSchema$(with relations)
3:
4: $MappingRules \leftarrow \emptyset$
5: for each mapping $M \in X3ML$ do
6: \hspace{1em} domainEntity $\leftarrow$ createEntity($M$/domain/targetNode/entity)
7: \hspace{1em} xpathExpression $\leftarrow$ getXpath($M$/domain/sourceNode)
8: \hspace{1em} for each element $E \in SourceSchema$ do
9: \hspace{2em} if xpathExpression refersTo $E$ then
10: \hspace{3em} createRelation($E \leftrightarrow domainEntity$)
11: \hspace{3em} $E$ has covered $\leftarrow$ true
12: \hspace{1em} end if
13: end for
14: $MappingRules \leftarrow$ add domainEntity
15: end for

4. **Mapping rules.** Interactive representation allows the selection of source elements and depicts their correlations according to mappings. Also, target schema elements can be expanded in order to explore more mapping rules.

5. **Target Schema structure.** The system draws a sub-graph of target schema which includes involved classes regarding the mapping. Also, we are able to identify incoming and outgoing properties, creating edges between them.

![Figure 4.4: Mapping-centered Representation](image-url)
4.5. Proposed Functionality

4.5.5 Comparison of Mappings

In this subsection we present a different perspective regarding the mapping process. We refer to the point that the mapping has come to an end and we try to compare different approaches regarding the same concept. It is a common case for experts to compare mapping approaches and it can be considered a powerful technique in order to observe different thoughts and evaluate several mapping cases. The proposed functionalities of comparison include three different techniques a) textual, b) graphical and c) instance comparisons between two mappings.

4.5.5.1 Textual Comparison

The correlation of mappings constitutes a major challenge for experts, such as finding the differences and comparing several approaches. Also, the comparison contributes to conversions of concepts regarding relative patterns. These cases are a real problem which is time consuming for the mapping process. This time can be reduced, providing a comparison of two mappings and presenting the points that differ completely or partly. Therefore, we propose the Textual Comparison by displaying a side-by-side projection based on the X3ML language and its syntactic elements.

The basic challenge of comparison is that the X3ML language ignores the order of its elements. As result, we cannot compare its elements in the initial form. For example, the following two X3ML files are equivalent considering that the rest parts of the mapping are the same:

Figure 4.5: Comparison of Mappings based on X3ML

In order to compare effectively the textual form, we try to re-order the nested elements of the mapping. We apply a simple algorithm based on mapping elements and their links. More precisely, for each mapping we sort alphabetically the link elements regarding the sourceRelation elements. Then, we find the sourceNode element for each mapping and sort them respectively. The Algorithm [2] describes in detail the whole procedure.
Algorithm 2 Order X3ML elements for Textual Comparison

1: **Input:** X3ML file
2: **Output:** comparable X3ML file
3: $SN, SR, mapElements, linkElements \leftarrow \emptyset$
4: **for** each mapping $M \in X3ML$ **do**
5: \hspace{1em} **for** each link $L \in M$ **do**
6: \hspace{2em} $SR \leftarrow$ add $sourceRelation \in L$
7: \hspace{2em} $linkElements \leftarrow$ add $L$
8: \hspace{2em} delete $L$ from $M$
9: \hspace{1em} **end for**
10: \hspace{1em} $SR \leftarrow$ short alphabetically($SR$)
11: \hspace{1em} **for** each $sourceRelation sr \in SR$ **do**
12: \hspace{2em} **for** each link $L \in linkElements$ **do**
13: \hspace{3em} if $sr = sourceRelation \in L$ then
14: \hspace{4em} $M \leftarrow$ add $L$
15: \hspace{3em} **end if**
16: \hspace{2em} **end for**
17: \hspace{1em} **end for**
18: $SN \leftarrow$ add $sourceNode \in M$
19: $mapElements \leftarrow$ add $M$
20: delete $M$ from X3ML
21: **end for**
22: $SN \leftarrow$ short alphabetically($SN$)
23: **for** each $sourceNode sn \in SN$ **do**
24: \hspace{1em} **for** each mapping $M \in mapElements$ **do**
25: \hspace{2em} if $sn = sourceNode \in M$ then
26: \hspace{3em} X3ML $\leftarrow$ add $M$
27: \hspace{2em} **end if**
28: \hspace{1em} **end for**
29: **end for**
4.5. PROPOSED FUNCTIONALITY

Since the pre-procedure is completed, the system can compare textually two mappings. In this view we mark the rows which differ partly or completely. The left side represents the first mapping, while the right side illustrates the second mapping. Each side contains the number of each row in order to help the comparison process.

Also, we mark the differences with different colors. The rows which remain the same do not have background color (white). In contrast, the differences have yellow or red background color depending how they differ. The yellow color indicates that the rows differ partly, while the red color shows that the mappings differ completely as regards this particular part. The Figure 4.6 shows the Textual Comparison of two mappings.

4.5.5.2 Graphical Comparison

Continuing to explore techniques for comparing two mappings, we propose a graphical representation of differences. In most cases, a simple textual comparison finding differences is not enough. Trying to make the comparison of rules simpler and more meaningful, we recommend a graphical representation which depicts all the differences based on Mapping-centered technique.

Graphical representation has been a powerful techniques to describe and understand data. In order to provide more sufficient functionality, this approach tries to find differences between two mappings and represent them as graph. This procedure constitutes a more complicated algorithm in a sense that we have to translate each mapping as graph, find the differences and draw a combining graph. Figure 4.7 draws in detail the related sets.

Firstly, in order to understand better our formalizations we provide visual example of graphical difference between two mappings. First step is to identify the different and common rules. Let two mappings $X3ML1$, $X3ML2$ and their rules $X3ML1(R1...Rn)$, $X3ML2(R1...Rn)$ respectively. We define the different rules as the Delta Explicit ($\Delta (G1 \rightarrow G2)$) of their rules taking into account mappings.
as graphs. The common rules are the union of each mapping rule set deleting the \( \Delta e(G_1 \rightarrow G_2) \) set.

\[
\Delta e(G_1 \rightarrow G_2) = \{ \text{Add}(R) | R \in G_2 - G_1 \} \cup \{ \text{Del}(R) | R \in G_1 - G_2 \}
\]

\[
Com(G_1, G_2) = \{ G_1 + G_2 \} - \Delta e(G_1 \rightarrow G_2)
\]

![Figure 4.7: Definition-Comparison of Mappings](image)

According to Algorithm [3] we try to identify the common and different rules of two X3ML files. More precisely, the algorithm takes as input the X3ML mapping and source and target schema. It creates mapping rules regarding the Algorithm [1] and returns the Delta Explicit and common rules.

**Algorithm 3** Main procedure for Graphical Comparison

1: **Input:** \( < S_1, M_1, T_1 >, < S_2, M_2, T_2 > \)
2: **Output:** \( \Delta e(G_1 \rightarrow G_2), Com(G_1, G_2) \)
3: 4: \( G_1, G_2, \Delta e(G_1 \rightarrow G_2), Com(G_1, G_2) \leftarrow \emptyset \)
5: **for** each \( < S_n, M_n, T_n > \) in Input **do**
6: \( G_n \leftarrow \text{CreateMappingRules}( < S_n, M_n, T_n > ) \) \hspace{1cm} \( \triangleright \) Algorithm [1]
7: **end for**
8: \( \Delta e(G_1 \rightarrow G_2) = \{ \text{Add}(R) | R \in G_2 - G_1 \} \cup \{ \text{Del}(R) | R \in G_1 - G_2 \} \)
9: \( Com(G_1, G_2) = \{ G_1 + G_2 \} - \Delta e(G_1 \rightarrow G_2) \)
10: **return** \( \Delta e(G_1 \rightarrow G_2), Com(G_1, G_2) \)

System provides the following capabilities:

1. **Source schema aggregation.** The system undertakes to create a graph, combining two source schemas of mappings.

2. **Mapping rules comparison.** As regards the mapping rules, we create a 3D graph creating correlations between source and target schemas. Also, we draw
classes with different shape in order to compare two mappings graphically (Figure 5.35 and 5.36).

Finally, this functionality includes all capabilities of Mapping-centered representation.

### 4.5.5.3 Instance Comparison

The functionality of Instance Comparison concludes the proposed comparisons of mappings. It constitutes a useful method to identify changes according to the generated results. The system compares generated instances between different mappings offering two graphs which display the same concept (Figure 5.38). The experts are able to find several variations in order to modify the mappings.

System provides the following capabilities:

1. **Search Instances.** The users are able to find generated instances of both mappings by searching direct URIs or their classes.

2. **Related resources and properties.** Two interactive graphs are available to depict parallel instances. They present all the related resources (instances and classes), incoming and outgoing properties.

3. **Exploration.** Also, the graphs offer exploration techniques to provide navigation between the generated instances.

4. **Comparison.** The parallel projection of two graphs allows users to find differences and compare generated instances between different mappings.

### 4.5.6 More Functionalities

The last subsection contains extra functionalities to facilitate the mapping process. We suggest two useful capabilities to users in order to cover several rising issues, such as the representation of generated instances and the evolution of the mapping over time.

#### 4.5.6.1 Instance Representation

Despite the analysis of mappings during the process, the system allows the opportunity of exploring the generated instances after the execution of the mapping. As already mentioned, the result of X3ML mappings are linked data produced by the execution of X3ML engine. The result may be in the form of RDF/S or triples. Although this format is a standard form, it is very difficult to be read and identify the generated instances by users. Thus, this functionality provides the capability of illustrating a graph and exploring all instances and their properties (Figure 5.30).

System provides the following capabilities:
CHAPTER 4. PROPOSED APPROACH

1. **Search Instances.** The user is able to search generated instances based on two criteria. The first allows direct search in URIs, while the second search instances of specific classes. Since the user has found the instance, the system creates a graph based on it.

2. **Related resources.** Also, the graph displays all the related resources (classes and other instances) regarding the focused instance. We can see their labels in short, while if we focus, the labels depict the URIs.

3. **Properties.** A necessary need is to display the properties of related resources. The graph connects the above-mentioned resources with incoming and outgoing properties displaying both their labels and URIs.

4. **Exploration.** Finally, the interactive graph allows the capability of exploration instances. The user can select a related instance and generate its structure.

4.5.6.2 Tracking Changes Over Time

During the time, there are various versions regarding the same mapping process. The versioning is based on the current scope and purpose of the mapping. Therefore, the system provides the user with the capability of monitoring the evolution of a mapping over time regarding the transformation rules and the evolution of the source and target schema.

System provides the following capabilities:

1. **Info about versions.** In order to understand the scopes of versioning, we identify the available versions and provide detailed information about them and their purpose (Figure 5.31).

2. **Track changes between two versions.** The system tracks the changes about how an actor translates the source schema to the target schema during a time period. It offers a side-by-side projection of two versions (Figure 5.32) based on the textual comparison which was proposed in previous subsection (§ 4.5.5.1).
Chapter 5

Implementation

This chapter discusses the implementation of the system. Section § 5.1 introduces the developed software, including a detailed description of its features. Section § 5.2 provides the analysis of system’s architecture, while it presents technical information and the composition of background environments. Section § 5.3 concludes this chapter by describing in detail the user’s manual of our system.

5.1 Description

Since we proposed a complete analysis of functionalities, we have developed a system that works as intermediary between expert users and mappings of X3ML. We introduce Mapping Analyzer (Maze), a web-based tool which undertakes to serve Source, Target and IT experts in order to provide a complete management of mappings.

The Mapping Analyzer constitutes an integrated system which has been developed with respect to the following software design principles.

(a) Organizational structure. An important feature of software is the orchestration of background components. All parts are in a structured form, creating an effective collaboration between front-end and back-end environment.

(b) Single responsibility principle. Each component handles only one responsibility regarding the functionalities. On future if we need to make one change we are going to make it in the component which handles it.

(c) Open/closed principle. This principle states “software entities should be open for extension, but closed for modification” [39] that is, such an entity can allow its behavior to be extended without modifying its source code.

(d) Dependency Inversion Principle. It refers to a specific form of decoupling software modules. The conventional dependency relationships established from high-level, policy-setting modules to low-level, dependency modules are reversed.
(e) Simplicity. Last but not least, simplicity and clarity should be achieved by tools in order to make clear meanings and functions.

Finally, Maze has been originally implemented and integrated in the context of Mapping Memory Manager (3M) and 3M Editor. The actors are able to use this tool through either the “Mapping Analysis” or “Compare Mappings” tab. Special feature is the direct access to all functionalities in real time during the mapping process. This implementation is maintained by the community on GIT repository\(^1\).

## 5.2 Technical Information and Architecture

The implementation has been developed as Maven web application project. It has been written mainly in Java and JavaScript, while we have used other web development technologies such as HTML5 [11], CSS3 [27] and jQuery [2] (the implementation contains about 25000 lines of source code). Maze consists of two main environments (Server and Client), while there are external components which contribute to several functionalities. Figure 5.1 depicts the Maze global architecture.

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{maze-global-architecture.png}
\caption{The Maze Global Architecture}
\end{figure}

**Server.** The server-side part is a Java application. It has been developed in order to provide a complete API of all provided functionalities. It was designed as a REST based-Web Service for X3ML mappings using the Java API for Restful Service (JAX-RS API) with Jersey that exposes multiple resources.

\(^{1}\)GitHub repository: https://github.com/isl/Maze
5.2. TECHNICAL INFORMATION AND ARCHITECTURE

**Client.** The client-side environment consumes the server’s services as XML or JSON data and provides representations using all the latest Web features like HTML5, CSS3, Ajax and JQuery that all modern browsers support. Finally, it uses multiple web libraries for visualizing purposes.

**External Components.** The Mapping Memory Manager (3M) and 3M Editor work as intermediary between users and Maze. Finally, several services from 3M Editor constitutes important components, providing X3ML, source, target and other necessary files in order to analyze them.

Maze components are further explained in more detail in the next subsections.

5.2.1 Server-side Environment

Considering the backend of our tool, we have designed an effective system which undertakes to analyze mappings and provide generated results as services. It is written in Java for several reasons. First, this is the language we master the best. Second, Java is world-wide used and possesses a large community of developer which facilitate the access to information and code examples. Finally, the main libraries that are used in this implementation (Jena, Jersey, etc.) were existing in Java.

The server-side environment core consists of six major components: Maze Rest API, Controller, Provided Functionalities, Data Generators, Utilities and Services Consumers. In Figure 5.2 the overall structure of server is depicted.

The Maze Rest API service is responsible for exchanging data between the server and services consumers. The REST based API is provided in order to serve clients for X3ML mappings. It consumes and produces XML or JSON format depending on the requested data forms.

The Controller is the orchestration component of the server’s architecture. It is aware of any operation that takes place in the backend. In more details, the Controller is responsible for making decisions for every process workflow regarding requested operations, and controlling the collaboration of Maze subcomponents.

The Provided Functionalities implements the main processes regarding each request. They consists of five packages, containing data structures, functions, etc. These packages regard source and target reasoners, coverage metrics, instance and mapping rules analyzer.

The role of Data Generators is twofold: (a) they are responsible for conducting the implemented functions, ordering the involved executions and (b) they generate data in order to be produced by services.

The Utilities facilitate several functionalities, including general procedures. They are an intermediate component which receives documents from 3M Editor, load ontology models to memory and manipulates X3ML files (i.e. marshaling, unmarshaling XML files). Moreover, they formulate dynamic SparQL queries in order to serve particular operations.
The Services Consumers constitutes an external component which consumes server’s results. Although our web application plays the role of consumer, there is the capability of retrieving data from other external clients.

Besides these basic components, the server use a configuration file, provided by a java file that contains all the information required by the tool (config.properties). Concluding, we support log files for better monitoring of tool which are also accessible by service.

![Figure 5.2: Server-side Environment](image)

5.2.2 Client-side Environment

The client-side environment constitutes the user interface of our system. It has been implemented based on modern design patterns, following the basic user interface design principles. As regards development information, it is written in JavaScript and jQuery, using the latest web technologies such as HTML5, CSS3, Ajax requests, etc. Finally, it has been designed in order to be adapted to mobile devices and to be supported by the majority of modern web browsers such as Google Chrome, Mozilla Firefox, Opera, etc.

As regards the client’s architecture (Figure 5.3), it consists of five major components: Controller, Single Mapping Analysis and Comparison of Mappings pages, Ajax Requests and Cache Memory manager. The Controller constitutes the administration component which manages internal processes. It is responsible for addressing users’ needs to the corresponded components and instructing the exchanged data. The Single Mapping Analysis page provides the majority of implemented functionalities. It regards a specific mapping, providing comprehensive view in order to visualize analyzed results such as coverage metrics, source and target schema graphs instance representation, etc. The Comparison of Mappings page takes as input two mappings and is responsible for depicting comparison func-
5.2. TECHNICAL INFORMATION AND ARCHITECTURE

The Ajax Requests works as intermediary between client and server, orchestrating data exchanges. It directs the data in order to send and receive the appropriate contents, making asynchronous requests to server’s API. The last main component is the Cache Memory manager which stores and retrieves temporary data in session storage of web browsers.

Furthermore, there are external complementary components. The 3M editor and Mapping Memory manager redirect users to our system, while they supply client with required documents and information.

For visualization purposes, we used multiple web-based libraries. We used Bootstrap [1] in order to make front-end web development faster and easier and offer mobile-friendly user interface. Moreover, we used wide-spread web libraries for representation purposes. The most important are: D3 [2], Threejs [3], Arborjs [4], Jointjs [5], Chartjs [6], Difflibjs [7], etc. Finally, configuration file is available which allow users to adjust client and gathers the several necessary information, such as location and port of server, url of Rest services, etc.

---

5.3 Maze User’s Manual

Installation

Since Maze has been developed as Maven project, it provides all required libraries in pom.xml. Also, the folder src contains all the files needed to build the web app and create a war file. Therefore, the installation requires only four steps:

1. **Download.** You can download this project from GitHub repository (https://github.com/isl/Maze).

2. **Build.** In order to compile the source files and package the application, open this project with an IDE (i.e. Netbeans) and build with dependencies.

3. **Deploy.** You may use any application server that supports war files (recommended: Apache Tomcat).

4. **Run.** In order to launch application, you only need to open the server’s URL followed by tool’s folder (e.g. http://localhost:8080/Maze/).

Moreover, if you need further configuration, you can edit the Controller which is located at Maze/src/main/webapp/app/js/Controller.js. For design or further development purposes, you can set Maze in DebugMode (true). Also, you may configure ServerURL (hosted url), port, Maze’s Rest services URL (url which hosts Maze’s server) and target schema service of 3M. Figure 5.4 illustrates the default configuration.

![Configuration File - Client](image)

Figure 5.4: Configuration File - Client

Finally, you may set services to retrieve resources (i.e. X3ML files, source and target schema, generated data records, etc.) for server. Figure 5.5 illustrates the default configuration of server. The file is located at Maze/src/main/webapp/WEB-INF/config.properties.

**Using Maze**

The main page is index.html that conducts users in order to select one of two main functionalities. Single Mapping Analysis tab contains a textbox in which you can give the unique mapping ID. By clicking Analyze you are transferred to
single mapping analysis. Comparison of Mappings allows you to give two different mapping IDs and you may click on Compare button in order to compare them.

Other important pages are about.html and errorpage.html. The first contains information about Maze such as license conditions, description, contact info, GitHub repository and other resources, while the second is appeared in cases when the mapping ID is incorrect or the requested page is not available.

Mapping Memory Manager (3M) and 3M Editor work as intermediary between users and Maze. You can access this project through Mapping Memory Manager community. More specifically, you can use Maze through:

1. **3M** – More option contains direct links to single mapping analysis (Analysis) or comparison of two mappings (Compare).

2. **3M Editor** – You may view analysis of your mapping by using the Analysis tab and clicking the “view Analysis” link.

3. **Direct link** – Finally, you may give directly the mapping ID as parameter in URL. For instance you add the following parameter “?id=Mapping” followed by your mapping ID.

**Maze Web Services**

Despite the graphical user interface, the Maze implementation has been developed in order to provide a complete API of all provided functionalities. It was designed as a REST based-Web Service for X3ML mappings using the Java API for Restful Service (JAX-RS API) with Jersey that exposes multiple resources. More precisely, you may consume data in XML or JSON format making simple requests.

Let assume that the implementation has been deployed locally. The complete API is described below:

```
URLConnection to retrieve XML from 3M service
ServiceXML=http://119.91.183.3/MEditor/ServiceXML

URLConnection to retrieve XML file from 3M service
ServiceSourceFile=http://119.91.183.3/MEditor/ServiceSourceFile

URLConnection to retrieve Target schema file from 3M service
ServiceTargetSchema=http://119.91.183.3/MEditor/ServiceTargetSchema

URLConnection to retrieve Data records (instances) file from service
ServiceDataRecords=http://119.91.183.3/MEditor/ServiceDataRecords

URLConnection to retrieve Versions of mapping
ServiceGetVersionList=http://119.91.183.3/MEditor/ServiceGetVersionList

URLConnection to retrieve XML mapping from version collection
ServiceVersionXMLPart=http://119.91.183.3/MEditor/ServiceVersionXMLPart
```

Figure 5.5: Configuration File - Server
<table>
<thead>
<tr>
<th>URI</th>
<th><a href="http://localhost:8080/Maze/webresources/x3ml/singlemapping/%7Bid%7D">http://localhost:8080/Maze/webresources/x3ml/singlemapping/{id}</a></th>
</tr>
</thead>
<tbody>
<tr>
<td>Method</td>
<td>GET</td>
</tr>
<tr>
<td>Produces</td>
<td>XML/JSON</td>
</tr>
<tr>
<td>Functionality</td>
<td>Returns basic X3ML structure depending on requested format</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>URI</th>
<th><a href="http://localhost:8080/Maze/webresources/x3ml/singlemapping/plain/%7Bid%7D">http://localhost:8080/Maze/webresources/x3ml/singlemapping/plain/{id}</a></th>
</tr>
</thead>
<tbody>
<tr>
<td>Method</td>
<td>GET</td>
</tr>
<tr>
<td>Produces</td>
<td>XML</td>
</tr>
<tr>
<td>Functionality</td>
<td>Returns X3ML as is, ordering <strong>mapping</strong> and <strong>link</strong> elements. It is used for textual comparison.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>URI</th>
<th><a href="http://localhost:8080/Maze/webresources/x3ml/singlemapping/coveragemetrics/%7Bid%7D">http://localhost:8080/Maze/webresources/x3ml/singlemapping/coveragemetrics/{id}</a></th>
</tr>
</thead>
<tbody>
<tr>
<td>Method</td>
<td>GET</td>
</tr>
<tr>
<td>Produces</td>
<td>XML/JSON</td>
</tr>
<tr>
<td>Functionality</td>
<td>Returns all coverage metrics of source and target schema of mapping.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>URI</th>
<th><a href="http://localhost:8080/Maze/webresources/x3ml/source_schema/er/%7Bid%7D">http://localhost:8080/Maze/webresources/x3ml/source_schema/er/{id}</a></th>
</tr>
</thead>
<tbody>
<tr>
<td>Method</td>
<td>GET</td>
</tr>
<tr>
<td>Produces</td>
<td>XML/JSON</td>
</tr>
<tr>
<td>Functionality</td>
<td>Return in structured format tables and their attributes.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>URI</th>
<th><a href="http://localhost:8080/Maze/webresources/x3ml/source_schema/tree/%7Bid%7D">http://localhost:8080/Maze/webresources/x3ml/source_schema/tree/{id}</a></th>
</tr>
</thead>
<tbody>
<tr>
<td>Method</td>
<td>GET</td>
</tr>
<tr>
<td>Produces</td>
<td>XML/JSON</td>
</tr>
<tr>
<td>Functionality</td>
<td>Returns source schema as tree graph including parent-child relations.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>URI</th>
<th><a href="http://localhost:8080/Maze/webresources/x3ml/target_schema/all/%7Bid%7D">http://localhost:8080/Maze/webresources/x3ml/target_schema/all/{id}</a></th>
</tr>
</thead>
<tbody>
<tr>
<td>Method</td>
<td>GET</td>
</tr>
<tr>
<td>Produces</td>
<td>XML/JSON</td>
</tr>
<tr>
<td>Functionality</td>
<td>Returns statistics, classes, subclasses, properties, subproperties and other related information of target schemas.</td>
</tr>
</tbody>
</table>
5.3. MAZE USER’S MANUAL

<table>
<thead>
<tr>
<th>URI</th>
<th><a href="http://localhost:8080/Maze/webresources/x3ml/mappingrules/er/%7Bid%7D">http://localhost:8080/Maze/webresources/x3ml/mappingrules/er/{id}</a></th>
</tr>
</thead>
<tbody>
<tr>
<td>Method</td>
<td>GET</td>
</tr>
<tr>
<td>Produces</td>
<td>XML/JSON</td>
</tr>
<tr>
<td>Functionality</td>
<td>Returns as structured format mapping rules between source (ER) and target schema (Mapping-centered representation).</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>URI</th>
<th><a href="http://localhost:8080/Maze/webresources/x3ml/mappingrules/tree/%7Bid%7D">http://localhost:8080/Maze/webresources/x3ml/mappingrules/tree/{id}</a></th>
</tr>
</thead>
<tbody>
<tr>
<td>Method</td>
<td>GET</td>
</tr>
<tr>
<td>Produces</td>
<td>XML/JSON</td>
</tr>
<tr>
<td>Functionality</td>
<td>Returns as structured format mapping rules between source (TREE) and target schema (Mapping-centered representation).</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>URI</th>
<th><a href="http://localhost:8080/Maze/webresources/x3ml/source_schema/coveredelements/%7Bid%7D">http://localhost:8080/Maze/webresources/x3ml/source_schema/coveredelements/{id}</a></th>
</tr>
</thead>
<tbody>
<tr>
<td>Method</td>
<td>GET</td>
</tr>
<tr>
<td>Produces</td>
<td>XML/JSON</td>
</tr>
<tr>
<td>Functionality</td>
<td>Returns all covered elements considering the mapping and source schema.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>URI</th>
<th><a href="http://localhost:8080/Maze/webresources/x3ml/target_schema/coveredelements/%7Bid%7D">http://localhost:8080/Maze/webresources/x3ml/target_schema/coveredelements/{id}</a></th>
</tr>
</thead>
<tbody>
<tr>
<td>Method</td>
<td>GET</td>
</tr>
<tr>
<td>Produces</td>
<td>XML/JSON</td>
</tr>
<tr>
<td>Functionality</td>
<td>Returns all covered elements considering the mapping and target schema.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>URI</th>
<th><a href="http://localhost:8080/Maze/webresources/x3ml/source_schema/metrics/excludinglist/%7Bid%7D">http://localhost:8080/Maze/webresources/x3ml/source_schema/metrics/excludinglist/{id}</a></th>
</tr>
</thead>
<tbody>
<tr>
<td>Method</td>
<td>POST</td>
</tr>
<tr>
<td>Consumes</td>
<td>TEXT-PLAIN</td>
</tr>
<tr>
<td>Produces</td>
<td>XML/JSON</td>
</tr>
<tr>
<td>Functionality</td>
<td>Consumes plain text as JSON format the requested excluding elements of source schema and returns new percentage coverage metrics.</td>
</tr>
<tr>
<td>URI</td>
<td><a href="http://localhost:8080/Maze/webresources/x3ml/target_schema/metrics/excludinglist/%7Bid%7D">http://localhost:8080/Maze/webresources/x3ml/target_schema/metrics/excludinglist/{id}</a></td>
</tr>
<tr>
<td>-----</td>
<td>-------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Method</td>
<td>POST</td>
</tr>
<tr>
<td>Consumes</td>
<td>TEXT-PLAIN</td>
</tr>
<tr>
<td>Produces</td>
<td>XML/JSON</td>
</tr>
<tr>
<td>Functionality</td>
<td>Consumes plain text as JSON format the requested excluding classes or properties of target schema and returns new percentage coverage metrics.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>URI</th>
<th><a href="http://localhost:8080/Maze/webresources/x3ml/instances/%7Bid%7D">http://localhost:8080/Maze/webresources/x3ml/instances/{id}</a></th>
</tr>
</thead>
<tbody>
<tr>
<td>Method</td>
<td>GET</td>
</tr>
<tr>
<td>Produces</td>
<td>XML/JSON</td>
</tr>
<tr>
<td>Functionality</td>
<td>Returns all instances in structured format, including URI, label, incoming and outgoing properties and type of class.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>URI</th>
<th><a href="http://localhost:8080/Maze/webresources/x3ml/graphical_comparison/%7Bmappid1%7D/%7Bmappid2%7D">http://localhost:8080/Maze/webresources/x3ml/graphical_comparison/{mappid1}/{mappid2}</a></th>
</tr>
</thead>
<tbody>
<tr>
<td>Method</td>
<td>GET</td>
</tr>
<tr>
<td>Produces</td>
<td>XML/JSON</td>
</tr>
<tr>
<td>Functionality</td>
<td>Returns a list of common and different mapping assertions between two mappings. It contributes to graphical comparison.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>URI</th>
<th><a href="http://localhost:8080/Maze/webresources/x3ml/versions/%7Bid%7D">http://localhost:8080/Maze/webresources/x3ml/versions/{id}</a></th>
</tr>
</thead>
<tbody>
<tr>
<td>Method</td>
<td>GET</td>
</tr>
<tr>
<td>Produces</td>
<td>XML/JSON</td>
</tr>
<tr>
<td>Functionality</td>
<td>Returns a list of available versions and related information such as id, date, user and comments.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>URI</th>
<th><a href="http://localhost:8080/Maze/webresources/logs/get">http://localhost:8080/Maze/webresources/logs/get</a></th>
</tr>
</thead>
<tbody>
<tr>
<td>Method</td>
<td>GET</td>
</tr>
<tr>
<td>Produces</td>
<td>Maze-logfile.log File</td>
</tr>
<tr>
<td>Functionality</td>
<td>Downloads log file of Maze implementation.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>URI</th>
<th><a href="http://localhost:8080/Maze/webresources/logs/show">http://localhost:8080/Maze/webresources/logs/show</a></th>
</tr>
</thead>
<tbody>
<tr>
<td>Method</td>
<td>GET</td>
</tr>
<tr>
<td>Produces</td>
<td>TEXT-PLAIN</td>
</tr>
<tr>
<td>Functionality</td>
<td>Displays log file of Maze implementation.</td>
</tr>
</tbody>
</table>
5.3. Single Mapping Analysis

Since you have installed the implementation, the first major management panel is the analysis of single mapping. As already described, you only need to give as input a mapping ID and Maze’s backend retrieve and analyze the X3ML mapping file automatically. Figure 5.6 depicts the Home page of interface.

In order to be more precise, you may explore all provided pages through the left side bar, providing quick navigation among them. Also, each menu item contains submenus in order to categorize functionalities, while the right side of page is the main content of the page. Home page contains an overview of all functionalities and provides a small description about their capabilities.

- The Single Mapping menu item regards source and target schema analysis, warnings and errors.
- The Mapping Graphs contains representation of mappings (X3ML-centered and Mapping-centered).
- The Percentage Coverage item includes metrics and other information about coverages between involved schemas and mapping.
- Instance Representation illustrates graphs about the generated instances after the mapping process. This functionality requires the involvement of X3ML engine.
- Finally, the Evolution menu item allows users to track changes about several versions of selected mapping.

Source Schema

This functionality is used to explore the given source schema of mapping. It requires to an uploaded source schema through 3M Editor, which might be an example XML file or an XML schema file. Valid formats are, XML (a data file), XSD (a XML schema file). In cases which the source schema is ER, you only need to export the relational database as XML format.

Firstly, the implementation allows user to be informed about source schema details. It describes the name, type, version and info about collection (Figure 5.7).

Furthermore, some additional information (Figure 5.8) can be provided including sample data, for example source schema type, total parent and child elements.

An important capability is the declaration of undesired data for mapping. In most cases source schemas contains private data or data for internal use only. Thus, you can define them searching in parent or child elements according to Figure 5.9. By clicking "Save & Analyze", you may be informed if these elements have already mapped, while new source schema coverage metrics are available.
CHAPTER 5. IMPLEMENTATION

The most important functionality is the representation of source schema. Although the XML data is an understandable format for experts, there are complex structures which are difficult to be explored. The representation tries to create a simple graph in order to facilitate users. By default, the generated graph is a tree (XML) which allows the capability of search and focus on specific elements by clicking on them. Another representation regards ER schemas depicting tables and attributes. Although relations between tables are not specified in source schema, the graph tracks them automatically and connects them with edges. In cases where there are false derivations the user is able to change them (Figure 5.10 & 5.11).

Target Schema

Explore Target Schema page is used to explore and understand the involved target schemas of mapping. It contains general info, statistics, interactive graphs and other additional options such as exploration of classes and properties. The main
input file is in RDF/S format.

**Target Info** tab provides an overview of target schemas (Figure 5.12). The columns are: a) **Schema** describes the name of source schema, b) **Type** specifies the schema’s format, c) **Version** declares the current version, and d) **Schema File** contains direct link to initial schema’s format.

The Target Schema Statistics container provides information about the selected target schema when you click it from the previous list. Schema Type indicates the data format, while there are two number counting total classes and properties respectively. For example, CIDOC CRM is rdfs type and contains 85 classes and 286 properties (Figure 5.13).

Equivalent to source, you are capable to define undesired classes and properties in target schema. You may either search specific classes and properties or load default excluded classes for CIDOC CRM pressing the ‘Load Default for CIDOC-CRM’ button. Figure 5.14 presents this list. By clicking **Save & Analyze**, you may be informed if these classes or properties have already used in mapping, while
new target schema coverage metrics are available.

Two graphs illustrate relations for classes and properties (Figure 5.15 & 5.16). Both of them contain hierarchy relations. For example, each class is depicted as a node, while edges indicated its super-classes or sub-classes. Another characteristic is the size of each node. It depends on the total child concepts (sub-classes/sub-properties). Finally, as regards colors of classes are based on official color definitions according to CIDOC CRM.

The interactive behavior constitutes a major advantage. You are capable to focus on classes or properties, zoom in/out, select class or even search specific appellations. Figure 5.17 shows a focused class and its properties on mouse hover.

When you select a class/property, the implementation shows specific information about it. As regards classes, it shows label, URI, Comments and subclasses. Clicking a link of subclass, the graph is updated and focuses on this class. The "View Properties" button allows you to explore in detail the selected class (Fig-
Figure 5.15: Target Schema Classes

Figure 5.16: Target Schema Properties

Figure 5.17: Focused class and its subproperties

Figure 5.18. Below, in Figure 5.19 you can see incoming and outgoing properties of selected class. There attached labels and arrows which show the properties between classes.

Explore Target Schema functionality supports triangles of classes. This means that three classes are connected with three properties in a row, where the domain class of first property is the same with range of the third property. The implementation creates a list of triangles for each class, allowing you to draw each of them. An indicative example of triangle’s graph is depicted in Figure 5.20.
CHAPTER 5. IMPLEMENTATION

Figure 5.18: Details of focused class

Figure 5.19: Incoming and Outgoing Properties of Class

Figure 5.20: List and graph of triangle
5.3. MAZE USER’S MANUAL

Percentage Coverage of Source Schema

Since you have created a mapping, there are several metrics about coverage. This functionality permits to identify percentage coverages regarding source schema. Users should be informed about the coverage of the entire source schema or be sure that he has covered the elements with high interest. This functionality requires as input an XML file(source) and an X3ML mapping file.

The first graph contains three different metrics related to the source schema (Figure 5.21). The first metric displays the percentage coverage of parent elements, while the second counts only the leaves of the source schema that exist in the mapping. Finally, the third metric calculates the total coverage of the source schema.

Moreover, the system depicts the source schema as tree (XML form) highlighting the covered elements. Figure 5.22 draws an indicative example. Highlighted and red bordered elements are considered covered, while transparent elements are not mapped.

Percentage Coverage of Target Schema

The Target Analyzer allows users to explore the target schema, offering an effective schema analysis. It requires as input an RDF/S file(target) and an X3ML mapping file. Also, it offers metrics which are divided into two categories: (a) Direct and (b) Ancestors / Descendants.

At the category of Direct metrics each node is a separate element. Three different rates are provided, about classes, properties and resources. An element is considered to be covered only if it is mentioned in the mapping directly. On the other hand, the category of Ancestors / Descendants provides the same rates (classes, properties and resources), but it is differentiated on how the covered el-
CHAPTER 5. IMPLEMENTATION

Figure 5.22: Source Schema Covered Elements

Elements are calculated. An element is considered to be covered, if at least one ancestor or the element itself is referred in the mapping.

Figure 5.23: Target Schema Metrics

Figure 5.24 regards metrics about total target schemas, while Figure 5.25 depicts specific metrics for each involved target schema. You are capable to click each schema from the list and explore coverages about classes, properties and total average of selected schema.
X3ML-centered Representation

The implementation provides a graph with all mapping rules focusing on syntactic elements of the X3ML language. This particular representation depicts each mapping as a node.

As concerns the mapping nodes, the graph shows the sourceNode and targetNode which are connected with the main mapping. Finally, we are able to navigate through links which are linked to each mapping. Figure 5.26 illustrates an indicative example of this particular representation. Also, you may filter nodes through three buttons “TargetNodes”, “SourceNodes” and “Links”, while zoom is available to facilitate representation purposes.
Mapping-centered Representation

The Mapping-centered representation offers an interactive three-dimensional world that displays source, target and mapping assertions between them. You may explore scene making zoom in/out and select objects.

Firstly, you may explore source schema which is depicted at floor. Two representations are available depending on the type of schema (ER/XML). Figure 5.27 draws the initial representation of an ER source schema. Tables are depicted as big squares at floor of scene, while small cubes are their attributes. Each shape contains a label that describes their names. For example, the table “COIN” has attributes “COUNTRY_ID”, “FIND_DATE” etc. The representation of XML source schema is depicted as an interactive tree structure that is been expanded based on parent-children relations. Moreover, coverage information is available depending on objects format. An element is considered as covered if its structure is not-wireframe, while uncovered elements are depicted as wireframe object.

Each source schema object allows you to view mapping assertion regarding the target schema. By clicking on it, we create each related resource as sphere and connect them with properties. Below we selected the “COIN” table and it is created the “E22_Man-made_Object” sphere. This indicates that there is a mapping rule which translates each row of “COIN” table as this particular class. Also, you can see properties between classes based on target schema. The color is selected regarding the specific color-palette of CIDOC-CRM.

Another view of the same mapping is illustrated in Figure 5.29. The system draws a subgraph of target schema which includes involved classes regarding the mapping. Also, we are able to identify incoming and outgoing properties, creating edges between them.
5.3. MAZE USER’S MANUAL

Figure 5.27: Source schema 3D representation

**Instance Representation**

Despite mapping considerations, Instance Representation is a main functionality which depicts the results. You may explore generated instances when the mapping process is completed. It provides an interactive behavior which facilitates users to navigate and explore instances. At first, you are able to search instances either directly giving their URI or through their class. For example a generated instance of coin can be found by searching “http://www.oeaw.ac.at/COIN/627” or based on its class “http://www.cidoc-crm.org/cidoc-crm/E22_Man-Made_Object”.

The implementation finds automatically the requested instance and creates an interactive illustrating its related resources. More precisely, you may explore in incoming and outgoing properties between the focused instance and others nodes (i.e. instances, classes and literals) displaying both their labels and URIs. Finally, the graph allows you to navigate through nodes by clicking on their labels. Figure 5.30 depicts an indicative example of coin. The yellow-colored circle indicates its type; grey-colored triangles are related instances while green-colored squares are literals. Also, you can see a specific focused relation between instance of coin and its class, depicting their URIs.
Figure 5.28: Mapping assertions of source and target schema

Figure 5.29: Mapping assertions of source and target schema (2)
Figure 5.30: Instance Representation
CHAPTER 5. IMPLEMENTATION

Tracking changes over time

System provides the capability of monitoring the evolution of a mapping over time. You are able to identify all available versions of specific mapping and compare them. Firstly, a slider illustrates all versions based on date of creation while the current mapping is first in line. Also, it depicts all available information such as title, ID, date, user and comments. You may select two versions and track their changes by clicking Compare Versions button.

![Figure 5.31: Manage Versions of Mapping](image)

After the required analysis, system tracks the changes about selected versions. It offers a side-by-side projection based on the textual comparison of X3ML files. The yellow color indicates that the rows differ partly, while the red color shows that the mappings differ completely as regards this particular part. An indicative example is depicted in Figure 5.32.

![Figure 5.32: Comparison of versions](image)
5.3.2 Comparison of Mappings

The other major functionality is the comparison of mappings. It considers two mappings which are compared in the context of their initial form (textual), mapping assertions (graphical) and generated results (instances). You are able to give as input two mapping IDs and system undertakes to retrieve and compare them. The Home page (Figure 5.33) contains a sidebar which provides quick navigation through the available comparison functionalities. The right side of page constitutes the main content containing information about selected mappings and short description about each provided functionality.

![Figure 5.33: Home page of comparison](image)

**Textual Comparison**

Textual Comparison displays a side-by-side projection based on the X3ML language and its syntactic elements. In this view you are able to identify rows which differ partly or completely. The left side represents the first mapping, while the right side illustrates the second mapping. Each side contains the number of each row in order to help the comparison process.

Also, we mark with different colors the differences. The rows which remain the same have not background color (white). In contrast, differences are colored yellow or red background color depending how they differ. The yellow color indicates that the rows differ partly, while the red color shows that the mappings differ completely as regards this particular part. Figure 5.34 shows the Textual Comparison of two mappings where there is a partly difference.
CHAPTER 5. IMPLEMENTATION

Figure 5.34: Textual Comparison

Graphical Comparison

Graphical representation is a powerful technique to describe and understand data. This comparison tries to cover rising issues such as direct comparison of mapping assertions and depict those, combining source and target schemas.

As regards the representation, source schema is depicted at the flour of scene as tree. The structure is started at the center (root node), while nested elements are connected based on its structure. You can select covered elements and explore mappings according to target schema. Also, the source schemas of selected mappings are combined in order to create a common graph. Despite the source schema structure, users are capable to view covered elements directly. If an element is not covered by the mapping, it has transparent substance in scene.

The target schema is depicted at the upper level, displaying differences, related classes and properties. For instance, Figure 5.35 shows our running example, comparing two mapping about coins. Since you select a specific element, you may see related classes. The first mapping is presented as sphere, while the second as rhombus. The depicted connections show relations between schemas and properties about target schema. Green labels describe names of classes, while the orange labels are names of properties.

More precisely, below you can see more closely the mappings. The white rhombus indicates that first mapping describes coins as “E70_Thing”, while red sphere shows that the second approach maps coins as “E22_Man-made_Object”. Also, “E22_Man-made_Object” is connected with “E37_Material” through “P45_consists_of” property (Figure 5.36).

Instance Comparison

This functionality constitutes a useful method to identify changes according to the generated results. The system compares generated instances between different
mappings offering two graphs which display the same concept.

You are able to search instances either based on their URIs or their classes. Figure 47 draws two autocomplete text fields which search automatically concepts. **Search Instance** text field regards direct search on generated URIs, while **Search Instance of Class** regards URIs of available classes. By clicking the search button, we create a graph for selected instance.

Two interactive graphs are available to depict parallel instances. They present all the related resources (instances and classes), incoming and outgoing properties. Also, the graphs offer you exploration techniques for navigation purposes between the generated instances. Finally, you may identify differences between them through the side-by-side projection (Figure 5.38).
CHAPTER 5. IMPLEMENTATION

Figure 5.36: Comparison of specific concepts

Figure 5.37: Search of Instances

Figure 5.38: Side-by-side projection of Instance Comparison
Chapter 6

Conclusions and Future Work

Discussion

In this MSc thesis we analyzed and implemented an integrated system for mapping management, in order to provide a complete analysis for converting and publishing content as linked data. We focused on language X3ML; a mapping language designed by the Institute of Computer Science of FORTH, which is human readable, generic enough to cover most of the data models used nowadays and more explicit with the URI generating process. Although there is a plethora of mapping approaches and tools, the majority is based on mapping techniques and languages. To the best of our knowledge, this is the first work that proposed an extended mapping management which targets to optimize mappings, to improve their consistency and provide statistics related to source and target schemas.

More specifically, this work was based on X3ML mappings and tried to propose an advanced analysis and representation techniques of mappings, in order to contribute to more effective and correct way of data conversion. In the first place, we defined the major business use cases taking into account the main purposes of mapping process. In a nutshell, we suggested two business use cases, considering the process in which actors generate a mapping from scratch (Create a Mapping), and cases which actors have already implemented a mapping and try to improve it, check its quality, compare with others and monitor its evolution (Improve a Mapping). We defined three main actors who participate in mapping processes: (a) Source Schema Expert, (b) Target Schema Expert, and (c) IT Expert. Thereafter, we described the actors’ needs into clear, achievable, and verifiable high-level requirements.

Based on those requirements and system use cases, we proposed and designed eleven main functionalities, which aim to provide appropriate mechanisms for mapping management. They regard analysis of source and target schema, percentage coverage metrics, mapping representations, comparison of mappings, exploration of generated linked data and detection of changes through versions. In order to be more precise, we offered analysis of schemas, which gives exploratory capabilities to actors and allows them to understand schemas content. Percentage coverages

85
include two different metrics which constitute the opportunity given to the user to have direct overview of used data. Graphical representations were designed in order to focus on syntactic elements of X3ML language and illustrate correlations between target and source schemas. As far as concerns comparison of mappings, we implemented comparisons about the syntax of X3ML language, graphical representation of correlations and comparison of generated instances. Finally, the functionality of evolution tracking of mapping helped users to understand changes over time between versions.

We have concluded this thesis by presenting Mapping Analyzer (Maze); an integrated system for management of mappings, which undertakes to implement the aforementioned functionalities. It works as intermediary between actors and mappings, providing an adaptive user interface. All in all, Maze complements Mapping Memory Manager and 3M Editor, constituting an essential tool for mapping management.

**Future Work**

There are many directions that we are currently exploring or plan to work in the immediate future. First of all, an important issue deserving of further consideration is the evaluation in order to extract results from real users. We need to evaluate our system regarding usability assessed by testing the system’s functionality as concerns realistic use cases. Our main intention is to gather mapping experts and collect information based on system’s usability; how they were helped to create mappings from scratch and how the system contributes to improve them. Mapping Analyzer implementation, like any software, can accept many additional interventions to improve its functionality through evaluation results.

Moreover, an important future effort will be the consideration of extending system functionalities based on mappings identity. As already described, a mapping is identified by the combination of source, target, mapping assertions, time, application and person. Although the current proposed approach focused mainly on the first five parameters, application and profiling deserve of further scientific study. Analyzing more extensively the parameter of mapping’s application will also allow addressing more complex problems that are interesting from the expert’s point of view such as how complete are mappings, given a specific scope of usage. Furthermore, profiling of mapping experts would be an important achievement, contributing to more complex analysis and management of mappings.
Appendix A

A  X3ML Mapping

```xml
<x3ml source_type="xpath" version="1.0">
  <info>
    <namespaces>
      <mappings>
        <mapping>
          <domain>
            <source_node>//COIN</source_node>
            <target_node>
              <entity>
                <type>crm:E22_Man-Made_Object</type>
                <instance_info/>
                <instance_generator name="LocalTermURI">
                  <arg name="hierarchy" type="constant">COIN</arg>
                </instance_generator>
                <label_generator name="Literal">
                  <arg name="text">ID/text()</arg>
                </label_generator>
              </entity>
              <relationship>
                <type>crm:P52_has_current_owner</type>
              </relationship>
            </entity>
            <additional>
              <instance_info/>
              <instance_generator name="URIorUUID">
                <arg name="text" type="constant">http://www.oewa.ac.at/</arg>
              </instance_generator>
            </additional>
          </domain>
          <source_node>//COIN</source_node>
        </mapping>
      </mappings>
    </namespaces>
  </info>
</x3ml>
```
```xml
<target_node>
  <domain>
    <link>
      <path>
        <source_relation>
          <relation>ID</relation>
        </source_relation>
        <target_relation>
          <relationship>crm:P1_is_identified_by</relationship>
        </target_relation>
      </path>
      <range>
        <source_node>ID</source_node>
        <target_node>
          <entity>
            <type>crm:E41_Appellation</type>
          </entity>
        </target_node>
      </range>
    </link>
    <link>
      <path>
        <source_relation>
          <relation>country_id == country_id</relation>
        </source_relation>
        <target_relation>
          <relationship>crm:P30_has_current_keeper</relationship>
        </target_relation>
      </path>
      <range>
        <source_node>ID</source_node>
        <target_node>
          <entity>
            <type>crm:E40_Legal_Body</type>
            <instance_info>
              <constant>OEAW</constant>
            </instance_info>
            <instance_generator name="URIDUUID">
              <arg name="text" type="constant">
                http://www.oeaw.ac.at/
              </arg>
            </instance_generator>
          </entity>
        </target_node>
      </range>
    </link>
    <link>
      <path>
        <source_relation>
          <relation>country_id == country_id</relation>
        </source_relation>
        <target_relation>
          <relationship>crm:P2_has_type</relationship>
        </target_relation>
      </path>
      <range>
        <source_node>ID</source_node>
        <target_node>
          <entity>
            <type>crm:E55_Type</type>
            <instance_info>
              <constant>coin</constant>
            </instance_info>
            <instance_generator name="LocalTermURI">
              <arg name="hierarchy" type="constant">terms</arg>
              <arg name="term" type="constant">coin</arg>
            </instance_generator>
          </entity>
        </target_node>
      </range>
    </link>
  </domain>
</target_node>
```
<source_relation>
  <target_relation>
    <relationship>crm:P108i_was_produced_by</relationship>
    <entity_variable="p1">
      <type>crm:E12_Production</type>
    </entity>
    <relationship>crm:P10_falls_within</relationship>
  </target_relation>
</source_relation>

<range>
  <source_node>//COUNTRY</source_node>
  <target_node>
    <entity>
      <type>crm:E4_Period</type>
    </entity>
  </target_node>
</range>

<link>
  <path>
    <source_relation>
      <relation>AUTHORITY_ID == AUTH_ID</relation>
    </source_relation>
    <target_relation>
      <relationship>crm:P108i_was_produced_by</relationship>
      <entity_variable="p1">
        <type>crm:E12_Production</type>
      </entity>
      <relationship>crm:P9i_forms_part_of</relationship>
    </target_relation>
  </path>
  <range>
    <source_node>//AUTHORITY</source_node>
    <target_node>
      <entity>
        <type>crm:E4_Period</type>
      </entity>
    </target_node>
  </range>
</link>

<link>
  <path>
    <source_relation>
      <relation>WEIGHT</relation>
    </source_relation>
    <target_relation>
      <relationship>crm:P43_has_dimension</relationship>
      <entity_variable="d1">
        <type>crm:E54_Dimension</type>
        <additional>
          <relationship>crm:P91_has_unit</relationship>
          <entity>
            <type>crm:E58_Measurement_Unit</type>
          </entity>
        </additional>
      </entity>
    </target_relation>
  </path>
  <range>
    <source_node>//COUNTRY</source_node>
    <target_node>
      <entity>
        <type>crm:E4_Period</type>
      </entity>
    </target_node>
  </range>
</link>
Bibliography


[41] Nikos Minadakis, Yannis Markakis, Haridimos Kondylakis, Giorgos Flouris, Maria Theodoridou, Martin Doerr, and Gerald de Jong. X3ml framework: An effective suite for supporting data mappings.


