"A Model-Driven Engineering Approach for the Uniquely Identity Reconciliation of Heterogeneous Data Sources"

TESIS DOCTORAL

Autor
D. José González Enríquez

Directores
Doctora D.ª María José Escalona Cuaresma
Doctor D. Francisco José Domínguez Mayo

Sevilla, 22 de mayo de 2017

"A Model-Driven Engineering Approach for the Uniquely Identity Reconciliation of Heterogeneous Data Sources"

DOCTORAL THESIS

Author
D. José González Enríquez

Supervisors
Ph.D. Dª. María José Escalona Cuaresma
Ph.D. D. Francisco José Domínguez Mayo

Seville, 22th May, 2017

To my parents, Antonio y Mercedes,
for their tireless support, help and affection.

“My biggest motivation?
Just to keep challenging myself.”

– Richard Charles Nicholas Branson

“La vida sin esfuerzo
es una vida mediocre”

– Papa Francisco

ACKNOWLEDGEMENTS

After several years of hard work, it is time to write one of the parts that I consider to be the most important part of this doctoral thesis, the acknowledgements, because without the effort, support, advice and encouragement of all the people who I am willing to name, the desire to do this work, which today becomes a reality, would have not been possible.

First, thank my family, my parents Antonio and Mercedes, my brothers Antonio and Jesus and my nephews Marta and Alejandro and my sister in law Ana. Thank you for always being supporting me, giving me affection and making the distance not to be a problem for feeling you very close every day.

Thanks to my traveling companion on the road to life, Maria. The name of the solution proposed in this work, since it could not have turned out any other way, refers to yours and it is because you are, have been and will be, one of the most important persons of my life. Thank you for understanding my work, for suffering me and waiting for me all the time that I have spent away from home, for being the first shoulder where to find comfort and the reason to smile every day. You are the faithful reflection of effort and constancy. Today I have achieved one of my goals and I am sure that soon, you will do with your own as well, but the most important thing is that we will always achieve what we want together.

To my adoptive family, Paqui and Paco, Pablo and the Palacios grandfather. Thank you for making me feel one more of the family, for the advice and encouragement to never give up.

To all my friends especially Andrew. Thank you for always accompanying me in good and bad times and for the moments of talk that made me disconnect from the routine of daily work. We still have many projects to do together, thank you for becoming in one more brother.

To my directors of this work of doctoral thesis. María José, thank you very much for having trusted me from the beginning and having initiated myself into this wonderful world of research. You have taught me to put a good face on the bad news and not to limit my dreams. Franci, apart from being my tutor, you have become a great friend for me. Thank you very much for your dedication, for your time, for the advice, for the moments of talk, for the discussions, for accompanying me the first time that I presented an article in an international congress, in short, for being always been for everything. Without your work, this doctoral thesis would have been only one more idea.

To all my colleagues of the “Ingeniería Web y Testing Temprano” research group for suffering me daily. Thank you very much Julian for all the conversations you had during my stays abroad, on topics related to this work and those which are not, you are an example of overcoming, do not be bored to raise awareness. To Juanmi and Antonio for having always been willing to manage anything while I have been making my research stays and all other companions for your support and conversation moments. To Virginia and Leticia for enduring and suffering me and giving me their affection every moment.

During the development of this doctoral thesis, I have been lucky to carry out several research stays, both national and international. Many thanks to the colleagues of the Research Group on Software Engineering at the University of Oviedo, especially Javier Tuya for the advice and guidelines, Raquel Blanco for the dedication, effort, work and help that she has given me. Also, thanks to Jesús Morán for the moments of talk in the day to day work. To Margaret Ross and Geoff Staples of "Solent University of Southampton". Thanks for the effort in showing me the most important sites of the city and for all the reviews and advice on the works that came true, you are endearing. To Susan Giesecke of the University of California Berkeley for her dedication and always ready to help in whatever I needed.

Thanks to Fujitsu laboratories in Europe (FLE) for being my sponsors. Thanks to your help, I could finance part of these studies by publishing articles, attending congresses and conducting research stays at national and international levels. Without your support, these activities would have been much more difficult to accomplish.

Thanks, God for giving me health, for filling my spirit of joy in moments of solitude and giving answers to questions that do not have.

I may have forgotten to mention many others who contributed directly or indirectly to the development of this work. From here, my sincere thanks to all of them.

To all, a thousand thanks, this work could hardly have been possible without your support.

José González Enríquez (Pepe)

INDEX

TABLES

FIGURES

CHAPTER I
INTRODUCTION

CHAPTER I. INTRODUCTION

Information is a phenomenon that provides meaning or sense to things. In general terms, information is an organized set of processed data, which constitutes a message about a concrete entity or phenomenon. The data are perceived, integrated and generate the necessary information to produce the knowledge that is the one that finally allows to make decisions to carry out the daily actions.

Information also processes and generates human knowledge. When a concrete problem needs to be solved or a decision must be taken, people usually use different sources of information and build what is generally called knowledge or organized information that allows problem solving or decision making.

The main topic of this doctoral thesis is related to the information management. Specifically, this first chapter aims to describes the context of the work developed in this doctoral thesis. To this end, the first section presents a brief introduction to focus work. Then, the second and third sections, describe what the structure of this document and brief conclusions of the chapter respectively.

1. INTRODUCTION

Currently, the information management is critical in many aspects of our lives. However, the incorporation of information and communications technology (ICT) in everyday life causes people to experience an overshooting of information, also known by the term “infoxication”. This term refers to the difficulty that someone has to understand a problem and make decisions about it because of an excess of information presence (Yang et al., 2003).

In the first era of ICT, the main problem that researchers had was how to find information and how to store and manage it efficiently. Currently, due to the presence of a lot of systems that store and generate information, the biggest problem that researchers have is how to extract knowledge of this information based on the needs of each in an effective way (Enríquez et al., 2015).

The complexity of information has increased not only by the high capacity of information production, but these huge amounts are distributed in multiple databases, not just one, and these databases are different, they do not have the same structure, someone’s repeat information and it is not always possible to make a faithful comparison between them, in conclusion, it can be said that these data sources are heterogeneous, it means, although the store information related to the same topic, they share neither structure nor content. Then, it would be very useful to integrate all information related to the same subject into a single data source. It does not means creating a new one unifying all the existing ones, it means that it is necessary to make a system that respond efficiently to queries that are made, despite their differences providing a consolidate information from all the data sources queried and it is very important to make it quickly. In this context, the problem of reconciling entities in heterogeneous data sources mentioned before (which is the start point of this doctoral thesis) takes a very important value.

Entity reconciliation (also called entity resolution or ER) is a fundamental problem in data integration and it is no new. It refers to combining data from different sources for a unified vision or, in other words, identifying entities from the digital world that refers to the same real-world entity. It is an uncertain process because the decision to allocate a set of records with the same entity, cannot be taken with certainty, unless these records are identical in all their attributes or they have a common key (Getoor and Machanavajjhala, 2012; Wang et al., 2013). This problem can be applied to many kinds of scenarios. Entity reconciliation is a well-known problem and it has been investigated since the birth of relational databases (Whang and Garcia-Molina, 2014). If to everything mentioned, a very trending topic nowadays such as the Big Data is added, this problem receives a much more significant attention due to the new challenges that it arises.

Although this problem is not new, the management of heterogeneous or big and heterogeneous data sources presents new challenges. (Enríquez et al., 2015; Gal, 2014). In the paper written by Getoor and Machanavajjhala, (2013), the authors exposed some of the main challenges of entity reconciliation in this environment such as: data heterogeneity, it is becoming more common that data are unstructured, unclean or incomplete and also there are diverse data types; data more linked, where it is expressed the necessity of inferring relationships; multi-relational data, dealing with the structure of entities; and building multi-domain systems, trying to customize methods that span across domains. In this sense this doctoral thesis tries to address the most of these problems providing a solution that covers these aspects.

In the literature, it is possible to find a wide variety of approaches to try to solve the problem of reconciliation of entities. Taking into account the classification presented by Gal (2014), entity reconciliation problems can be briefly classified into: deterministic rule-based, probabilistic-based methods, learning based techniques and graph-based techniques.

- **Deterministic rule-based**: Lee et al., (2013), proposed an approach to co-reference resolution that combines the global information and precise features of machine-learning models with deterministic, rule-based systems. Galhardas et al., (2001), presented a language, an execution model and algorithms that enable users to express data cleaning specifications declaratively using for their demonstration an example of a set of bibliographic references. Bhattacharya and Getoor, (2005), proposed a probabilistic model for collective entity resolution for relational domains where references are connected to each other. Thus, they presented an algorithm for collective entity resolution which is unsupervised and takes entity relations into account.
- **Probabilistic methods**: Winkler, (2002), presented methods for Record Linkage and Bayesian Networks. Verykios et al., (2003), presented a Bayesian decision model for cost optimal record matching using the ratio of the prior odds of a match along with appropriate values of thresholds to partition the decision space into three decision areas. This model is an improved version of the one proposed by Fellegi and Sunter, (1969).
- **Learning-based techniques**: Sarawagi and Bhamidipaty, (2002) presented a system that use a method of interactively discovering challenging training pair using active learning. Cohen and Richman, (2002), described techniques for

clustering and matching identifier names that are both scalable and adaptive, in the sense that they can be trained to obtain better performance in a concrete domain.

- **Graph-based techniques**: Ioannou et al., (2010), described a framework for entity linkage with uncertainty where the possible linkages are stored alongside the data with their belief value. They use a probabilistic query answering technique to take the probabilistic linkage into consideration. Wang et al., (2013), focused on the construction of effective reference table by relying on co-occurring relationship between tokens to identify suitable entity names. The firstly model data set as graph, and then cluster the vertices in the graph. They also mine synonyms and get the expansive reference table. Wang et al., (2016), modeled the entity resolution problem as the partition of the vertices in a weighted graph into cohesive subgraphs. They propose an approximate algorithm with approximation ratio bound is proposed and a heuristic algorithm for performing entity resolution on a large data set efficiently.

Let’s get a simple example of entity reconciliation found in the literature (McCallum et al., 2000) based on a coauthor network from bibliographic data used for InfoVis 2004.

At the left side of the Figure I-1, it is possible to see the complete network, but it is possible to note that, there are elements that represents the same author but they are not named equally.

In this sense and after applying an entity reconciliation process, it is possible to see how the final result (right side of Figure I-1), presents a much clearer vision of the complete coauthor network. In this process the references to concrete entities have been canonized and linked.

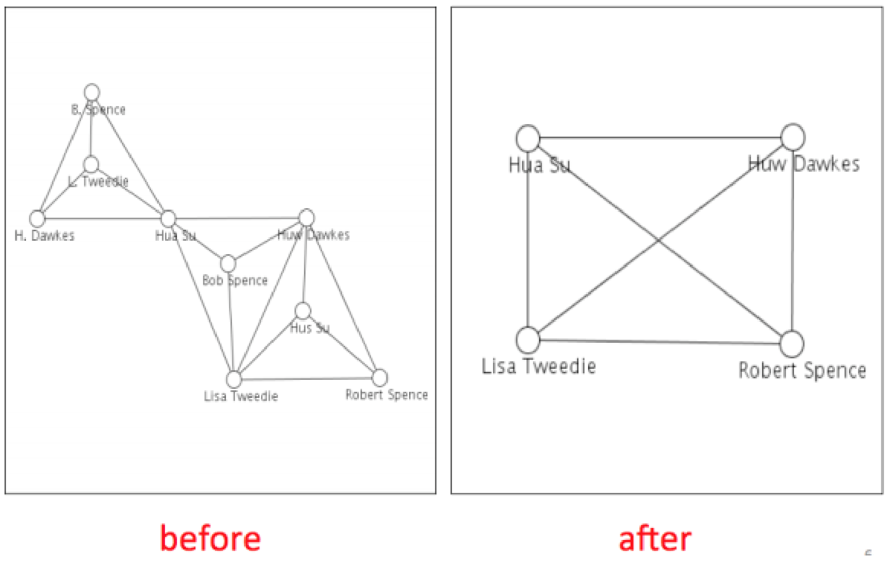


Figure I-1. Entity Reconciliation example (McCallum et al., 2000).

2. STRUCTURE OF THE DOCTORAL THESIS

After this introduction chapter, it will be presented the different chapters that compose this doctoral thesis. All these aspects are developed along them are structured as follows:

Chapter II offers a systematic study of the state of the art about which are the related works of this thesis. In it, techniques, methods or tools that allow to solve the entity reconciliation problem are analyzed. This general vision will allow to analyze how the entity reconciliation problem is being and have been solved.

The general view presented in the Chapter II helps to understand the actual situation and to set the basis so that during the Chapter III, the definition of the problem to be addressed could be performed. Chapter III provides, so, the definition of the problem, the challenges and objectives to meet, the working environment and a description of the approach to the problem dealt with in this thesis. This chapter also defines in detail the influences that have driven the realization of this work.

Raised the problem and achieve the objectives, the next three chapters describe proposed **M**odel-driven entity **R**econcil**IA**tion (MaRIA) Framework, deepening on each of the elements on which the framework is based on. In this sense, Chapter IV presents the MaRIA process, which is a set of activities that should be added to any software developing methodology to carry out an entity reconciliation process. This set of processes will be focused in the requirement, analysis and testing phases. The chapter, includes a concrete example of its use in a web software methodology. Chapter V presents the metamodels used to allow the user of the framework to model entity reconciliation problems and Chapter VI presents how the Early Testing has been included to the solution. In this chapter, the theoretical transformations that will allow to automatically generate the business rules that will make the model testable are presented. These business rules will derive in the test requirements of the application that consume data that generated after the entity reconciliation process.

In order to materialize and automate the metamodels, constraints and transformations proposed in Chapters V and VI, a support tool has been developed and it is presented in Chapter VII.

Next, Chapter VIII presents a case study taken as validation scenario for this doctoral thesis. The proposal presented has been applied to this scenario using the support tool presented in Chapter VII.

Finally, chapter IX closes the memory of this doctoral thesis presenting a set of conclusion, the contributions that this research contributes to the scientific community, presenting the future work that has been proposed as well as the new complementary research lines that have been created from this work.

This doctoral thesis is completed with five more sections: references and four annexes.

References section represents the description of all the literature references that the student has consulted during the development of this doctoral thesis.

Annex A describes an introduction to DSL-Tools (IDE which the support tool of this approach has been developed), explaining the software requirements needed for the installation, a short installation manual and a brief description of the IDE.

Annex B describes the user manual of the support tool of this proposal presented in Chapter VII.

Annex C presents the glossary of terms that will make easy to understand all the relevant concepts of this doctoral thesis.

Finally, Annex D summarizes the recognitions and research activity of the PhD student achieved during the development of this work.

3. CONCLUSIONS

The Doctoral Thesis presented in this work is motivated by a problem that has been identified within organizations engaged in the business of software: the need to establish systematic and automated mechanisms that allow to perform the entity reconciliation in heterogeneous data sources. All this to help organizations to improve the management and quality of data they work with.

The introduction chapter, in addition to frame and contextualize the work of the thesis, provides a definition of entity reconciliation, as understood throughout this work. Furthermore, it globally shows what the structure of presentation is and the research results which have resulted in the conduct of this doctoral thesis.

At this point, it is convenient to mention that this research work has been carried out within the “Ingeniería Web y Testing Temprano” (IWT2) research group of the “Escuela Técnica Superior de Ingeniería Informática (ETSII)” of the University of Seville. This research group is referenced in the Andalusian research plan as PAIDI TIC021.

Moreover, and more specifically, this thesis has been developed within a strategic framework of IWT2 group that aims to explore and investigate how to combine satisfactorily the Model-Driven Engineering (MDE) paradigm with the entity reconciliation problem.

Finally, this Doctoral Thesis has been sponsored by Fujitsu Laboratories of Europe (FLE). Fujitsu Laboratories Limited has had an active presence in Europe since 1990, when they set up their first facility at Stockley Park near Heathrow. In subsequent years, Fujitsu Laboratories opened branches in France and Germany, undertaking advanced research in the fields of telecommunications, information technology and computational science. In 2015, FLE established a new Data Analytics Research Center in Madrid, Spain, reflecting their commitment to “think global, act local” - applying their R&D expertise to address specific regional challenges.

CHAPTER II
RELATED WORK

CHAPTER II. RELATED WORK

The first natural step to propose an original contribution, is to study the existing knowledge regarding proposals for the ER process. In this sense, this chapter aims to study those proposals based on: tools, techniques or algorithms proposed for the ER.

For obtaining the actual state of the art in terms of tools, techniques or algorithms proposed for the ER process, a Systematic Mapping Study has been carried out trying to look for studies located between 2010 to 2017. In the SMS, the related papers found in terms of systematic mapping studies, systematic literature reviews or surveys have been described. Then the results obtained after the application of the proposed method and an analysis of them have been performed. Finally, this chapter concludes summarizing the most relevant state of the art presented along the same aspects.

1. RELATED WORK

ER is a topic that has been discussed and studied for many years. In this section, some related works such as systematic literature reviews, surveys or comparisons are presented.

Maddodi et al., (2010) discuss different strategies of deduplication with their pros and cons and some methods to prevent duplication in databases. This paper discusses seven techniques for detecting duplicate data (deduplication using correlated subquery, using temporary table, using derived table, by creating new tables and renaming it, using common table expression and using merge statements) and three preventive methods for SQL (the primary key, the unique key and the IGNORE_DUP_KEY constraint). Finally, the authors make a performance evaluation with Microsoft SQL-Server 2008 in different Data Warehouses.

Dorneles et al., (2011) divided “approximate data matching” into two basic groups: (i) those which compare data based on data values; and (ii) those which compare data based on their structure, exploiting and extracting relevant data to the comparison. They review both categories identifying different approaches and they present a comparative analysis. The authors only focus on work that relies on a similarity function when executing the data matching process. Costa et al., (2011) present an overview of research on data deduplication with the aim to provide a general assessment of useful references and ideas on this topic. Firstly, the authors describe the problem and after that, they propose two categories of techniques for deduplication: supervised (relational data, multidimensional data, data-mining/data-results, linked and XML data and streaming data approaches) and unsupervised (based on clustering, (dis)similarity-search in metric spaces and locality-sensitive hashing).

Yumusak et al., (2014), present a brief survey dealing with linked data ranking, classifying methods in: ontology ranking, RDF (Resource Description Framework) document ranking, graph ranking, entity ranking and document/source ranking.

Gaikwad and Bogiri (2015), present a survey analysis on duplicated detection in the domain of hierarchical data. They have oriented the paper to experts who are doing research in duplicate detection in xml data or hierarchical data.

Brizan and Tansel (2015), divide techniques for performing ER or record linkage into: (i) establishing good match criteria between any pair of tuples, and (ii) applying these criteria to one or more relations and they described both. Otero-Cerdeira et al., (2015), present a literature review regarding ontology matching, with the purpose of helping in guiding new practitioners to get an idea on the state of the field and determines possible research lines based on the decade of 2005 to 2015 and classifying the papers following the framework they proposed.

Beheshti et al., (2016), present a systematic review and a comparative analysis of Cross-document Coreference Resolution methods and tools (CDCR). The authors present a systematic review of the state-of-the-art of challenges and solutions to CDCR, a taxonomy of CDCR and an identification of a set of quality attributes approaches. Papadakis et al., (2016), propose a comparative analysis of approximate blocking techniques for ER presenting 17 state-of-the-art blocking methods, 6 popular real datasets and 7 established synthetic datasets that range from 10,000 to 2 million entities.

A comparison of previous approaches in terms of pros and cons is described in Table II-1 to clarify the contribution that this chapter proposes.

Reference	**Number of Databases**	**Apply a Specific Methodology**	**Classification Framework**	**Systematic Process**	**General Scope**	**Specific Scope**
Maddodi et al., (2010)	Unknown	X	X	X	X	✓
Dorneles et al., (2011)	Unknown	X	✓	X	X	✓
Costa et al., (2011)	Unknown	X	✓	X	X	✓
Yumusak et al., (2014)	Unknown	X	✓	X	X	✓
Gaikwad and Bogiri (2015)	Unknown	X	✓	X	X	✓
Brizan and Tansel, (2015)	Unknown	X	✓	X	X	✓
Otero-Cerdeira et al., (2015)	5	✓	✓	✓	X	✓
Beheshti et al., (2016)	4	✓	✓	✓	X	✓
Papadakis et al., (2016)	Unknown	X	✓	X	X	✓
Our proposal	11	✓	✓	✓	✓	X

Table II-1. Comparison of previous approaches

As reflected in Table II-1, the two found closest papers to our research are those presented by Beheshti et al., (2016) and Papadakis et al., (2016), respectively. Both use a specific methodology, define the number of databases that were consulted and apply a systematic process. However, their scope is specific, that is to say, a particular topic for ER, but not for ER in general. Although most of the papers propose a classification framework, they neither perform a systematic process nor specify the number of databases consulted. Finally, it is important to note that the number of databases consulted in the study that this chapter presents is more than the double proposed in the other papers.

2. METHOD

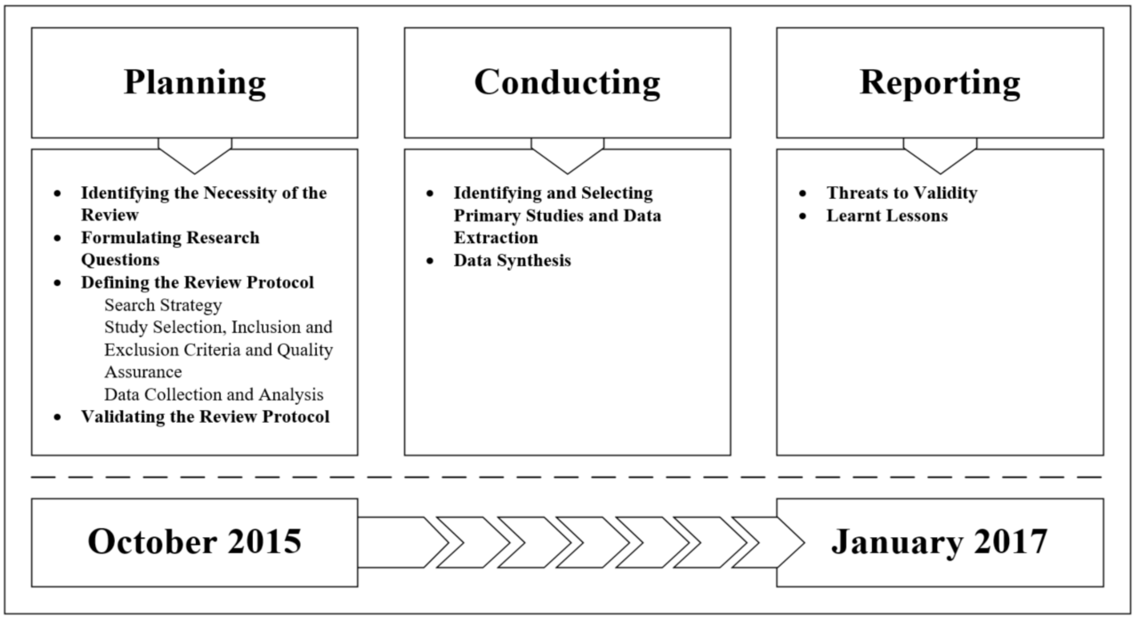
To carry out this study, the model of Systematic Mapping Study (SMS) has been taken as reference. It is a form of Systematic Literature Review (SLR) proposed by Kitchenham (Kitchenham and Charters, 2007) that aims to identify and categorize the available research on a broad software engineering topic (Kitchenham et al., 2011).

SMSs (Genero et al., 2014) are secondary studies with a broader scope than SLRs because the aim is to provide an overview of an interesting topic and identify the number and type of research and the available results about. This allows identifying subjects where empirical evidence is lacking and performing more empirical studies is needed. It is very common in SMSs to calculate the frequency of publications over time to identify trends or classify the found items in a default classification scheme. SMSs typically consume less time than SLRs and are useful for researchers as a basis to do further work with high level of rigor.

Considering that a SMS is a form of SLR, the last model of SLR proposed by Kitchenham and Brereton, (2013), had also been taken as reference. It establishes that a review should be composed of three phases: planning, conducting and reporting.

- Planning the review: Prior to a SLR, it is necessary to confirm the necessity of the research. The most important activity is defining the research questions that will define the review protocol.
- Conducting the review: It deals with executing the protocol that has been defined.
- Reporting the review: It describes how the final report has been elaborated.

Figure II-1 shows the different phases with all the activities that compose each of them and the time that has been invested to do so are shown. Details of the complete method step by step can be found below.



Planning
Conducting
Reporting
Identifying the Necessity of the Review
Formulating Research Questions
Defining the Review Protocol
Search Strategy
Study Selection, Inclusion and Exclusion Criteria and Quality Assurance
Data Collection and Analysis
Validating the Review Protocol
Identifying and Selecting Primary Studies and Data Extraction
Data Synthesis
Threats to Validity
Learnt Lessons
October 2015
January 2017

Figure II-1. Method for the Systematic Mapping Study

2.1. PLANNING

In this section, each of the tasks that have been made during the process the planning the SMS are presented. These are: identifying the necessity of the review, formulating research questions, defining the review protocol and validating the review protocol.

IDENTIFYING THE NECESSITY OF THE REVIEW.

ER is not a new necessity. Such a necessity aroused since databases started to be used. Extracting data from a same identity or integrating them into different databases are not simple tasks, since to determine which entities are the same entities for each database is difficult. In this task, the existing literature reviews in the context of frameworks, methodologies or techniques that solve the ER problem in heterogeneous data sources have been evaluated.

The objective of this SMS is identifying what has been already done and what must be done in the future in the context of ER in heterogeneous data sources. It is completely different to the reviews performed until today. This SMS differs from the revisions made earlier in the related work section in four aspects: (i) the goal is different, (ii) some works are based in a specific area and others are more general but do not cover all the areas, (iii) this revision is broader and more systematic (considering that only one of existing makes a systematic review) and the classification of primary studies is more exhaustive and (iv), we have not found recent surveys or systematic studies related to this domain that are not based on a unique method, technology or technique. Thus, an extension of the work done up to date is proposed in this work.

FORMULATING RESEARCH QUESTIONS

Fulfilling the objective of understanding the existing research proposals within ER problem in heterogeneous data sources, it was necessary to formulate some research questions (RQ). RQs will guide and center our research and they will be clearly focused on the topic, as well as they will synthesize multiple sources to present our unique argument. Table II-2 lists the RQs proposed for this SMS together with their motivations.

Research Question	**Motivation**
RQ1. What methods, techniques or tools have been investigated for ER in the heterogeneous data sources environment?	Find out what methods, techniques or tools have been investigated for ER in heterogeneous data sources environment.
RQ2. What methods, techniques and tools have been used for ER in the heterogeneous data sources environment?	Determine if the proposed research works in this field are more practical or theoretical and identify opportunities for future research works.
RQ3. What is the nature of found methods, techniques and tools for solving ER in heterogeneous data sources environment?	Identify the nature of found methods, techniques and tools for ER in heterogeneous data sources environment and assess the state of this field.
RQ4. What are the objectives pursued in research works for solving the ER in heterogeneous data sources environment?	Point out what the major point of research interest is and which areas have been less investigated, by exploring concepts, compiling current knowledge or advancing the practice through the science of design practices.

Table II-2. Research Questions

DEFINING THE REVIEW PROTOCOL

Now, once the necessity of undertaking this research work has been identified and the research questions that guide it has been formulated, we will describe each of the elements of the protocol defined for this SMS defining: search strategy, procedure for selection of studies, checklists and procedure for evaluating the quality of studies, data extraction strategy, data synthesis, dissemination strategy and project calendar.

SEARCH STRATEGY

This section describes the method that has let us deeply search the most relevant papers related to the topic that we are working on in the principal digital libraries. The searching strategy has been divided into three phases: pre-search, systematic search and manual search.

In the pre-search phase, keywords for the search were selected. As this selection was known to be relevant for the quality of results, general terms have been used with the aim of confirming that most of the research papers are included in the study. We have classified these terms in two main categories: problem, and technologies, tools, frameworks and concepts. The problem category is based on the ER problem, having this key as the main one and getting all the synonyms identified in the pre-search that refer to this problem. The technologies, tools, frameworks and concepts category is based on the domain where we tend to apply the category of the problem. After that, a combination between both categories and all the sets of words identified for each one was carried out. The initial list of words is shown in Table II-3.

Concept	Keywords
ER problem	Entity Matching. Entity Identity, Entity Name System, Entity Recognition, Entity Parsing, Entity Linking, Entity disambiguation, Entity Resolution, Entity Reconciliation, Identity Matching, Identity Management, Identity Resolution, Identity Attributes, Identity Search, Identity Linking, Duplicate Detection, Deduplication, Record Linkage, Object Identification, Reference Matching, Co-Reference Detection, Non-identical Duplicates, Redundancy elimination, Object Matching , Fuzzy Matching, Similarity join processing, Duplication Detection, Reference Reconciliation, Co-Reference Resolution, Relational Blocking
Technologies, tools, frameworks and concepts	Data Integration, Heterogeneous Data Sources, Data Sources, Data warehouse, Unstracted data, Inter-media data retrieval, ETL, Extract transform load, Extract transform and load, Big Data, Open Data, Database Management, Data quality, DSL, Domain specific language, Massive Data, Large Data, MDE, Model Driven Engineering

Table II-3. First set of keywords giving main terms

The first set of databases were taken according to the criteria presented by Ngai et Al., (2011) adding some other new ones proposed by the PhD student. These are: ABI/INFORM Database, Academic Search Premier, ACM, Business Source Premier, Emerald Full text, IEEE Xplore Digital Library, Science Direct, Springer-Link Journals, World Scientific Net, SCOPUS and Web Of Knowledge. Once the searching process was finished, it was realized that some databases included articles already found in others, therefore, they did not bring new value. Thus, the articles were grouped into four large databases: ACM, IEEE Xplore Digital Library, SCOPUS and Web Of Knowledge.

In the systematic search of phase two, once the relevant keywords have been found and some pilot testing was carried out, a Python script was developed for making the combination between all of them. In this context, two category files were created: one for the ER problem and another one for the technologies, tools, frameworks and concepts. Having these two files, the script was programed by taking one of the keywords of the ER problem file and combining it with all the keywords of the second file. Besides, search queries were generated concretely for each database selected to conduct the systematic search. All kind of papers have been included such as: journal papers and presentations at conferences, congresses, tutorials and workshops. A very large number of queries have been executed and for each database they have been customized depending on: the query syntax of the relevant database, the possibilities that the database offers to make filters, year of publication and specific topics. Table II-4 shows some examples of the queries that have been executed.

	Database	Keywords
Query 1	Scopus	2010 (TITLE("Data fusion") AND KEY("Entity Matching")) OR (TITLE("Data fusion") AND KEY("Entity Identity")) OR (TITLE("Data fusion") AND KEY("Entity Name System")) OR (TITLE("Data fusion") AND KEY("Entity Recognition")) OR (TITLE("Data fusion") AND KEY("Entity Parsing")) OR (TITLE("Data fusion") AND KEY("Entity Linking")) OR (TITLE("Data fusion") AND KEY("Entity disambiguation")) OR (TITLE("Data fusion") AND KEY("Entity Resolution")) OR (TITLE("Data fusion") AND KEY("Entity Reconciliation")) OR (TITLE("Data fusion") AND KEY("Identity Matching")) OR (TITLE("Data fusion") AND KEY("Identity Management")) OR (TITLE("Data fusion") AND KEY("Identity Resolution")) OR (TITLE("Data fusion") AND KEY("Identity Attributes")) OR (TITLE("Data fusion") AND KEY("Identity Search")) OR (TITLE("Data fusion") AND KEY("Identity Linking")) OR (TITLE("Data fusion") AND KEY("Duplicate Detection")) OR (TITLE("Data fusion") AND KEY("Deduplication")) OR (TITLE("Data fusion") AND KEY("Record Linkage")) OR (TITLE("Data fusion") AND KEY("Object Identification")) OR (TITLE("Data fusion") AND KEY("Reference Matching")) OR (TITLE("Data fusion") AND KEY("Co-Reference Detection")) OR (TITLE("Data fusion") AND KEY("Non-identical Duplicates")) OR (TITLE("Data fusion") AND KEY("Redundancy elimination")) OR (TITLE("Data fusion") AND KEY("Object Matching ")) OR (TITLE("Data fusion") AND KEY("Fuzzy Matching")) OR (TITLE("Data fusion") AND KEY("Similarity join processing")) OR (TITLE("Data fusion") AND KEY("Duplication Detection")) OR (TITLE("Data fusion") AND KEY("Reference Reconciliation")) OR (TITLE("Data fusion") AND KEY("Co-Reference Resolution")) OR (TITLE("Data fusion") AND KEY("Relational Blocking"))
Query 2	ACM	2010 ("Title":"large data" AND "Title":"entity matching") OR+("Title":"large data" AND "Title":"entity identity") OR+("Title":"large data" AND "Title":"entity name system") OR+("Title":"large data" AND "Title":"entity recognition") OR+("Title":"large data" AND "Title":"entity parsing") OR+("Title":"large data" AND "Title":"entity linking") OR+("Title":"large data" AND "Title":"entity disambiguation") OR+("Title":"large data" AND "Title":"entity resolution") OR+("Title":"large data" AND "Title":"entity reconciliation") OR+("Title":"large data" AND "Title":"identity matching") OR+("Title":"large data" AND "Title":"identity management") OR+("Title":"large data" AND "Title":"identity resolution") OR+("Title":"large data" AND "Title":"identity attributes") OR+("Title":"large data" AND "Title":"identity search") OR+("Title":"large data" AND "Title":"identity linking") OR+("Title":"large data" AND "Title":"duplicate detection") OR+("Title":"large data" AND "Title":"deduplication") OR+("Title":"large data" AND "Title":"record linkage") OR+("Title":"large data" AND "Title":"object identification") OR+("Title":"large data" AND "Title":"reference matching") OR+("Title":"large data" AND "Title":"co-reference detection") OR+("Title":"large data" AND "Title":"non-identical duplicates") OR+("Title":"large data" AND "Title":"redundancy elimination") OR+("Title":"large data" AND "Title":"object matching ") OR+("Title":"large data" AND "Title":"fuzzy

		matching") OR+("Title":"large data" AND "Title":"similarity join processing") OR+("Title":"large data" AND "Title":"duplication detection") OR+("Title":"large data" AND "Title":"reference reconciliation") OR+("Title":"large data" AND "Title":"co-reference resolution") OR+("Title":"large data" AND "Title":"relational blocking")
Query 3	IEEE	2010 ("Document Title":"Data fusion" AND "Document Title":"Entity Matching") OR ("Document Title":"Data fusion" AND "Document Title":"Entity Identity") OR ("Document Title":"Data fusion" AND "Document Title":"Entity Name System") OR ("Document Title":"Data fusion" AND "Document Title":"Entity Recognition") OR ("Document Title":"Data fusion" AND "Document Title":"Entity Parsing") OR ("Document Title":"Data fusion" AND "Document Title":"Entity Linking") OR ("Document Title":"Data fusion" AND "Document Title":"Entity disambiguation") OR ("Document Title":"Data fusion" AND "Document Title":"Entity Resolution") OR ("Document Title":"Data fusion" AND "Document Title":"Entity Reconciliation") OR ("Document Title":"Data fusion" AND "Document Title":"Identity Matching") OR ("Document Title":"Data fusion" AND "Document Title":"Identity Management") OR ("Document Title":"Data fusion" AND "Document Title":"Identity Resolution") OR ("Document Title":"Data fusion" AND "Document Title":"Identity Attributes") OR ("Document Title":"Data fusion" AND "Document Title":"Identity Search") OR ("Document Title":"Data fusion" AND "Document Title":"Identity Linking") OR ("Document Title":"Data fusion" AND "Document Title":"Duplicate Detection") OR ("Document Title":"Data fusion" AND "Document Title":"Deduplication") OR ("Document Title":"Data fusion" AND "Document Title":"Record Linkage") OR ("Document Title":"Data fusion" AND "Document Title":"Object Identification") OR ("Document Title":"Data fusion" AND "Document Title":"Reference Matching") OR ("Document Title":"Data fusion" AND "Document Title":"Co-Reference Detection") OR ("Document Title":"Data fusion" AND "Document Title":"Non-identical Duplicates") OR ("Document Title":"Data fusion" AND "Document Title":"Redundancy elimination") OR ("Document Title":"Data fusion" AND "Document Title":"Object Matching ") OR ("Document Title":"Data fusion" AND "Document Title":"Fuzzy Matching") OR ("Document Title":"Data fusion" AND "Document Title":"Similarity join processing") OR ("Document Title":"Data fusion" AND "Document Title":"Duplication Detection") OR ("Document Title":"Data fusion" AND "Document Title":"Reference Reconciliation") OR ("Document Title":"Data fusion" AND "Document Title":"Co-Reference Resolution") OR ("Document Title":"Data fusion" AND "Document Title":"Relational Blocking")
Query 4	Web Of Knowledge	2010 (Data fusion AND Entity Matching) OR (Data fusion AND Entity Identity) OR (Data fusion AND Entity Name System) OR (Data fusion AND Entity Recognition) OR (Data fusion AND Entity Parsing) OR (Data fusion AND Entity Linking) OR (Data fusion AND Entity disambiguation) OR (Data fusion AND Entity Resolution) OR (Data fusion AND Entity Reconciliation) OR (Data fusion AND Identity Matching) OR (Data fusion AND Identity Management) OR (Data fusion AND Identity Resolution) OR (Data fusion AND Identity Attributes) OR (Data fusion AND Identity Search) OR (Data fusion AND Identity Linking) OR (Data fusion AND Duplicate Detection) OR (Data fusion AND Deduplication) OR (Data fusion AND Record Linkage) OR (Data fusion AND Object Identification) OR (Data fusion AND Reference Matching) OR (Data fusion AND Co-Reference Detection) OR (Data fusion AND Non-identical Duplicates) OR (Data fusion AND Redundancy elimination) OR (Data fusion AND Object Matching) OR (Data fusion AND Fuzzy Matching) OR (Data fusion AND Similarity join processing) OR (Data fusion AND Duplication Detection) OR (Data fusion AND Reference Reconciliation) OR (Data fusion AND Co-Reference Resolution) OR (Data fusion AND Relational Blocking)

Table II-4. Example of queries

Once the queries for each database were created, a new specific Python script was designed for each one. Besides, Selenium, a software testing tool for Web-based applications (Selenium, 2017) was used. The process of searching a paper in each database was replicated and it was automated for getting the results based on the queries created before using this Python script and Selenium. Finally, another Python script was developed for removing the duplicate records found out during the process of search.

Because the process of analysis of the results obtained was quite long over time, the searching process previously described was repeated twice.

In the last phase of manual search, papers recommended by experts in the ER problem were looked for. These papers were very important because they were very close to the topic and we could discard them because of the problem of bias.

The Web version of the application Mendeley was used for managing this amount of data. It is a reference manager tool that helps to handle papers. Mendeley is integrated into the Web browser, allowing adding directly the articles from the digital libraries to a personal document database, avoiding the duplicated ones and saving them (when possible) in PDF format (Mendeley Support Team, 2011).

STUDY SELECTION, INCLUSION AND EXCLUSION CRITERIA AND QUALITY INSURANCE

This SMS includes papers written in English that refer to ER problems and technologies, tools or frameworks that try to solve this problem, published from 2010 up to January 2017 in indexed journals, such as Journal Citation Reports (JCR) and prestigious conferences, congresses or workshops categorized in the CORE ranking (CORE Conference Ranking).

It excludes discussion or opinion papers or those that are only available in PowerPoint or abstract formats, duplicates (always considering the most completed one) and those whose main contribution is not referred to ER problems and technologies, tools or frameworks that try to solve it or just scarcely mention it.

The first filter for selecting primary studies was based on the title and abstract of the paper. If it is not relevant to the study, it is automatically excluded. After this process, the inclusion/exclusion criteria were applied when reading the abstracts of the found items. Once read, if there was still any doubt with the inclusion/exclusion criteria, the paper was completely read. The PhD student conducted the selection of the studies and his supervisors, the 30% of the articles to corroborate if the inclusion/exclusion criteria were applied correctly. He/she would consult the other mates in case of doubts or discrepancies.

DATA COLLECTION AND ANALYSIS

First, a quantitative synthesis considering the number and/or percentage of items in each category was made, illustrating them with tables and graphics, to thereby give an answer to each research question, matching each question with category. Moreover, an interpretation of retrieved results and some suggestions deduced from the synthesis are presented.

In addition, it was analyzed: (i) the number of publications per year to detect and justify trends and (ii) the number of publications by publication type to detect the journals in which more has been.

2.2. CONDUCTING

Once the protocol was agreed, the proper study started. There were two main sections during the process of carrying out this SMS: (i) detect and select primary studies and data extraction, and (ii) apply the inclusion and exclusion criteria for selecting the primary studies that will be used for the work, showing the finally selected ones and the data synthesis phase, where a statistical study was conducted. It showed the main conclusions that obtained after running the previous phase.

DETECT AND SELECT PRIMARY STUDIES AND DATA EXTRACTION

Papers published between 2010 and 2017 were found using the search strategy defined in the protocol. Because of the limitations that certain search sources offered (for example, not allowing the use of complex search strings), it was necessary to design specific strings for each source and manipulate the outcome of searches to get the same results that may have been obtained using the original search string. The search was made on the title and abstract of the papers except for those databases that did not allow this. In that case, the search had to be performed in the full text.

Each search source stored search strings, metadata of found items (title, author, year of publication, etc.) and abstracts of the papers.

After reading their abstracts and excluding those irrelevant to the ER problem, 276 papers out of 2,255 were eliminated for being duplicates. Then, according to the range of years that we have chosen, 434 papers that were written before 2010 were also eliminated.

Consequently, the inclusion/exclusion criteria were applied to the 1,545 remaining items, and 1,024 papers that were not classified into the computer science or information systems category were eliminated. The last filter was applied to the heterogeneous data area and 382 papers were discarded, remaining 139 candidates. From them, 72 papers were supposed to be duplicated, remaining 67 papers. Then, 6 old versions were also eliminated and finally, 61 primary studies were analyzed in depth reading the full text. As it was classified in the protocol, primary studies were identified and selected by the first author of the article, while the second author chose randomly 30% to corroborate the correct choice. The doubts that arose during the selection of items were resolved among all the PhD student and his supervisors. Table II-5 shows the 61 primary studies selected.

Title	**Reference**
Entity resolution for distributed probabilistic data	(Ayat et al., 2013)
Incremental entity resolution on rules and data	(Whang and Garcia-Molina, 2014)
Efficient entity resolution based on subgraph cohesion	(Wang et al., 2015)
Domain-specific entity extraction from noisy, unstructured data using ontology-guided search	(Bratus et al., 2011)
Entity resolution for probabilistic data	(Ayat et al., 2014)
Entity resolution based EM for integrating heterogeneous distributed probabilistic data	(Dharavath and Kumar, 2015)
Pay-As-You-Go Entity Resolution	(Whang et al., 2013)
Interaction between Record Matching and Data Repairing	(Fan et al., 2011)
Conflict Resolution with Data Currency and Consistency	(Fan et al., 2014)
Information Fusion for Entity Matching in Unstructured Data	(Ali and Cristianini, 2010)

Dynamic Sorted Neighborhood Indexing for Real-Time Entity Resolution	(Ramadan et al., 2014)
Disambiguation of named entities in cultural heritage texts using linked data sets	(Brando et al., 2015)
Adaptive Connection Strength Models for Relationship-Based Entity Resolution	(Nuray-turan et al., 2013)
Context-based Entity Description Rule for Entity Resolution	(Li et al., 2011)
Efficient and Effective Duplicate Detection in Hierarchical Data	(Leitão et al., 2013)
Entity Disambiguation in Anonymized Graphs Using Graph Kernels	(Hermansson et al., 2013)
HIL: A High-Level Scripting Language for Entity Integration	(Hernández and Koutrika, 2013)
A Clustering-Based Framework to Control Block Sizes for Entity Resolution	(Fisher et al., 2015)
A Probabilistic Model for Linking Named Entities in Web Text with Heterogeneous Information Networks	(Shen et al., 2014)
A Scalable Machine-Learning Approach for Semi-Structured Named Entity Recognition	(Irmak and Kraft, 2010)
BEAR: Block Elimination Approach for Random Walk with Restart on Large Graphs	(Shin et al., 2015)
Beyond 100 million entities: large-scale blocking-based resolution for heterogeneous data	(Papadakis et al., 2012)
Efficient Entity Resolution for Large Heterogeneous Information Spaces	(Papadakis et al., 2011a)
Efficient SPectrAl Neighborhood blocking for entity resolution	(Shu et al., 2011)
Entity Linking on Graph Data	(Yu, 2014)
Entity Matching across Heterogeneous Sources	(Yang et al., 2015)
Entity type recognition for heterogeneous semantic graphs	(Sleeman and Finin, 2013)
A load-balanced mapreduce algorithm for blocking-based entity-resolution with multiple keys	(Hsueh et al., 2014)
Web-based Graphical Querying of Databases through an Ontology: the WONDER System	(Calvanese et al., 2010)
Large-Scale entity resolution for semantic web data integration	(Costa, 2016)
Populating Entity Name Systems for Big Data Integration	(Kejriwal, 2014)
Domain-adapted named-entity linker using Linked Data	(Frontini et al., 2015)
Learning-based entity resolution with MapReduce.	(Kolb et al., 2011)
ERGP: A combined entity resolution approach with genetic programming	(C. Sun et al., 2014)
Learning an accurate entity resolution model from crowdsourced labels	(Wang et al., 2014)
Entity resolution for high velocity streams using semantic measures	(Priya et al., 2015)
A confidence-based entity resolution approach with incomplete information	(Gu et al., 2014)
A framework for entity resolution with efficient blocking	(Shu et al., 2012)
Entity matching: A case study in the medical domain	(Carvalho et al., 2015)
An Identification Ontology for Entity Matching	(Bortoli et al., 2014)
DS-Dedupe: A scalable, low network overhead data routing algorithm for inline cluster deduplication system	(Z. Sun et al., 2014)
A fast entity resolution method based on wave of records	(Liu et al., 2011)
Cleaning Framework for Big Data - Object Identification and Linkage	(Liu et al., 2015)
To compare or not to compare: making entity resolution more efficient	(Papadakis et al., 2011b)
Entity Resolution for High Velocity Streams Using Semantic Measures	(Priya et al., 2015)
An Ensemble Blocking Scheme for Entity Resolution of Large and Sparse Datasets	(Balaji et al., 2016)
Unsupervised Entity Resolution on Multi-type Graphs	(Zhu et al., 2016)
Entity Matching Across Multiple Heterogeneous Data Sources	(Kong et al., 2016)
Efficient Entity Resolution on Heterogeneous Records	(Lin et al., 2016)
Linked Data Entity Resolution System Enhanced by Configuration Learning Algorithm	(Nguyen and Ichise, 2016)
Linking Heterogeneous Data in the Semantic Web Using Scalable and Domain-Independent Candidate Selection	(Song et al., 2016)
Using Memetic Algorithm for Instance Coreference Resolution	(Xue and Wang, 2016)
Rule-Based Method for Entity Resolution	(Li et al., 2015)
Entity resolution in disjoint graphs: an application on genealogical data	(Rahmani et al., 2016)

Parallel Meta-blocking for Scaling Entity Resolution over Big Heterogeneous Data	(Efthymiou et al., 2016a)
Minoan ER: Progressive Entity Resolution in the Web of Data	(Efthymiou et al., 2016b)
Entity resolution in disjoint graphs: an application on genealogical data	(Rahmani et al., 2016)
Semantic-Aware Blocking for Entity Resolution	(Q. Wang et al., 2016)
Online entity resolution using an Oracle	(Firmani et al., 2016)
Entity Resolution-Based Jaccard Similarity Coefficient for Heterogeneous Distributed Databases	(Dharavath and Singh, 2016)
A Blocking Scheme for Entity Resolution in the Semantic Web	(de Assis Costa and de Oliveira, 2016)

Table II-5. Selected primary studies

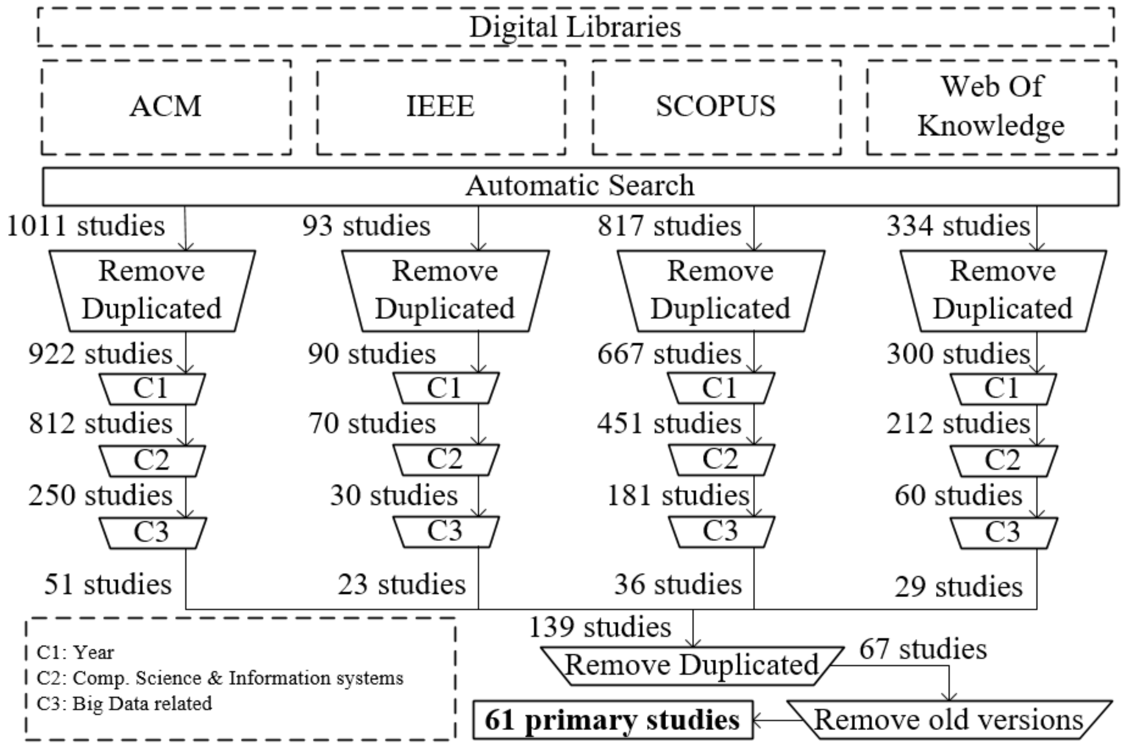


Figure II-2. Primary studies Selection Process

DATA SYNTHESIS

In this section, the information contained in the data extraction form is displayed to answer the research questions formulated previously. In addition to the quantitative data shown through tables and graphics, an interpretation of the results is also presented.

- RQ1. As reflected in Table II-6, a classification that divides the publications in those based on algorithms as solution and those data structured-based is presented. There is a total of 68 publications (seven more than the number of primary studies because there are some papers that mention both categories). Studying data, the percentage of publications is very similar in both categories, having 48.53% the structural ones, the 50% the algorithmic ones and just 1.47% represents the others.

Type	Number	Percent
Structural	33	48.53%
Algorithmic	34	50%
Others	1	1.47%

Table II-6. RQ1 - Data Synthesis

- RQ2. As shown in Table II-7, it is interesting to note that most research works have been validated with a theoretical approach (defining a theoretical approach as that which has validated its proposals with any dataset). They represent 95.08%. Two papers (3.28%) do not present any validation and just one (1.64%) presents a validation based on a real-industry scenario.

Validation	Number	Percent
Not Validated - Theoretical Approach	2	3.28%
Validated - Theoretical Approach	58	95.08%
Validated - Approach in Industry	1	1.64%

Table II-7. RQ2 - Data Synthesis (validation)

Table II-8 presents the datasets used by the authors for validating their approaches. Most of the proposals have been validated with real datasets (76.74%), followed by those which have used both real and synthetic datasets (22.95%) and finally those which have used synthetic datasets (14.75%). It is important to note that there are 77 papers because there are two papers that do not present any validation and 9 of them include the two types of datasets.

Dataset	Number	Percent
Real	54	76.74%
Synthetic	14	22.95%
Real + Synthetic	9	14.75%

Table II-8. RQ2 - Data Synthesis (datasets)

- RQ3. As shown in Table II-9, most research efforts have been focused on graph-based works (26.23%), followed by those based on Clustering/Blocking (22.95%). It also highlights rule-based works (14.75%), and those based on algorithms (16.39%) and probabilistic methods (11.48%). There are two categories based on programming languages and ontologies that represent 4.92%, as well as the learning category that represents 3.28%. Finally, there are three categories based on hints, sorted neighborhood and patterns that represent 1.64%.

Method, Technique, Tools	**Number**	**Percent**
Rule Based	9	14.75%
Probabilistic Method	7	11.48%
Learning Based	3	4.92%
Graph Based	16	26.23%
Programming Languages	2	3.28%
Clustering / Blocking Based	14	22.95%
Ontology	3	4.92%
Patterns	1	1.64%
Sorted Neighborhood	1	1.64%
Algorithms	10	16.39%
Hints	1	1.64%

Table II-9. RQ3 - Data Synthesis

- RQ4. As shown in Table II-10, all the primary studies are based on the operation phase in contrast to the phase of design that only takes 4.92% (three studies present both design and operation phases). More than half of the selected studies (62.30%) apply their experiments to heterogeneous data sources. The lowest result is on multi-applications, where one study that represents 1.64% is mentioned. Finally, automation and multi-domain objectives are poorly represented with only 3.28% and 8.20%, respectively, and just 12 studies that represent 19.67%, mention the multi-relational objective.

	Objective	**Number**	**Percent**
UML	Design	3	4.92%
	Operation	61	100%
ER Challenges	Automation	2	3.28%
	Multi-Relational	12	19.67%
	Multi-Domain	5	8.20%
	Multi-Applications	1	1.64%
Type of Dataset	Heterogeneous	38	62.30%
	Non-heterogeneous	23	37.70%

Table II-10. RQ4 - Data Synthesis

Once the research questions have been answered and after an in-depth study of the retrieved data, some other conclusions are presented.

At it is observed in Table II-11, we can conclude that the topic that we are analyzing in this SMS is arising a lot of interest. From 2010 up to date, the numbers of papers related to ER have been increasing (omitting year 2012 where just one paper less than in 2010 was published). The growth curve between 2012 and 2014 is quite large, almost tripling the number of publications. The number of publications in 2015 remains constant with respect to those published in 2014 and finally, it increases in one more publication having 15 in total. Moreover, Table II-12 summarizes the evolution of the publications based on its category and the year of publication. It shows a clearest trend in this area of research is focused on graph-based methods, techniques or tools followed by those clustering/blocking-based. Those learning-based are the most scattered, finding only one publication in the beginning of the search period and another one in the end. Those sorted neighborhoods and pattern-based but in this case, they are placed at end of the search period.

Table II-13 shows the result that has been retrieved from the different digital libraries. In this case, the ACM Digital Library is on top of the selected primary studies with 36.69% followed by Scopus with 25.90%, Web of Knowledge with 20.86%, and finally IEEE Xplore Digital Library with 16.55%. It is important to remark that the amount of papers of this table is higher because the duplications among databases were not eliminated.

Year	**Number**	**Percent**
2010	3	4.92%
2011	7	11.48%
2012	2	3.28%
2013	6	9.84%
2014	14	22.95%
2015	14	22.95%
2016	15	24.59%

Table II-11. RQ3 - Data Synthesis

Year	**Rules**	**Probabilistic**	**Learning**	**Graph**	**Programming**	**Clustering Blocking**	**Ontology**	**Patterns**	**Sorted Neigh.**	**Algorithm**	**Hints**
2010	0	0	1	2	0	0	0	0	0	0	0
2011	2	0	0	0	0	3	1	0	0	1	0
2012	0	0	0	0	0	2	0	0	0	0	0
2013	0	2	0	2	1	0	0	0	0	0	1
2014	3	3	1	1	1	2	1	0	0	3	0
2015	4	1	0	4	0	3	0	1	1	2	0
2016	0	1	1	7	0	4	1	0	0	4	0
Totals	9	7	3	16	2	14	3	1	1	10	1

Table II-12. RQ3 - Data Synthesis

Library	**Total**	**Percent**
ACM	51	36.69%
IEEE	23	16.55%
SCOPUS	36	25.90%
WOK	29	20.86%

Table II-13. RQ3 - Data Synthesis

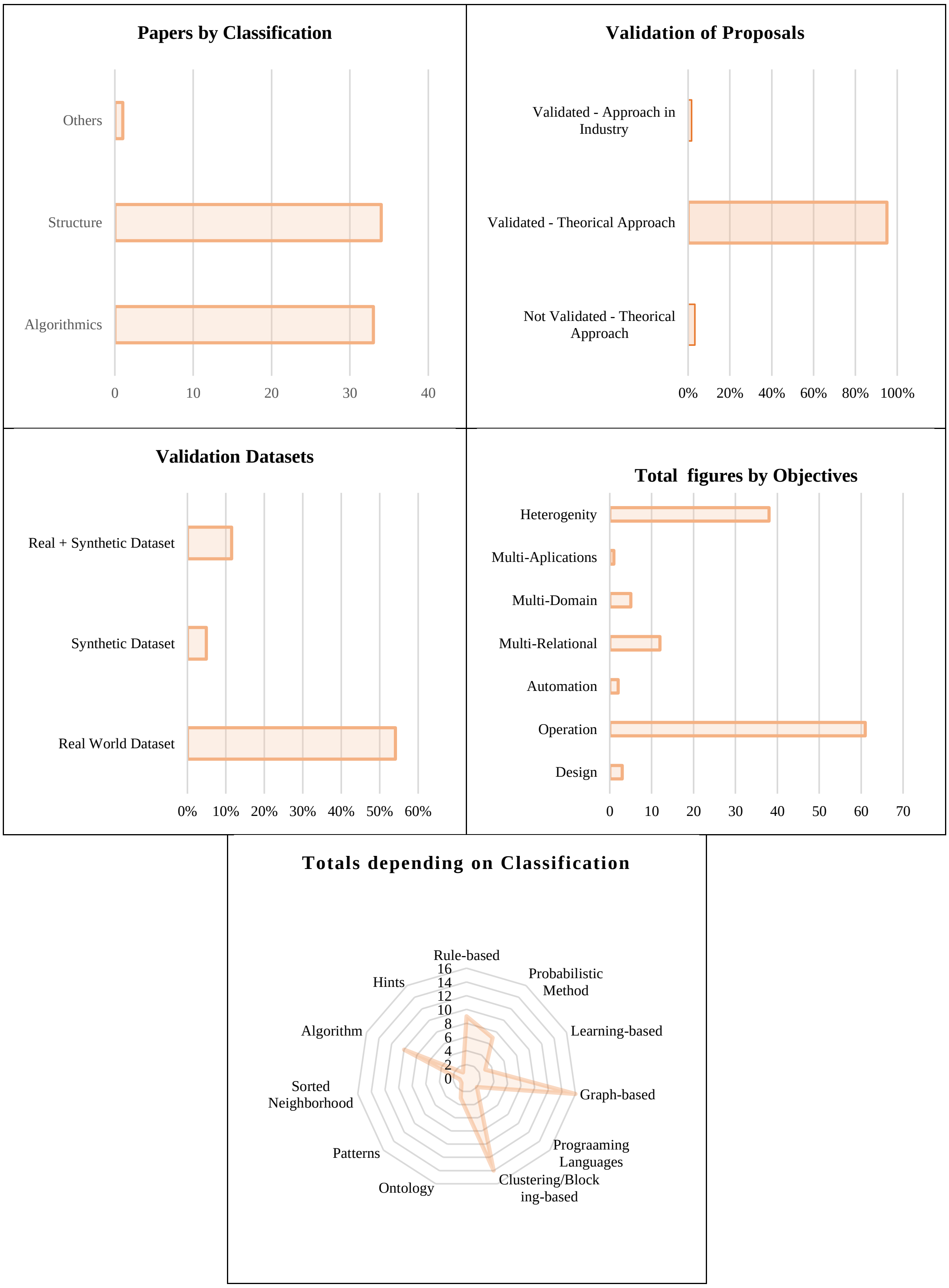


Figure II-3. Data Synthesis Graphics

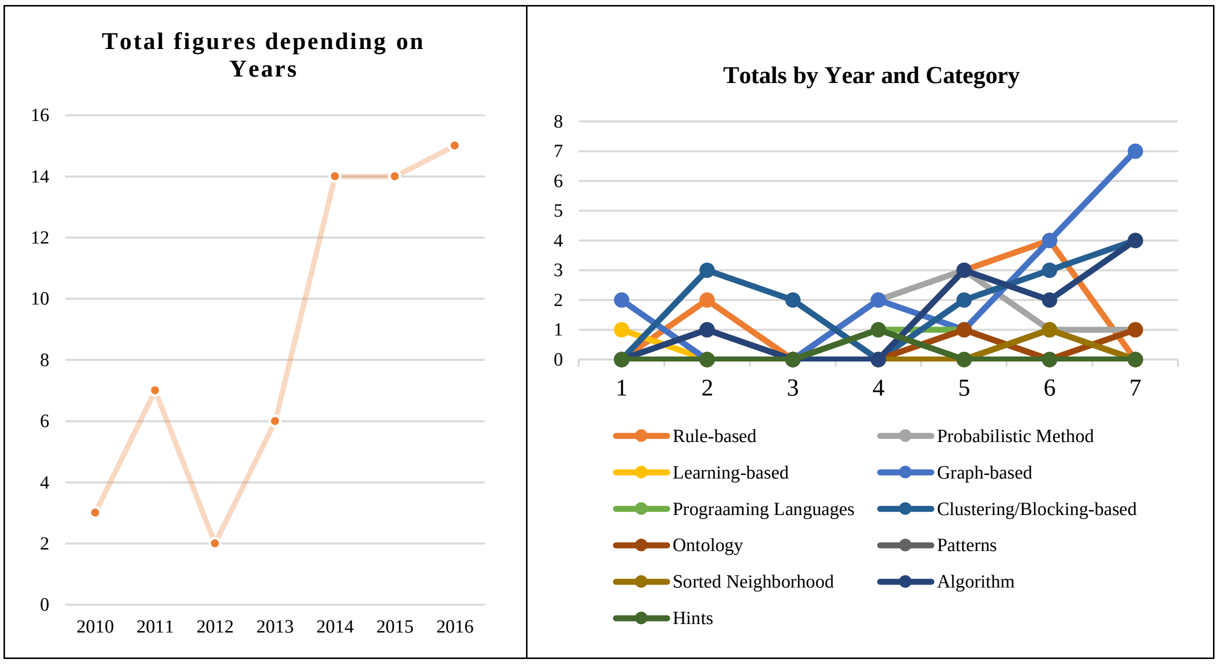


Figure II-4. Total of papers by Year and Category

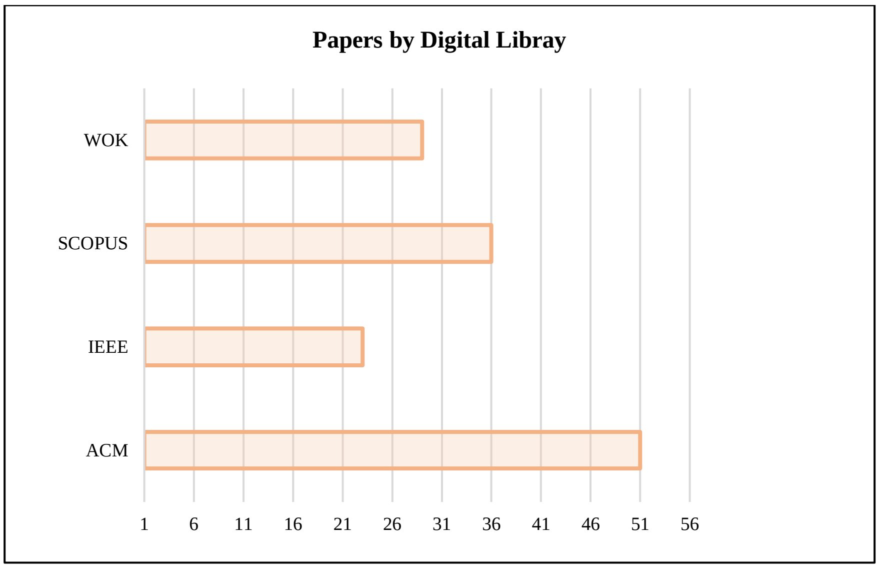


Figure II-5. Total of papers by Digital Library

2.3.Reporting

The main threats to validate this work follow the ones proposed in Shull et al., (2008): bias in the selection of articles, inaccuracy in data extraction and errors that could be taken through the process of classification.

It is impossible to achieve full coverage of everything written on a topic. Four digital research databases were used, including journals, conferences and relevant workshops related to the ER topic. The scope of journals and conferences that has been discussed in this SMS is large enough to reach a reasonable completeness in the studied field.

Helping to ensure a fair selection process, the research questions were defined in advance and the selection of items was organized in a series of stages in which the PhD student and his supervisors were involved. As discussed above, the decision to select primary studies for this SMS was made by the PhD student and rules were rigorously enforced.

Duplication of articles is a potential threat to calculate the frequency of articles and statistical data. As it is also discussed above, an automatic process was applied to remove the duplicated publications, therefore we do not believe that any undetected duplications exist.

3. Discussion

This SMS discovered 61 primary studies classified in peer-reviewed journals, conferences and workshops. They were classified in 4 sections represented by the 4 research questions proposed in section 3. In this section, we will discuss the obtained results.

RQ1 asked: “What methods, techniques or tools have been investigated for ER in heterogeneous data sources environment?”. It was necessary to set a classification of the found solutions. Following Unified Modeling Language (UML) specification (Group, 2017) that classifies diagrams in two categories: (i) structure-based diagrams, which show the static structure of the system and its parts on different abstraction and implementation levels and how they are related to each other, (ii) and behavior-based diagrams, which show the dynamic behavior of the objects in a system extrapolating it to our problem, we have organized the selected primary studies in two big groups with the aim of finding out what methods, techniques or tools have been investigated: (i) structural-based and algorithmic-based solutions, understanding structural-based as those studies that propose a solution supported by data structures, (ii) and algorithmic-based solutions, which refer to those studies in which the solution comes from applying an algorithm. Results show that the structural-based solutions are a little bit important than the algorithmic-based ones with a difference of just 1.57%. It is also relevant to highlight that there are some papers that represent both structural-based and algorithmic-based solutions. With this analysis, it is concluded that the efforts invested by the researches in both solutions is very similar.

RQ2 asked: “What methods, techniques and tools have been used for ER in heterogeneous data sources environment?”. For determining if the research study is more practical or theoretical, we have made a classification based on solutions that have not been validated (present a theoretical approach) and solutions that have been validated (either with own experiments or in the industry), to determine whether the research study is more practical or theoretical. Besides, for the solutions that have been validated, we have classified the type of datasets that was used for the validation as follows: real (real-world dataset), synthetic (non-real-world dataset) and real + synthetic (those which have validated their approach with both types of datasets).

The selection criteria for the selected studies have been very strict, trying to obtain very good quality publications. Therefore, as expected in this type of publications, most studies show a validation. Only one of them presents a theoretical solution which does not mention whether the proposal has been validated or not. Some others show a theoretical validation and only one of them shows a real-case validation performed in industry. Furthermore, most of the studies have been validated using real-world datasets, understandable when researchers aim to provide their proposals with a certain level of quality, and just a few studies show a validation with synthetic datasets and real-world + synthetic datasets.

RQ3 stated: “What is the nature of the methods, techniques and tools found for solving ER in heterogeneous data sources environment?”. A new subdivision of the classification obtained in RQ1 has been created. This subdivision presents: (i) for algorithmic-based solutions: rule-based, probabilistic-based methods, learning-based, programming languages, sorted neighborhood and others algorithms, (ii) for structure-based solutions: graph-based, clustering/blocking-based, ontology-based and pattern-based (iii) and finally, for others: hints.

Considering the previous classification, the ones which take more importance are graph-based and clustering/blocking-based solutions, representing 49.18% of the total. This is because when working with big datasets, the computational time is key and blocking is often used to improve efficiency and graph technology is a natural solution to treat problems related to Big Data and especially for the relationships among entities. Moreover, those solutions based on rules and probabilistic methods, often used since the beginning of the study of this problem in relational databases, are also important.

Finally, RQ4 asked: “What are the objectives pursued in research work for solving ER in heterogeneous data sources environment?”. With the aim to determine where the majority of research interest is and which areas have been little investigated, we have made a classification based on: taking the extrapolation mentioned before of UML, design and operation, challenges for ER proposed in Getoor and Machanavajjhala, (2013), multi-relational, dealing with structure of entities, multi-domain, dealing with customizable methods that span across domains and multi-applications, dealing with systems that serve diverse application with different accuracy requirements, level of automation of the proposal, and finally the types of the that were used for the validation of the proposals, understanding them as heterogeneous or non-heterogeneous.

Results show that most of the proposals focus on the implementation phase that provides a solution to a problem, however only three studies refer to design. It is important to note that in the challenges proposed in Getoor and Machanavajjhala, (2013), a great research effort has not been made since the best score (19.67% of studies) obtained is related to proposals that address the multi-relational objective. Nevertheless, it is not very significant because it is clearly related to graph-based studies. Only 5 studies work with different domains and it is surprising that none of the papers found mention anything about servicing to different applications with different requirements. Moreover, it is also quite relevant that only two of the works found show some level of automation in its solutions. Finally, the heterogeneity of the datasets is acceptable as more than half of the works found (62.30%) use heterogeneous data sources.

4. CONCLUSIONS

In this Chapter a SMS based on the ER problem in heterogeneous data sources has been presented. This SMS is composed of three phases: (i) planning the review, where the necessity of making this research has been demonstrated taking into account that the goal of papers presented in related work is different; some were based in one topic and others in different topics, but they did not cover all the proposed fields in this work and just one paper made a systematic process similar than the presented one, (ii) conducting the review, where the protocol defined was executed, (iii) and reporting results. After managing the defined protocol, the review was conducted, where a total of 61 primary studies were selected.

A comparison table was created to classify the primary studies. The tags identified for classifying them were created according to the specifications of the problem provided by the Regional Government of Andalusia. Criteria such as design or operation phase of the ER problem, level of automation, heterogeneity of datasets used for validation or multi-relational characteristics, among others, have been considered.

The analysis of the results shows that research efforts in structural-based and algorithmic-based solutions are practically the same. Most of the studies have been validated using real-world datasets, but just one of them has applied its proposal in a real case in the industry. The heterogeneity of the datasets is acceptable knowing that more than a half uses heterogeneous data sources. Besides, most of the research work has been focused on the operation phase of the reconciliation and not in the design phase. Finally, the efforts made to automate the process of reconciliation have been very limited.

Considering the obtained results and as future research work, it is proposed to extrapolate the solutions to the problems raised in this research area to real problems in the industry, as it has been noticed that most of the datasets used for testing the proposals are real-world datasets. However, the solutions are not applied to real-world problems in industry. It is necessary to invest more research efforts in the automation of the proposed solutions because we have found just a few alternatives in this regard. Moreover, a new search is required to increase the domains where this study has been applied.

It is also necessary to apply these solutions to multi-applications problems, where different applications with different requirements need to be served with the results of the reconciliation process. To conclude, it is very important to point out that this type of studies should continue to keep it updated and do not let it become obsolete.

CHAPTER III
PROBLEM STATEMENT

CHAPTER III. PROBLEM STATEMENT

In Chapter II, it has been performed a Systematic Mapping Study where it has been presented the actual state of the art in terms of methods, techniques or tools that allow to make the ER in heterogeneous data sources. The analysis of the current situation has allowed a glimpse of the problem to be solved and the approach that this thesis will raise, both will deepen and justify throughout this chapter.

To do this, in the first section are collected, in summary, relevant aspects that have determined the problem to be approached with this thesis, which will be clearly defined in the next section.

Once the problem is defined, the solution proposed in this thesis, the objectives to be achieved, a brief description of the structure of the solution and the most prominent influences affecting of its development is stated.

Finally, the last section or summarizes the main conclusions of this chapter.

1. RELEVANT ASPECTS THAT DETERMINE THE PROBLEM TO BE SOLVED

From the conclusions drawn in the previous chapter, it raises several problems that this work aims to solve:

1. Most of the research work speaking about entity reconciliation has been focused on the operation phase of the reconciliation and not in the analysis and design phase. It means that researches focus their efforts in creating a very efficient algorithm and apply it to any dataset or a set of them, but they do not pay attention in how a user can design a solution to apply this algorithm depending on his necessities and their concrete environment.

2. Efforts based on the automation of the process of the reconciliation are very limited. Just a one of the papers found spoke something about the level of automation of its proposal. Nowadays, the level of automation of any software process is essential for its maintenance, scalability and one of the most important things, reducing costs. These characteristics are critical for a transference to the enterprise environment.

3. The solutions found in the systematic mapping study showed that there was any solution that worked with multi-application problems, where different applications with different requirements need to be served with the results of the reconciliation process. Speaking about integration, it is very important that any application can use data that have been generated after the process of entity reconciliation.

4. The validation of the proposed solutions was used sometimes in real world datasets, sometimes in synthetic datasets and sometimes, in both. However, just one proposal tested its solution with a real dataset from a real problem of the industry. In this case, this is one of the objectives that this thesis work aims to cover and it is the validation with real world problems related to the industry.

5. We do not find any approach that addresses the issue of how to test if the expected solution was the one obtained after the entity reconciliation process.

With all this, therefore, it is made palpable the need to establish within software organizations, effective and automated mechanisms that enable the effective management of data produced after an entity reconciliation process in heterogeneous data sources. This doctoral thesis focus its work in this sense. However, the definition, implementation and deployment of this process is an extensive work and it is out of the scope of this thesis. In our work, we propose to focus our results in the first phase of the life cycle: requirements and analysis, including also acceptance testing. Thus, the main objective of this Doctoral Thesis can be defined as:

“To propose a suitable environment to support the entity reconciliation in the requirements and analysis phases. This environment allows the development team to prepare their future system to guarantee a suitable entity reconciliation with, besides could be systematically tested.”

2. OBJECTIVES OF THE DOCTORAL THESIS

Once the problem and the context of the problem to be solve have been raised, it's time to describe in concrete terms the objectives to be achieved with this thesis. Those are:

1. Perform a **study of the state of the art** of the different existing solutions for the entity reconciliation of heterogeneous data sources, checking if they are being used in real environments. This objective has been achieved with the study presented in the previous chapter.

2. Define and develop a **Framework** for designing the entity reconciliation models by a systematic way for the requirements and analysis phases. As it is introduced in the next sections, the model based paradigm could offer a suitable environment to support this idea. For this purpose, this objective has been divided in three sub objectives:

    2.1. Define a set of activities, represented as a **process** which can be added to any software development methodology to carry out the activities related to the entity reconciliation in the requirements and analysis phase of a software development life cycle. This objective is solved in the Chapter IV where the proposal is described.

    2.2. Define a **metamodel** that allows us to represent an abstract view of our model-based solution. it is formally defined in the Chapter V of the present doctoral thesis.

    2.3. Define a set of **derivation mechanisms** that allow to stablish the base for automate the testing of the solutions where the framework proposed in this doctoral thesis has been used. Considering that the process will be applied in the early stages of the development, it is possible to say that this proposal applies Early Testing. This objective will be covered in the Chapter VI where a description of how it has been achieved is performed.

3. Provide a **support tool** for the framework. This objective is covered in Chapter VII. The support will allow to a software engineer to define the analysis model of an entity reconciliation problem between different and heterogeneous data sources. The tool will be represented as a Domain Specific Language (DSL).

4. **Evaluate the results** obtained of the application of the proposal in a real-world case study.

Figure III-1 shows the architecture proposed for achieving all the objectives described above. At the left, the **"Software Engineer"** interacts with the system modeling an entity reconciliation solution for a concrete entity reconciliation problem. This model is transformed automatically in **"Business Rules"** thanks to the model to text **"Transformation Rules"** defined between the **"Metamodel"** and the **"Business Rules Metamodel"**. In this sense, this business rules are the result of the integration and application of Early Testing in the solution.

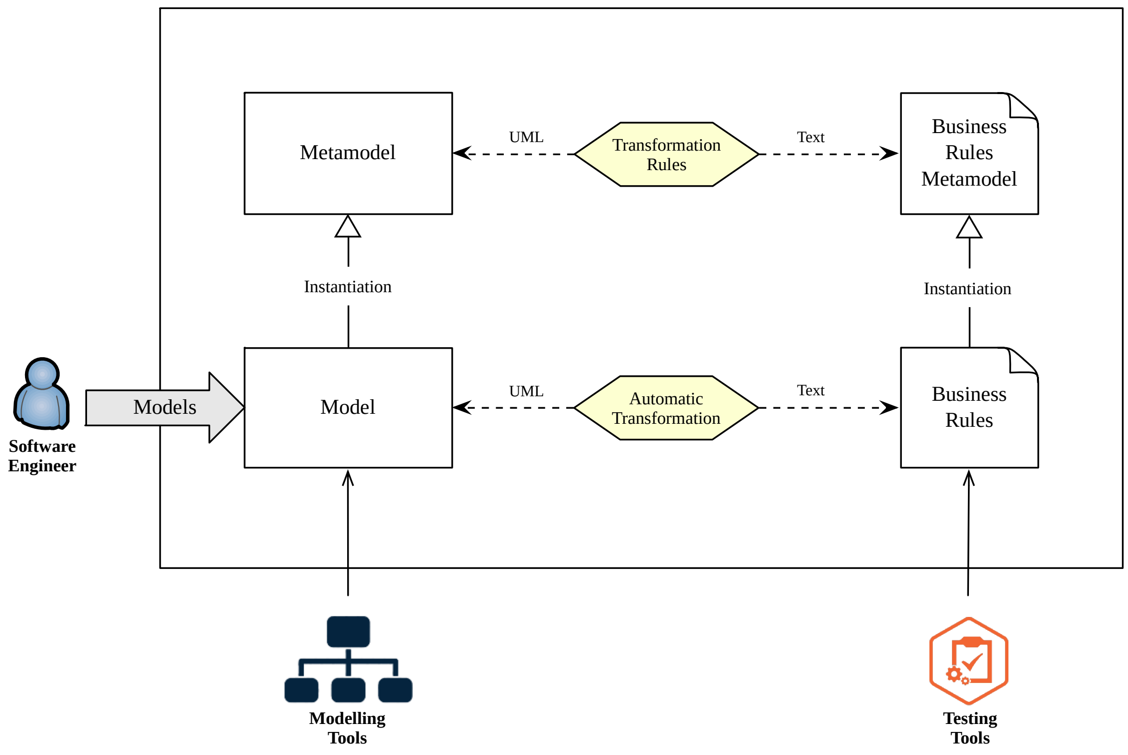


Figure III-1. Solution developed in this Doctoral Thesis

In this moment, once the motivation of the thesis and its objectives have been presented, the next step will be to present a brief introduction of the Framework proposed in this doctoral thesis.

3. PRESENTATION OF THE MARIA (MODEL-DRIVEN ENTITY RECONCILIATION) FRAMEWORK

The main objective of the proposed framework is giving support to the final user to model entity reconciliation problems. For this purpose, the Framework has been developed under the umbrella of the Model-Driven Engineering paradigm that will allows to achieve this objective by a systematic and easy way.

Figure III-2, shows a global view of the MaRIA Framework. Thanks to the proposed solution, the software engineers will be able to model their presumable entity reconciliation models for solving any type of entity reconciliation problem, understanding presumable, the capacity of test the final solutions for checking the coverage level of the new generated dataset without more efforts than the design of the problem thanks to the integration of Early Testing.

MaRIA Framework has been designed so that it can be integrated in any methodology of software development, whether they have classic or agile life cycles. Considering that any software development methodology is composed of several phases, the scope of this Doctoral Thesis has been set up to cover the requirements and analysis phases. Also, the testing phase has been considered adding Early Testing to allow the models to be systematically tested.

MaRIA Framework is composed of three fundamental pillars: “MaRIA process”, “Model-Driven Approach” and “MaRIA Tool”.

“MaRIA process” defines the set of activities that must be added and performed in any software development methodology to develop a solution to an entity reconciliation problem. “Model-Driven Approach” is defined by the metamodel that allows the user that use this framework to model the entity reconciliation problem and derivation mechanisms that to allow the models defined to be systematically tested. “MaRIA Tool” is the support tool developed to give support to the MaRIA Framework.

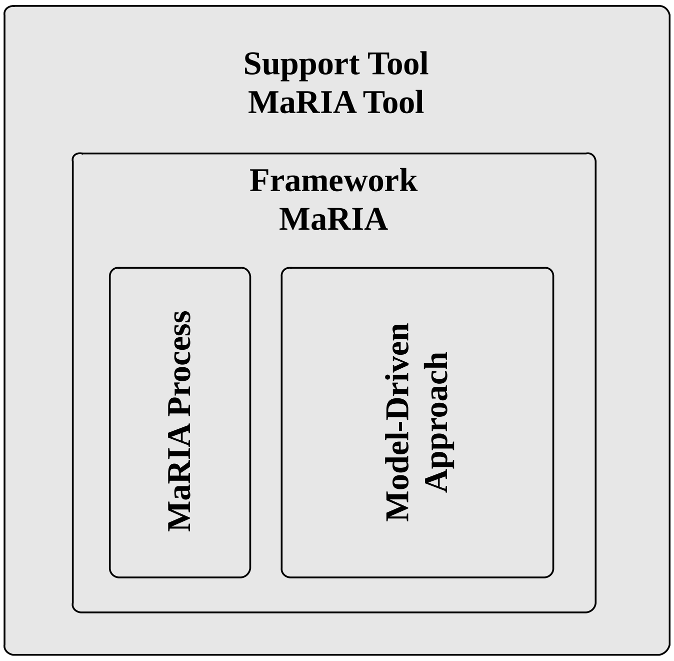


Figure III-2. MaRIA Framework.

In the next three chapters, this brief introduction to the MaRIA Framework will be developed in detail.

4. Conceptual and Technological Influences

When defining how to raise the development of this Doctoral Thesis, there were some aspects, technologies and previous works that influenced the way in which it has developed.

This section introduces the learned lessons of the systematic mapping study performed and developed in Chapter II. Section 3 of this Chapter provides a global view of the MaRIA Framework. As mentioned before, one of the pillars of the Framework is the "Model-Driven Approach" composed of a metamodel and a set of derivation mechanisms. In this sense, the Model-Driven Engineering paradigm has also been considered for the development of this Doctoral Thesis. Finally, this Model-Driven Approach is also based in the Virtual Graphs technology to represent and model the structure of data to be reconciled. Then, Virtual Graph is also covered by this section. All these influences are exposed in detail below.

Learned Lessons of the Systematic Mapping Study

At the beginning of the thesis work, to know which methods techniques or tools were being used for ER was the reason of performing a systematic mapping study. One of the first objectives was to find a method, technique or tool that works with heterogeneous databases and highly scalable allowing to work with any application or domain depending on the necessities of the final user.

During the long process of execution of the systematic mapping study and with the help of the suggestions received from reviewers it was not found any method, technique or tool that covered the goals of this work. There were none better than another, since some ones had advantages or disadvantages that others not depending on the area in which they were applied or some ones covered a set of functionalities that others do not.

Model-Driven Engineering

In the research group in which this Doctoral Thesis has been developed, one of the main lines of research that exists is the Model-Driven Engineering (MDE) paradigm. It allows to focus on the concepts and their relationships and to be free of their concrete representation, representing them abstractly.

Chapter II showed that the research topics in ER are not focused in the level of automation, multi-applications, multi-domain and design of the problem, the most of techniques, tools or methods found are focused in improving an older solution or compare the results of one proposal with the new one proposed. In this sense, the Model-Driven Engineering (MDE) paradigm provides the level of abstraction and automation that requires this proposal and therefore, it will be the technological support in which the solution of this work will be developed. Also, MDE provides a lot of advantages such as:

less error-prone, increased quality, it is more cost-effective or facilitates the reuse of parts of the system in new projects among others.

In the scientific activity, abstraction has been and is widely used, and often referred to it as the activity of modeling. If a model is defined as a partial or simplified reality representation, that allow the final user to address a complex task for a specific purpose, it can be considered a model as the result of abstraction (García-Borgoñón, 2015).

The complexity of software development has been growing up drastically. In this sense, developers noted in models, an alternative for addressing this complexity. MDE emerged to address the complexity of software systems in order to express the concepts of the problem domain in an effective way (Schmidt, 2006). Thus, in the early stages of development, models are more abstract than in the final stages where the models are much closer to implementation. It means, abstract models are transformed into concrete ones the aim of producing software. Studying this process, Brambilla et al., (2012) defined the two fundamental pillars of the MDE paradigm for creating software automatically: models and transformations.

- **Models** must be defined according to the rules of a concrete Modelling Language (ML). This language defines the syntax and semantic of the model. Figure III-3 (Metzger, 2008) shows graphically these relations between syntax and semantic. The ML syntax is composed of a concrete and an abstract syntax. The abstract one defines the language structure and how the different elements can be combined, regardless of its representation. The semantic one, that provides the static and dynamic part, poses restrictions and establishes the meaning of the elements of the language and different ways to combine them. In this moment, it appears the concept of metamodel. A metamodel can be defined as a special type of model that specifies a ML. The metamodel defines the structure and constraints for a family of models (Mellor et al., 2004).

- **Transformations** are the mechanisms that allow to derive models from other existing ones. A transformation between models represent a relation between two abstract syntaxes and it is defined by a set of relations between the elements of the metamodels (Thiry and Thirion, 2009). There are two types of transformations: horizontal (the derived model and the original one have the same abstraction level) and verticals (the derived model has a lower abstraction level than the original one).

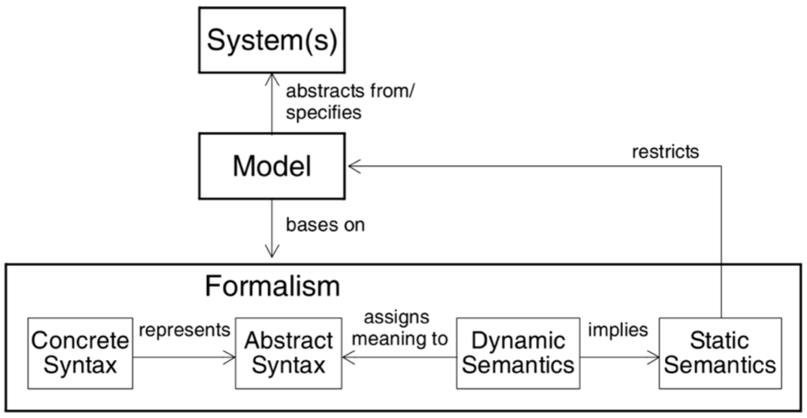


Figure III-3. Syntax and Semantic Relation of the Model (Metzger, 2008)

A very interesting concept found in the MDE literature is proposed by Bézivin, (2005) where “Everything is a model”. In this sense transformations, themselves, are also considered as models. Generally, Figure III-4 shows the transformations between models from a MDE perspective. A transformation models program takes as input a model according to an origin metamodel and produces as output a model according to the target metamodel. The transformation program, should be considered as a model itself.

One of the advantages of MDE is its support for automation, as the models can be automatically transformed from the early stages of development to the final stages. Therefore, MDE allows automating the tasks involved in a software development, such us the testing tasks.

These are, briefly, the main concepts that work in model-based or MDE environments and which are going to produce the theoretical fundaments for building the solution.

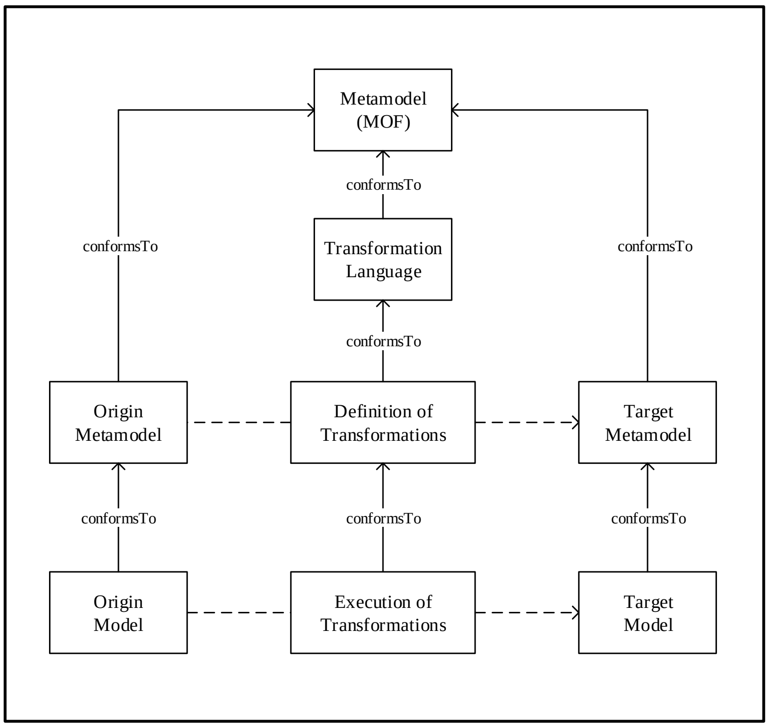


Figure III-4. Transformations between Models.

VIRTUAL GRAPHS

Graph technology is a natural solution to treat problems related to data management and especially for the relationships between entities. In the context of entity reconciliation problems, they do not only lie in comparing the content of the entities, but it also is related to compare the context of the entities for trying to discover entities that do not seem to be the same. This makes not all the data structures are good to represent data and considering that graph structures are the most versatile, it has been selected for representing the reconciled data. The wide variety of existing algorithms, for example: Dijkstra, A*, Kruskal, etc. offer great flexibility in different situations. Theoretically, graphs can be displayed in two ways: explicit and implicit.

An explicit graph is a collection of items (vertexes and edges) that can be stored in memory, which means that each vertex and each edge of the graph can be completely stored in memory.

The problem of using graphs technology is that if they are used implicitly, they occupy a lot of memory resources and in current times where Big Data claims a lot space, makes the task of information processing a handicap. Thus, it was decided to use the virtual version of graph technology for representing data to be reconciled. With this technology, it is possible to build structures on the fly. This will let building different solutions to address many scenarios within a business logic where the predefined data model cannot meet the extensibility or availability of the required data sources. Also, the processing time will be significantly reduced.

A virtual (or implicit) graph is a graph that cannot be completely stored in memory for various reasons, such as size or hardware limitations (Mondal and Deshpande, 2012). It is possible to add the label virtual to a set of elements when it is not needed or it is not wanted to have all the elements of the set on memory and there are only available some properties of it.

For instance, the virtual set of the even integers that compose the range [a,b). It is easily possible to implement the operations of the type of the set without the must of building a data structure that contains all the elements of the set in memory.

A virtual graph is something similar. When the necessities of the problem do not require to store all the vertex and edges of the graph in memory (caused by its size or by the characteristics of the problem), but it is possible to calculate all their neighbors and edges that connect them by an algorithm, we have a virtual graph. A virtual graph has the same types of an explicit graph, such as: directed, non-directed, simple, multigraph, pseudo graph among others. All the vertex and edges of the graph must be unique. For building them, it is available a factory of vertex and another one of edges.

In a virtual graph, it is not important adding or removing vertex or edges, knowing the full set of edges or vertex and iterate over them. By the other hand, it is very interesting to decide if a vertex or an edge belongs to a graph or not, knowing if there are an edge between two given vertexes or obtaining all the edges that exists between these two vertexes.

5. CONCLUSIONS

In this chapter, the relevant aspects that determine the problem to be solved has been presented.

Next, the objectives of the doctoral thesis have been presented. These objectives can be summarized in: study of the state of the art of the different existing solutions for the entity reconciliation of heterogeneous data sources, develop a Framework for designing the entity reconciliation models, provide a support tool for the framework and evaluate the results obtained of the application of the proposal in a real-world case study.

This chapter continues presented the MaRIA (Model-driven entity ReconcilIAtion) Framework which proposes this doctoral thesis. This Framework is composed of three main pillars: MaRIA Process (that can be integrated in any software development methodology for solving entity reconciliation problems) MaRIA Metamodel (that will allow to model the entity reconciliation solutions) and Derivation Mechanisms (that will allow to test the designed models).

In the following chapters, the Framework will be described depth.

CHAPTER IV
MARIA (MODEL-DRIVEN ENTITY RECONCILIATION) PROCESS

CHAPTER IV. MARIA (MODEL-DRIVEN ENTITY RECONCILIATION) PROCESS

Chapter III has defined a detailed approach of the solution offered by this Doctoral Thesis, defining the relevant aspects that determine the problem to be solved, the objectives to be achieved in this work, a high level formal definition of the proposed MaRIA Framework (covering requirements, analysis and testing phase of any software development methodology) and the conceptual and technological influences present in this work.

This Chapter presents the MaRIA process, one of the three pillars that define the MaRIA Framework. The name of the proposal, **MaRIA** (**M**odel-driven entity **R**econcil**IA**tion), is because of the close relationship with the Model-Driven Engineering paradigm and with the Entity Reconciliation process.

1. INTRODUCTION

In this Doctoral Thesis, it has been developed a Framework that is composed of three main pillars: the MaRIA process, a Model-Driven Approach and the MaRIA tool. This chapter aims to explain in detail the MaRIA process.

MaRIA process is defined by a set of activities to be incorporated into any development methodology of any organization to allow preparer the system that is being developed to address an entity reconciliation problem. Concretely, this proposal indicates what are the activities and the artifacts necessary to be able to carry out this type of development.

At this point, it is very important to note that the scope of this Doctoral Thesis is to cover the requirements, analysis and testing processes and also, to keep in mind that the goal of the PhD student is to continue this work extended it to all the different processes that complete the software development process of a software system.

2. INCLUDING MARIA PROCESS IN A SOFTWARE DEVELOPMENT METHODOLOGY

One of the big research areas of the research group where this Doctoral Thesis has been developed is the Web engineering. Then a Web engineering methodology has been the chosen to include the MaRIA Process to show a real use case application.

During the last years, the Web engineering community has proposed several different methodologies for Modeling Web applications with different concepts and definitions such as UWE (UML-based Web Engineering) (Koch et al., 2008), WebML (The Web Modeling Language) (Ceri et al., 2000), OOH4RIA (Meliá et al., 2008), RUX-Method (Rossi et al., 1995) or Navigational Development Techniques (NDT) (Escalona and Aragón, 2008) methodology among others (Domínguez-Mayo et al., 2014).

To show these activities in a real context and because of the great baggage of the development of projects of the research group where this doctoral thesis has been done, as well as the high experience and acceptance rate that this methodology has demonstrated in several projects, it has been selected the NDT methodology as the base for including this proposal.

2.1. Navigational Development Techniques (NDT)

Navigational Development Techniques (NDT) is a methodology based on model driven engineering that allows an organization to develop a software system. It has a set of automatic transformations to help the organization to automate this process. NDT-Q (Figure IV-1) is a framework that defines all the processes supported by NDT Methodology.

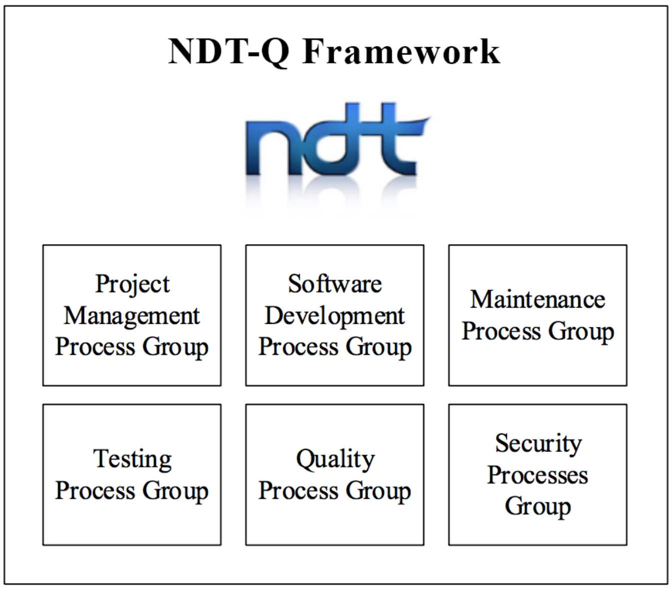

NDT-Q Framework
ndt
Project Management Process Group
Software Development Process Group
Maintenance Process Group
Testing Process Group
Quality Process Group
Security Processes Group

Figure IV-1. NDT-Q Framework

- **Project Management Process Group**. It defines all the processes necessary for the management of a software project. All the processes that NDT methodology specifies for the management of a project are based on the international guidelines of good practices for the management of projects: PRojects IN Controlled Environments (Prince2) (Axelos, 2016) and Project Management Body of Knowledge (PMBOK) (PMI, 2008).
- **Software Development Process Group**. It defines all the processes necessary for the development of a software project. All the processes that the NDT methodology specifies for the software development, work with the Unified Modelling Language (UML) (Group, 2017), being also compatible with Metric V3 (Escalona et Al., 2008). The main advantage of applying the NDT methodology in this context is that it has a set of tools that facilitate the quality assurance throughout the processes. These tools are distributed under the NDT-Suite.

- **Maintenance Process Group**. It defines all the processes necessary for the maintenance of a software project. All the processes that the NDT methodology specifies for the software maintenance are based on good practices defined in Information Technology Infrastructure Library (ITIL) (Cartlidge et Al., 2007) and Capability Maturity Model Integration (CMMI) (Software Engineering Institute, 2010).
- **Testing Process Group**. It defines all the processes necessary for the testing of a software project. This group or processes is subdivided into four major groups:
    - **Project Testing** is a generic process grouping all the processes related to the testing in the software project. All of these processes are based on the international standard ISO/IEC/IEEE 29119 (ISO/IEC/IEEE, 2013).
    - **Organizing the Testing Phase** formally defines a set of activities for carrying out the specification, maintenance and continuous improvement of strategic test information within the organization. The information that this specification must contemplate are the objectives and the global scope of the tests within the organization. This specification should also include organizational testing practices and provide a framework for the continuous review and improvement of these policies within the organization, as well as considering aspects such as what type of tests need to be performed in the organization, who will be the responsible of them, how and with what techniques they will be executed or what tools will be used among others.
    - **Managing the Testing Phase** can be done either at the software project level or for different stages of life cycle testing (unit testing in development, acceptance testing in implementation or early testing of requirements, for example)
    - **Running the Testing Phase** consists of those activities that will allow the execution of the test plan and establish the results of the tests. Once the test plan and the control mechanisms have been defined in the test management process, the execution of the test will be started.
- **Quality Process Group.** It defines all the processes necessary for the quality management of a software project. All processes that NDT specifies for quality management in the framework of a software project are based on the standard ISO 9001:2008 (ISO, 2008) and CMMI (Software Engineering Institute, 2010) good practices.
- **Security Process Group.** It defines all the processes necessary for the security of a software project. All processes that NDT specifies for the security management of a software project have been defined based on standard ISO 27001 (Certification Europe, 2014). All the security processes that have been defined have a strong organizational orientation, that is, they are oriented to be applied within the organization which in turn implies their application in the context of the development of software projects carried out within the organization itself.

2.2. INCLUDING MARIA PROCESS INTO NDT

MaRIA process is one of the three pillars that define the MaRIA Framework (Figure IV-2). It defines a set of activities that must be added and performed in any software development methodology to develop a solution to an entity reconciliation problem.

Having NDT methodology as base and considering as mentioned before, that this Doctoral Thesis will only cover the requirements, analysis and testing processes, the ones selected to include the MaRIA Process are: the software development process (in its requirements and analysis processes group) and the testing process group (in its run the tests process group) Figure IV-3 show the final view of how MaRIA Process acts on NDT-Q Framework.

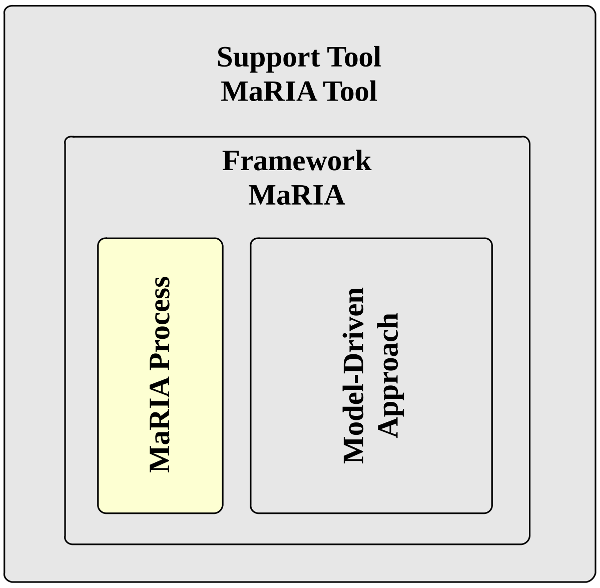


Figure IV-2. MaRIA Framework

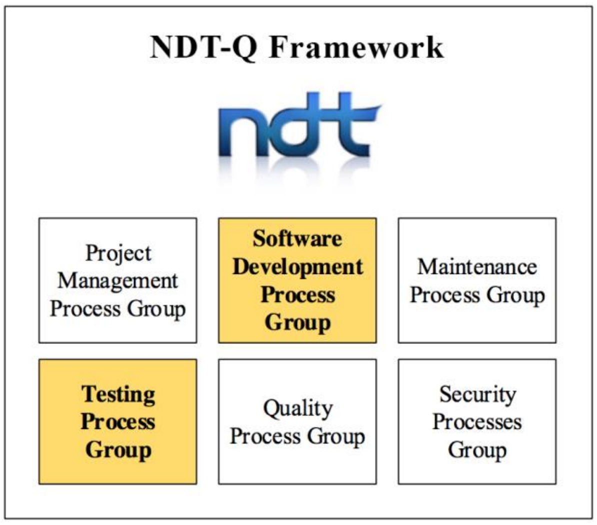

NDT-Q Framework
Project Management Process Group
Software Development Process Group
Maintenance Process Group
Testing Process Group
Quality Process Group
Security Processes Group

Figure IV-3. NDT-Q Framework + MaRIA Framework

3. MaRIA Process

This section will put the focus on the process itself, it means, in all the activities that have been defined and added to the NDT methodology to allow preparer the system that is being developed to address an entity reconciliation problem.

Concretely, this proposal indicates what are the activities and the artifacts necessary to be able to carry out this type of development considering that the scope of this doctoral thesis is focused in the requirements, analysis and testing phase. In this sense, the software development process (in its requirements and analysis processes group) and the testing process group (in its run the tests process group) of NDT methodology has been extended.

It is very important to note that, as NDT, the activities defined for the software development process group of this proposal compliances with Unified Modelling Language (UML) (Group, 2017), being also compatible with Metric V3 (Escalona et Al., 2008). In addition, it has been also defined according to the recommendations and restrictions of ISO/IEC JTC 1 – Big Data (ISO/IEC JTC 1, 2014) standard. Also, the testing process group has been defined in compliance with the international testing standard ISO/IEC/IEEE 29119 (ISO/IEC/IEEE, 2013).

3.1. Software Development Process Group

Software Development Process Group defines all the processes necessary for the development of a software project. To achieve this goal, NDT methodology proposes six big process groups: feasibility, requirements, system analysis, system design, system building and system implementation process.

- **Feasibility process** is carried out in those environments in which technological innovation, the complexity of the system in its own environment, especially in complex or unstable. The deliverable of this phase will be generated by the Enterprise Architect tool using the NDT-Suite plugin.
- **Requirements process** describes the technical sheet for requirements engineering. The deliverable of this phase will be generated by the Enterprise Architect tool using the NDT-Suite plugin.
- **System analysis process** describes the technical sheet for system analysis. The deliverable of this phase will be generated by the Enterprise Architect tool using the NDT-Suite plugin.
- **System design process** is the process by which the environment where the system will be implemented becomes concrete. In previous phases, the project has been platform independent. In this process, the concrete architecture of work is defined and the project is prepared so that it can be implemented.
- **System building process** describes the technical sheet for system building, defining three environments: the development, the pre-production and the production ones.
- **System implementation process** describes the technical sheet for system implementation, defining three environments: the development, the pre-production and the production ones.

As and mentioned before and Figure IV-4 shows, MaRIA process proposes to extend NDT methodology in the requirement and analysis processes. From this set of processes, it is also generated two new documents: data sources report and analysis model. Next, it will be described each of the activities that compose these processes in detail.

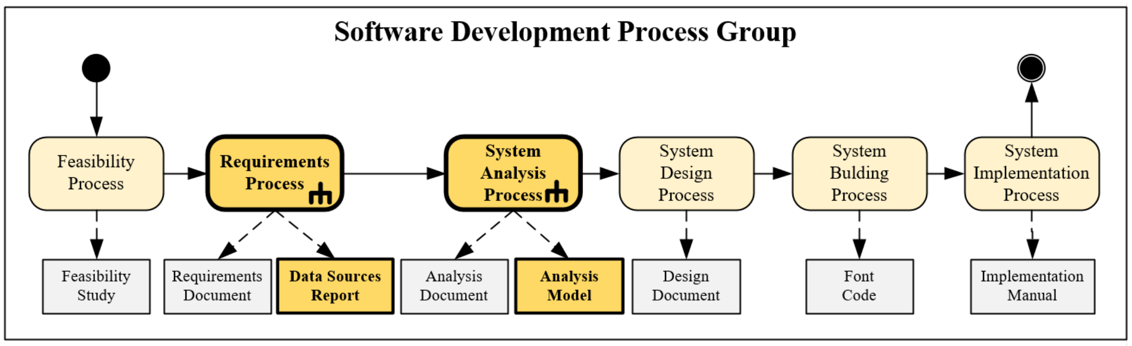


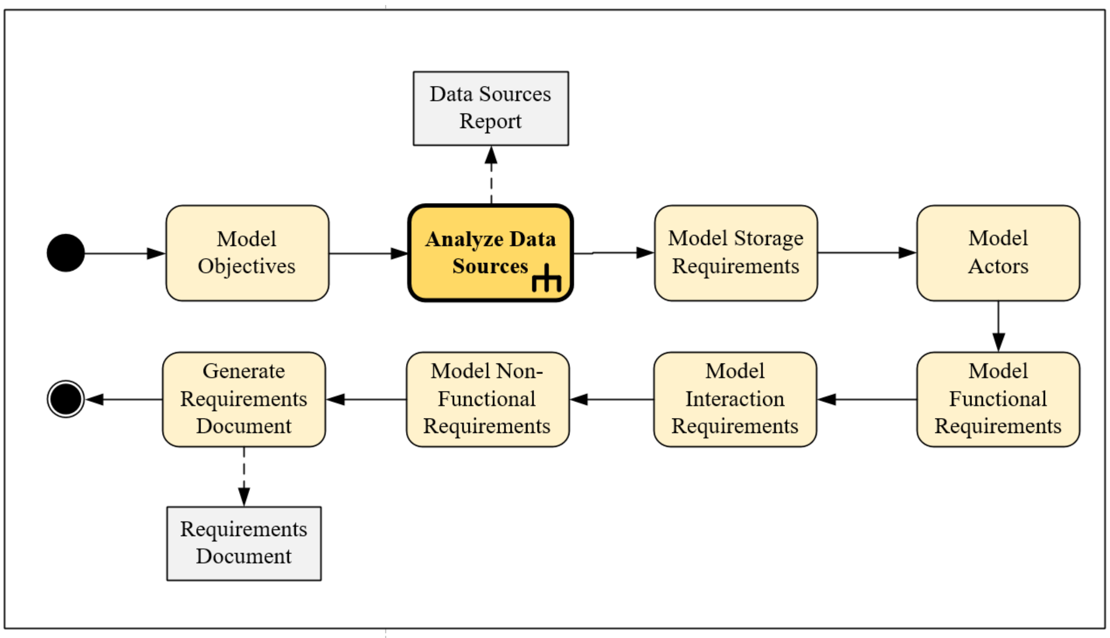
Figure IV-4. Extended software development process group

3.1.1. REQUIREMENT PROCESS

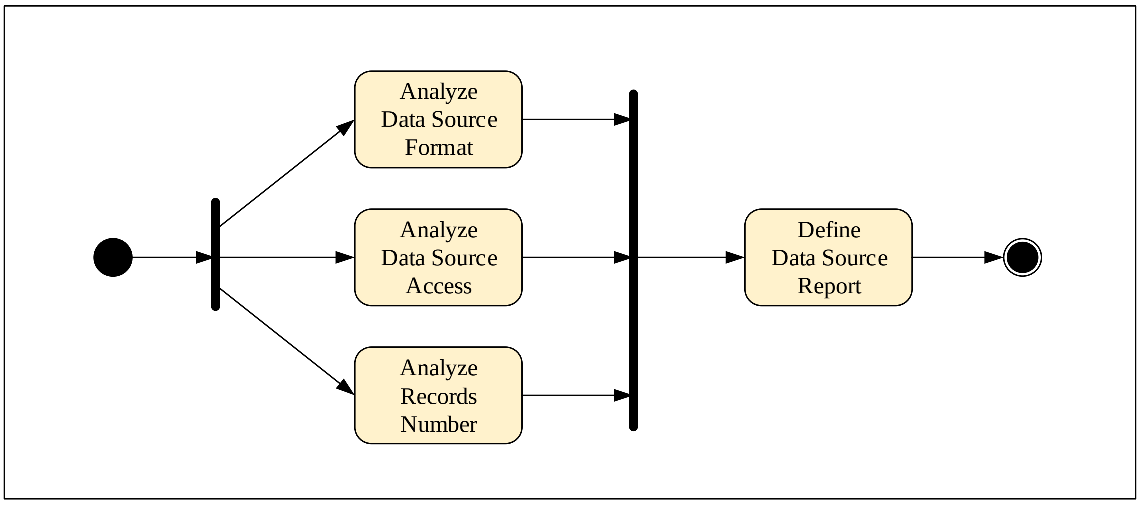
Requirements process of NDT methodology is composed of seven main activities: model objectives, model services, model storage requirements, model actors, model functional requirements, model interaction requirements, model non-functional requirements and generate requirements document.

- **Model objectives.** The development team must make an approach to the environment where the system is to be implemented. They must establish the vocabulary to be used, the users and clients that will participate in the project and the objectives of the project.
- **Model storage requirements.** Storage requirements define the information that must be stored in the system. They are defined through two elements: storage requirements and natures.
- **Model actors.** The objective of this activity is to model the different actors that will interact with the system to be developed.
- **Model functional requirements.** This activity aims to collect what is going to be able to do with the information and the functional possibilities of the system. The functional requirements answer the question of: what can be done in the system?
- **Model interaction requirements.** The objective of this activity is to model the navigation that the system to be developed must have. In this sense, it will be defined: recovery criteria or phrases, lists where the results of a data query are displayed and the display prototypes.
- **Model non-functional requirements.** This activity identifies and collects all these needs that have been left out of the previous activities.
- **Generate requirements document.** This final activity describes the process of creating the documentation that collect all the requirements modeled before.

In this context, what MaRIA process proposes is to add a new activity called "Analyze Data Sources" in the requirements process group of NDT methodology (Figure IV-5).


Data Sources Report
Model Objectives
Analyze Data Sources
Model Storage Requirements
Model Actors
Generate Requirements Document
Model Non-Functional Requirements
Model Interaction Requirements
Model Functional Requirements
Requirements Document
Figure IV-5. Extended requirement process

Analyze Data Sources activity is the activity where the data analysts study and analyze the different data sources that the data consumers have presented for their problem. This activity is composed of four main activities (Figure IV-6). Three of them can be performed in parallel: analyze data source format, analyze data access and analyze number of records of the data source. Once these activities have been finished, it takes place the generate data sources report activate where the data source report is generated. Each of these activities is then described in greater detail next.


Analyze Data Source Format
Analyze Data Source Access
Analyze Records Number
Define Data Source Report
Figure IV-6. Analyze Data Source Activity

- **Analyze Data Source Format** is the activity where the data analysts must study and analyze the data structure of the data sources that the data consumers have defined as problematics. In this sense, they must analyze if each data source refers to a plane text file, a relational database, Oracle, MySQL, a non-relational database or any other type.

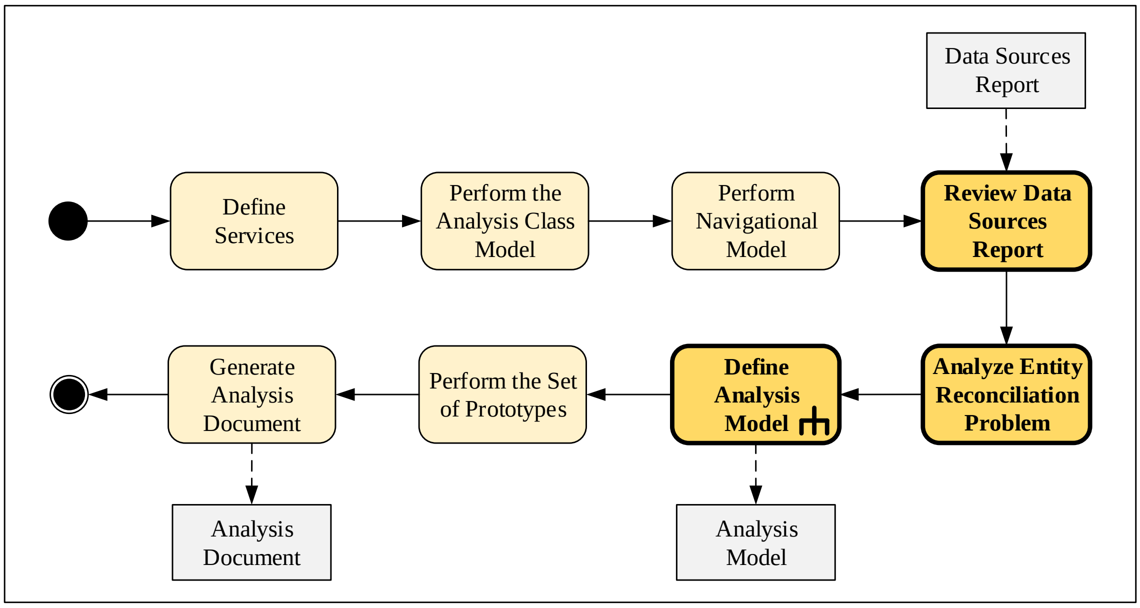
- **Analyze Data Access** is the activity where the data analysts must study and analyze the data access way of the data sources that the data consumers have defined as problematics. In this sense, they must analyze if each data source refers web service, ODBC any other type of database connection or access.
- **Analyze Records Number** is the activity where the data analysts must study and analyze the number of records of each of the data source that the data consumers have defined as problematics (hundred, thousands, millions, etc.). This activity will provide some knowledge and help to choose the technology in which the system will be implemented in the design phase.
- **Generate Data Sources Report** it the activity where the data analysts must create the data sources report with all the information studied and analyzed in previous steps.

3.1.2. Analysis Process

Analysis process of NDT methodology is composed of five main activities: define services, perform the analysis class model, perform navigational model, perform the set of prototypes and generate analysis document.

- **Define services.** This activity aims to define the services that must be created in the software system to be developed.
- **Perform the analysis class model.** The conceptual model represents the static structure of the system and it lets to model how the information structure that the system manages will be. It is represented by two elements: conceptual diagram classes and data dictionary.
- **Perform navigational model.** The navigational model represents how the user will be able to navigate through conceptual information, which elements will appear in that navigation and how they will adapt to the user interacting with the system and finally, the relationships that appear between said elements of navigation.
- **Perform the set of prototypes.** This activity aims to generate a set of prototypes that facilitate the task of validating the models developed in previous activities.
- **Generate analysis document.** This final activity describes the process of creating the documentation that collect all what have been created in previous activities.

In this context, what MaRIA process proposes is to add three new activities called “Review Data Sources Report”, “Define Entity Reconciliation Problem” and “Generate Analysis Model” in the analysis process group of NDT methodology (Figure IV-7).



Data Sources Report
Define Services
Perform the Analysis Class Model
Perform Navigational Model
Review Data Sources Report
Analyze Entity Reconciliation Problem
Define Analysis Model
Perform the Set of Prototypes
Generate Analysis Document
Analysis Document
Analysis Model

Figure IV-7. Extended analysis process

Review Data Sources Report activity is based on the on the study, analysis and review by the software engineer of the strategy document generated in the requirements process group (this document is received as input for performing this activity). This action will provide the software engineer with the knowledge of the data sources needed to model the entity reconciliation problem in the next activity.

Define Entity Reconciliation Problem is the activity where the software engineer must model the entity reconciliation problem. This activity will be carried out using the support tool developed in this doctoral thesis, the MaRIA tool, described in Chapter VII. This activity is composed of seven main steps (Figure IV-8). The four first ones, define wrappers, define data sources, define entities, define attributes, can be performed in parallel. Once defined, the software engineer hast to define the connectors, the data structure and finally, define the transformations.

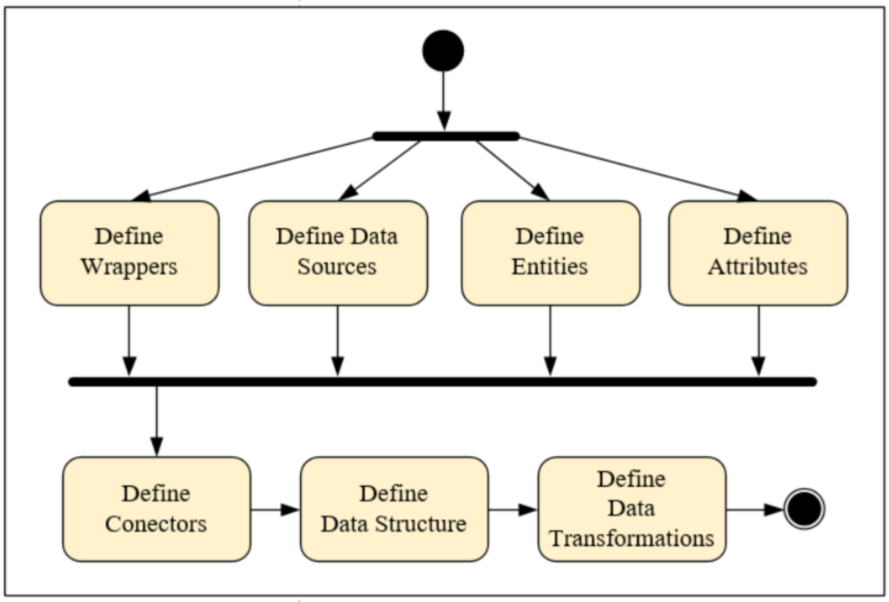


Figure IV-8. Define Entity Reconciliation Problem

- **Define Wrappers** is the activity where the software engineer must model the wrappers that will allow the transfer of information from the data sources that the data consumers have defined as problematics into the entities that will be defined later. In this sense, the software engineer must use the wrapper element of the MaRIA tool and model in the diagram one element for each data source.
- **Define Data Sources** is the activity where the software engineer must model the data sources that the data consumers have defined as problematics. In this sense, the software engineer must use the data source element of the MaRIA tool and model in the diagram one element for each data source.
- **Define Entities** is the activity where the software engineer must model the entities where the information coming from the data sources will be stored in. In this sense, the software engineer must use the data entity element of the MaRIA tool and model in the diagram one element for each data source.
- **Define Attributes** is the activity where the software engineer must model the attributes that compose the entities. In this sense, the software engineer must use the data source attribute element of the MaRIA tool and model as many attributes as each entity needs.
- **Define Connectors** is the activity where the software engineer must model the connectors between the elements that have been defined in the three previous steps. The connectors will relate the wrappers with the data sources, the data sources with the entities and the entities with their attributes. In this sense, the software engineer must use the different types of connectors element that MaRIA tool offers depending on the elements needed to be related.
- **Define Data Structure** is the activity where the software engineer must model the data structure where the data reconciled of the final solution will be stored. For performing this activity, the user must use the entity, attribute and connector elements of the MaRIA tool. In this activity, the software engineer must create the entities, related between them if necessary and the attributes that describe each entity. For each data source, it must be created a structure and in addition to this, it must be created another one that will store the final solution.
- **Define Data Transformations** is the activity where the software engineer must model the data transformations between the different attributes already created or between the data structures. In this sense, the software engineer must use the transformation operations that the MaRIA tool offers and use it for relating attributes or data structures depending on the necessities of the problem.

Finally, the main goal of the **Generate Analysis Model** activity is to generate the model that have been defined thanks to the achievement of the previous activities.

3.2. TESTING PROCESS GROUP

The Testing process group of NDT methodology is composed of four main set of processes: tests of the project, organize the testing phase, manage the testing phase and run the tests (Figure IV-9).

- **Tests of the project.** This generic process groups all the processes related to the Testing in the development of a software system. All of these processes are based on ISO/IEC/IEEE 29119 (ISO/IEC/IEEE, 2013).
- **Organize the testing phase.** This process formally defines a set of activities for carrying out the specification, maintenance and continuous improvement of the strategic information of the tests within the organization. The information that this specification must contemplate are the objectives and the global scope of the tests within the organization. This specification should also include organizational testing practices and provide a framework for the continuous review and improvement of these policies within the organization, as well as considering aspects such as: what type of evidence will be carried out in the organization, who will be the responsible of them, how and with what techniques they will be executed or what tools will be used among others
- **Manage the testing phase.** The test management process can be performed at the software project level for different stages of life cycle testing, it can be represented as: unit testing in development, acceptance testing in implementation or early testing of requirements among others.
- **Run the tests.** This phase of the test cycle will consist of those activities that will allow the execution of the test plan and the results of the tests. Once the test plan and control mechanisms have been defined in the test management process, the execution of the test management process will begin.

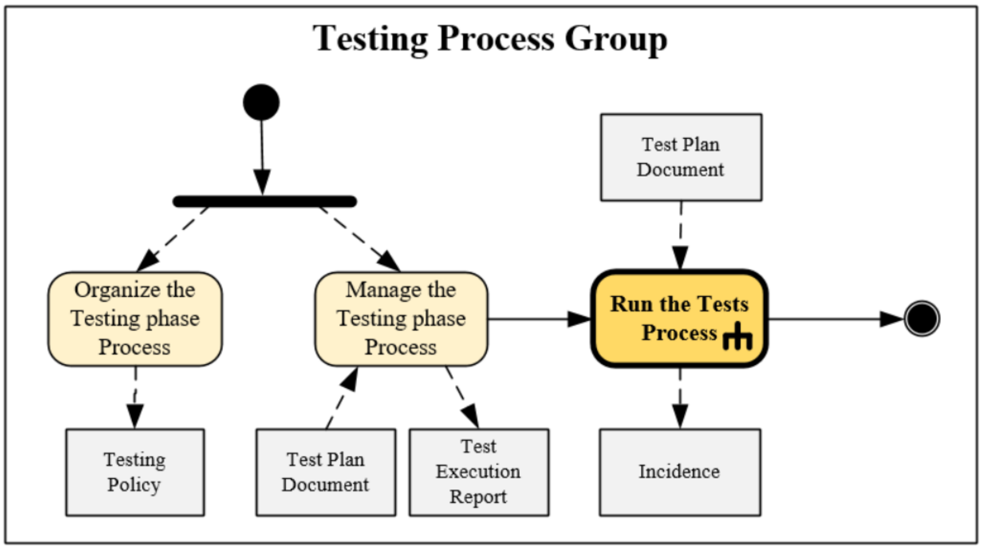


Figure IV-9. Extended testing process group

In this context, what MaRIA process proposes is to add two new activities called “Execute Analysis Model” and “Apply Specific Criteria” and the generation of a new document called “Business Rules Document” in the testing process group of NDT methodology (Figure IV-10).

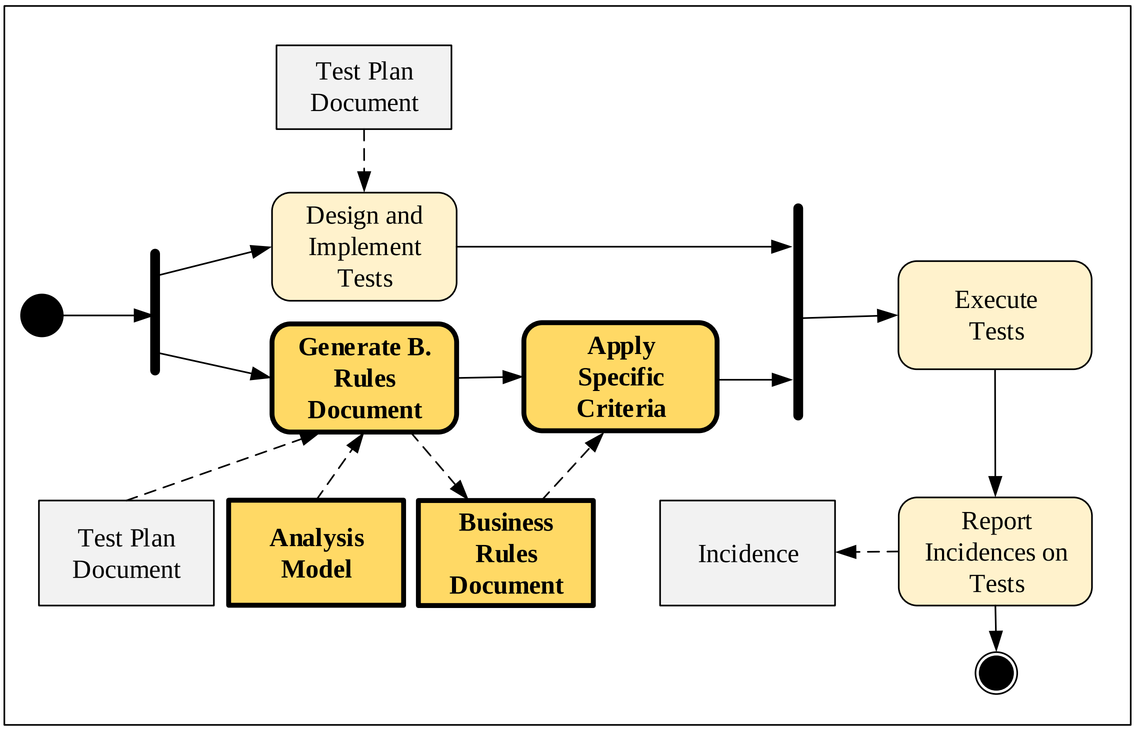


Figure IV-10. Extended Run the tests process

Execute Analysis Model activity receives as input the "Analysis Model" generated in the requirements process group. This model contains the definition of the entity reconciliation problem to solve in the software system that it is being developed. This model also contains a set of derivations defined for the transformations that the information must suffer during the process of reconciliation. The execution of this model will generate the "Business Rules Document".

Business Rules Document is the document generated after performing the execute analysis model activity. This document will contain all the business rules that will define the test requirements of the software system that it is being developed after the application of a concrete criteria.

Apply Specific Criteria activity receives as input the business rules document. To generate the test cases of the software system that it is being developed, it is necessary to apply a specific criterion to the business rules that the business rules document contains. One example of criteria may be Modified Condition/Decision Coverage (MCDC) (Chilenski, 2001). This coverage criterion has demonstrated its utility in previous work, such as (Tuya et al., 2010) (for testing SQL queries) and (Blanco et al., 2012a) (for testing the user-database interaction).

4. Conclusions

This Chapter has presented the MaRIA Process, one of the three pillars that define the MaRIA Framework. The name of the proposal, MaRIA (Model-driven entity ReconcilIAtion), is because of the close relationship with the Model-Driven Engineering paradigm and with the Entity Reconciliation process.

MaRIA process is defined by a set of activities to be incorporated into any development methodology of any organization to allow preparer the system that is being developed to address an entity reconciliation problem. Concretely, this proposal indicates what are the activities and the artifacts necessary to be able to carry out this type of development. This Doctoral Thesis covers the requirements, analysis and testing processes and keeping in mind that, the goal of the PhD student is to continue this work extended it to all the different processes that complete the software development process of a software system.

Due to the great baggage of the development of projects of the research group where this Doctoral Thesis has been developed, as well as the high experience and acceptance rate that this methodology has demonstrated in several projects, Navigational Development Techniques (NDT) methodology was selected to integrate MaRIA process in a real context.

Finally, considering the scope of this work (the requirements, analysis and testing processes), NDT was extended in its software development process (requirements and analysis processes group) and in its testing process group (run the tests process group), creating a total of 15 new activities and 3 new documents. It is very important to note that, although to illustrate the application of MaRIA Framework in a real software development methodology, it has been used NDT, it may be applied to any software methodology such as SCRUM (Schwaber and Beedle, 2001) or BDD (Lazăr et al., 2010) among others.

CHAPTER V
DEFINITION OF METAMODELS

CHAPTER V. DEFINITION OF METAMODELS

Chapter IV has defined a detailed view of the MaRIA Process that is part of the MaRIA Framework. This chapter, describes the languages and concepts needed for modelling the Reconciliation system to covering the requirement, analysis and testing set of processes of the Model-Driven Approach that MaRIA Framework proposes. This approach is composed of two main pillars: the MaRIA metamodel, described in this chapter and a set of derivation mechanisms, described in Chapter VI. The proposed metamodels are formally defined and formally represented by UML class diagrams.

The formal definition of the languages indicated in the previous paragraph is taken as a basis in later chapters to propose derivations that will let to be entity reconciliation problem modeled with the MaRIA metamodel systematically tested.

Finally, this chapter synthesizes a set of conclusions.

1. MODEL-DRIVEN APPROACH

MaRIA metamodel is one of the two pillars that conforms Model-Drvien Approach of the MaRIA Framework. Figure V-1 shows the position that this chapter is focusing inside the Framework.

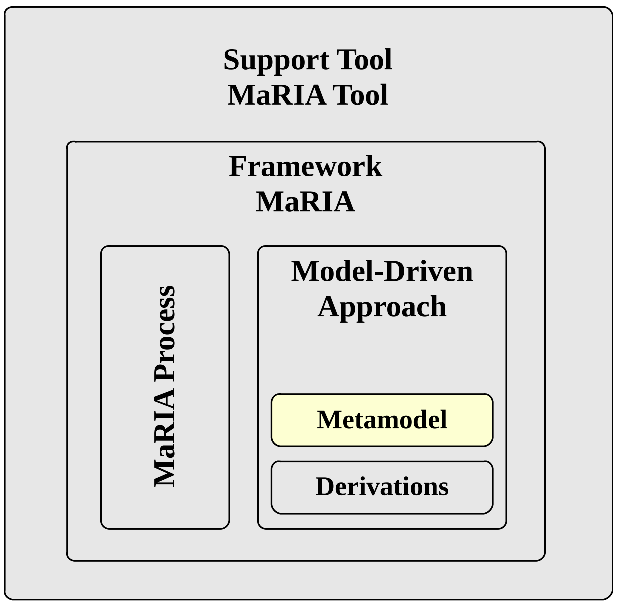


Figure V-1. Framework MaRIA Metamodel

The metamodel for the designing of Entity Reconciliation solutions is showed in Figure V-2. It is the final version of the proposal presented in (Enríquez et al., 2015). As it is possible to note, the global view of the metamodel is composed of these four main blocks: virtual graph metamodel, data source metamodel, transformation metamodel and the testing metamodel, all related between them.

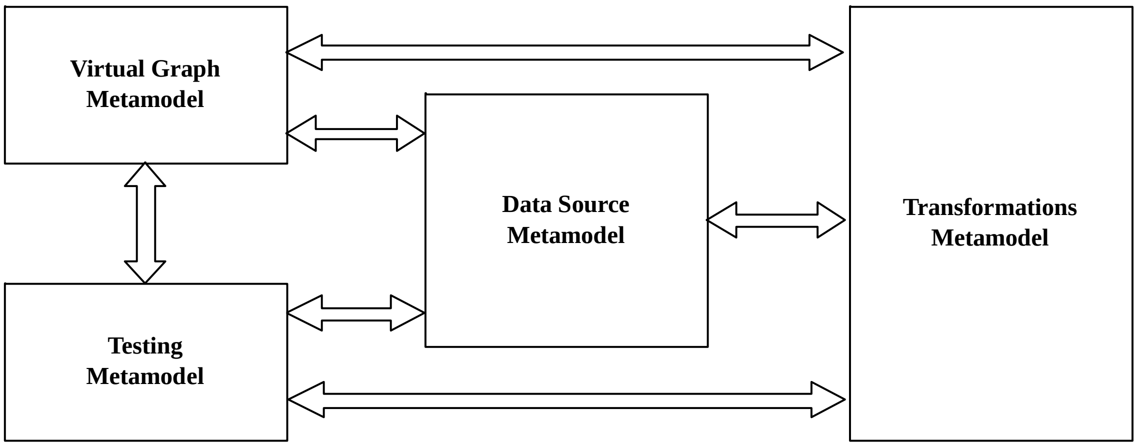


Figure V-2. Global view of MaRIA Metamodel

- **Data Sources Metamodel**: allows representing the information of the data sources to reconcile and the way of accessing to them. These data sources can be a structured or unstructured database, a web service, a warehouse or another information generator.

- **Virtual Graph Metamodel**: allows the user to design the conceptual data model that represents the reconciled solution to achieve, according to the ER problem domain, as a virtual graph.

- **Transformations Metamodel**: represents the different transformations that the data of the sources must undergo to carry out the entity reconciliation and to be consistent with the reconciled solution model. The description of this model is out of the scope of this work.

- **Testing Metamodel**: allows representing the testing objectives for the entity reconciliation application in the early stages of the development. The test models can be focused on different level, as unit testing or integration testing.

During the process of metamodels design, it has been included the attributes required for meeting with the standard ISO/IEC TR 24774 (OMG Group, 2010).

To this metamodel, it will be applied a set of derivations that will generate automatically the test requirements from the model specified by the user. These transformations will be applied to the relations that have been defined in the model. They will be transformations from a MaRIA metamodel to text (business rules). In this sense, the source and the target of these transformations are:

- **Source**: MaRIA Metamodel.
- **Target**: Set of business rules derived from the MaRIA Metamodel in the form of business rules.2

2. Data Source Metamodel

This section describes the Data Source Metamodel and all its components. Figure V-3 shows the metaclases that compose this metamodel and all their relationships. Also, with a low transparency level, it is shown the relationships with metaclases that belong to other blocks of the metamodel.

As mentioned before, this block of the global metamodel allows representing the information of the data sources to reconcile and the way of accessing to them. These data sources can be a structured or unstructured database, a web service, a warehouse or another information generator.

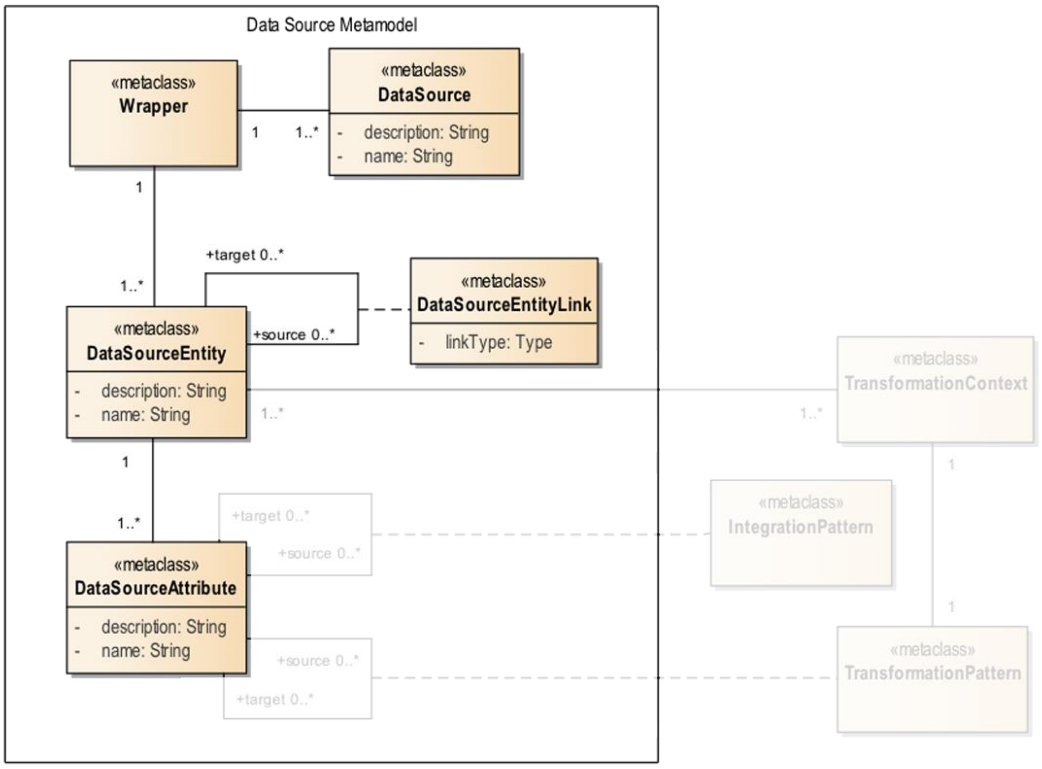


Figure V-3. Data Source Metamodel

This metamodel is composed of five main metaclasses: «Wrapper», «DataSource», «DataSourceEntity», «DataSourceEntityLink» and «DataSourceAttribute». Now, it will be defined each metaclass in detail, for that, it will be exposed their descriptions, generalizations, attributes, operations, associations and constraints.

2.1.«DataSource» Metaclass

Description: This metaclass allows to represent a data source. The final model that the user must define will have as many instances of this metaclass as number of data sources that want to be reconciled.

Generalization: N/A

Attributes: This metaclass is defined by two attributes:

- *name: String [1]*
  Short and concise description with which users can uniquely identify the data source entity.
- *description*: *String [1]*
  Detailed description about the information that a data source entity stores.

Operations: N/A

Associations: This metaclass is defined by one association:

- *has: Wrapper [1]*
  This relation represents the instance which the data source entity must be related to store data retrieved from the data source.

Constraints: N/A

2.2.«WRAPPER» METACLASS

Description: This metaclass allows to the way in which data stored in the instances of «DataSource» metaclass may be retrieved. The final model that the user must define will have as many instances of this metaclass as number of data sources that want to be reconciled.

Generalization: N/A

Attributes: This metaclass is defined by two attributes:

- *name: String [1]*
  Short and concise description with which users can uniquely identify the wrapper.
- *description*: *String [1]*
  Detailed description about the information that a wrapper needs such as: database port, path, host, etc.

Operations: N/A

Associations: This metaclass is defined by one association:

- *has: DataSource [1..*]*
  This relation represents the set of data source in which the instance of the wrapper operates. The same wrapper may be used for one or more than one data sources depending on the type of the data source.

Constraints: N/A

2.3.«DATASOURCEENTITY» METACLASS

Description: This metaclass allows to represent data that has been retrieved from a data source. The final model that the user must define will have as many instances of this metaclass as number of data sources that want to be reconciled.

Generalization: N/A

Attributes: This metaclass is defined by two attributes:

- *name: String [1]*
  Short and concise description with which users can uniquely identify the data source entity.
- *description*: *String [1]*
  Detailed description about the information that a data source entity stores.

Operations: N/A

Associations: This metaclass is defined by four associations:

- *has: Wrapper [1]*
  This relation represents the instance which the data source entity must be related to store data retrieved from the data source.
- *has: DataSourceEntityLink [0..*]*
  This relation represents that an instance of the «DataSourceEntity» metaclass may be related with one or more than one other instances of «DataSourceEntity» metaclass. This metaclass will be detailed in section 2.4.
- *has: TransformationContext [1..*]*
  This relation represents the dependency of this block of the metamodel with the testing metamodel, being able to generate business rules trough derivations later.
- *has: DataSourceAttribute [1..*]*
  This relation represents that an instance of a «DataSourceEntity» metaclass may have one or more instances of the « DataSourceAttribute » metaclass. This metaclass is defined in detail in the section 2.5.

Constraints: N/A

2.4.«DATASOURCELINK» METACLASS

Description: This metaclass allows to represent the relationships that may exists between the instances of the «DataSourceEntity» metaclass.

Generalization: N/A

Attributes: This metaclass is defined by one attributes:

- *linkType: Type [1]*
  Short and concise description with which users can uniquely identify the data source entity. The type "Type" defines the different kind of existing links.

Operations: N/A

Associations: This metaclass is defined by one associations:

- *has: DataSourceEntity [0..*]*
  This relation represents that an instance of the «DataSourceEntity» metaclass may be related with one or more than one other instances of «DataSourceEntity» metaclass.

Constraints: N/A

2.5.«DATASOURCEATTRIBUTE» METACLASS

Description: This metaclass allows to represent the definition of the properties that an instance of the «DataSourceEntity» metaclass has.

Generalization: N/A

Attributes: This metaclass is defined by two attributes:

- *name: String [1]*
  Short and concise description with which users can uniquely identify the attribute.
- *description*: *String [1]*
  Detailed description about the information that an instance of the «DataSourceAttribute» metaclass provides.

Operations: N/A

Associations: This metaclass is defined by two associations:

- *has: IntegrationPattern [1..*]*
  This relation represents the dependency of this block of the metamodel with the testing metamodel, being able to generate business rules trough derivations later.
- *has: TransformationPattern [1..*]*
  This relation represents the dependency of this block of the metamodel with the testing metamodel, being able to generate business rules trough derivations later.

Constraints: N/A

3. VIRTUAL GRAPH METAMODEL

This section describes the Virtual Metamodel and all its components. Figure V-4 shows the metaclases that compose this metamodel and all their relationships. Also, with a low transparency level, it is shown the relationships with metaclases that belong to other blocks of the metamodel.

As mentioned before, this block of the global metamodel allows the user to design the conceptual data model that represents the reconciled solution to achieve according to the Entity Reconciliation problem, as a virtual graph structure.

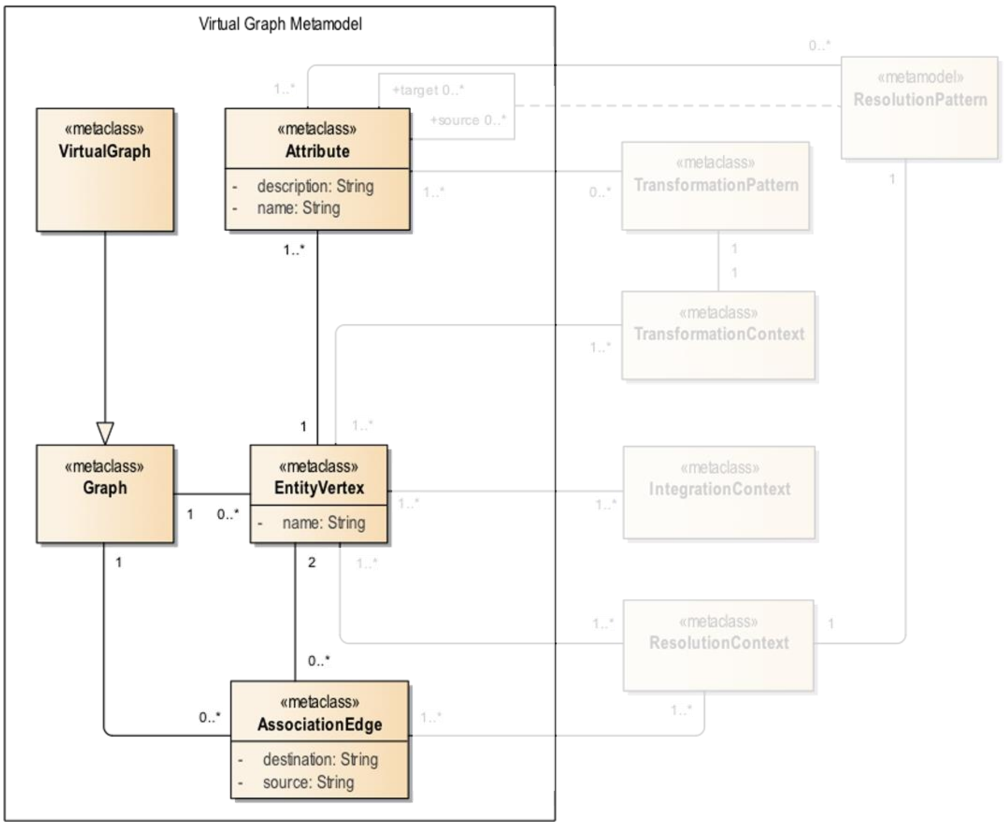


Figure V-4. Virtual Graph Metamodel

This metamodel is composed of five main metaclasses: «Graph», «EntityVertex», «Attribute», «AssociationEdge» and «VirtualGraph». Now, it will be defined each metaclass in detail, for that, it will be exposed their descriptions, generalizations, attributes, operations, associations and constraints. This metamodel is an extended version of the one presented by Mazanek (2011).

3.1.«Graph» Metaclass

Description: This metaclass allows the user to represent a graph structure being an instance of this, the structure where the final solution after the Entity Reconciliation process will be stored.

Generalization: N/A

Attributes: N/A

Operations: N/A

Associations: This metaclass is defined by two associations:

- *has: EntityVertex [0..*]*
  This relation represents that an instance of the «Graph» may be composed of none or more instance of the «EntityVertex» metaclass. This class will be detailed in the section 3.2.

- *has: AssociationEdge [0..*]*
  This relation represents that an instance of the «Graph» may be composed of none or more instance of the «AssociationEdge» metaclass. This class will be detailed in the section 3.4.

Constraints: N/A

3.2.«EntityVertex» Metaclass

Description: This metaclass allows the user to represent the vertexes that will compose the graph structure. The instances of this metaclass will represent the types of entities of the model designed by the user.

Generalization: N/A

Attributes: This metaclass is defined by two attributes:

- *name: String [1]*
  Short and concise description with which users can uniquely identify the vertex.

Operations: N/A

Associations: This metaclass is defined by six associations:

- *has: Graph [1]*
  This relation represents that an instance of the «EntityVertex» metaclass may be only part of one graph.
- *has: AssociationEdge [0..*]*
  This relation represents that an instance of the «EntityVertex» metaclass may be related through none or more instances of «AssociationEdge» metaclass.
- *has: Attribute [1..*]*
  This relation represents that an instance of the «EntityVertex» metaclass may be described by one or more instances of the «Attribute» metaclass.
- *has: TransformationContext [1..*]*
  This relation represents the dependency of this block of the metamodel with the testing metamodel, being able to generate business rules trough derivations later.
- *has: IntegrationContext [1..*]*
  This relation represents that an instance of the «EntityVertex» metaclass may be related through none or more instances of «AssociationEdge» metaclass.

- *has: ResolutionContext [1..*]*
  This relation represents that an instance of the «EntityVertex» metaclass may be related through none or more instances of «AssociationEdge» metaclass.

Constraints: N/A

3.3.«ATTRIBUTE» METACLASS

Description: This metaclass allows to represent the definition of the properties that an instance of the «DataSourceEntity» metaclass has.

Generalization: N/A

Attributes: This metaclass is defined by two attributes:

- *name: String [1]*
  Short and concise description with which users can uniquely identify the attribute.
- *description*: *String [1]*
  Detailed description about the information that an instance of the «Attribute» metaclass provides.

Operations: N/A

Associations: This metaclass is defined by three associations:

- *has: EntityVertex [1]*
  This relation represents that an instance of the «Attribute» metaclass may only be part of an instance of the «EntityVertex» metaclass.
- *has: ResolutionPattern [0..*]*
  This relation represents the dependency of this block of the metamodel with the testing metamodel, being able to generate business rules trough derivations later.
- *has: TransformationPattern [0..*]*
  This relation represents the dependency of this block of the metamodel with the testing metamodel, being able to generate business rules trough derivations later.

Constraints: N/A

3.4.«ASSOCIATIONEDGE» METACLASS

Description: This metaclass allows to represent the relationships that may exist between different instances of the «EntityVertex» metaclass has.

Generalization: N/A

Attributes: This metaclass is defined by two attributes:

- *destination: String [1]*
  Instance of the destination «EntityVertex» metaclass.
- *source*: *String [1]*
  Instance of the source «EntityVertex» metaclass.

Operations: N/A

Associations: This metaclass is defined by three associations:

- *has: Graph [1]*
  This relation represents that an instance of the «AssociationEdge» metaclass may only be part of an instance of the «EntityVertex» metaclass.
- *has: EntityVertex [2]*
  This relation represents that an instance of the «AssociationEdge» metaclass must be part related with two instances of the «EntityVertex» metaclass, being one the source and the other one, the destination.
- *has: ResolutionContext [1..*]*
  This relation represents the dependency of this block of the metamodel with the testing metamodel, being able to generate business rules trough derivations later.

Constraints: N/A

3.5.«VIRTUALGRAPH» METACLASS

Description: This metaclass allows to represent a Virtual Graph data structure. This metaclass is modeled as an abstract class that implements the «Graph» metaclass, it will throw an exception in those methods that are not available for a virtual graph and such as: add edges, add vertexes or remove edges among others. Thus, the instantiation of this metaclass produces a virtual graph that will store the entities (and their relationships) that have been reconciled.

Generalization: «Graph» metaclass

Attributes: N/A

Operations: N/A

Associations: N/A

Constraints: N/A

4. TRANSFORMATIONS METAMODEL

This section describes the Transformation Metamodel and all its components. Figure V-5 shows the metaclases that compose this metamodel and all their relationships. Also, with a low transparency level, it is shown the relationships with metaclases that belong to other blocks of the metamodel.

As mentioned before, this block of the global metamodel allows the user to represent the different transformations that the data of the sources must undergo to carry out the entity reconciliation and to be consistent with the instantiation of the «VirtualGraph» metaclass.

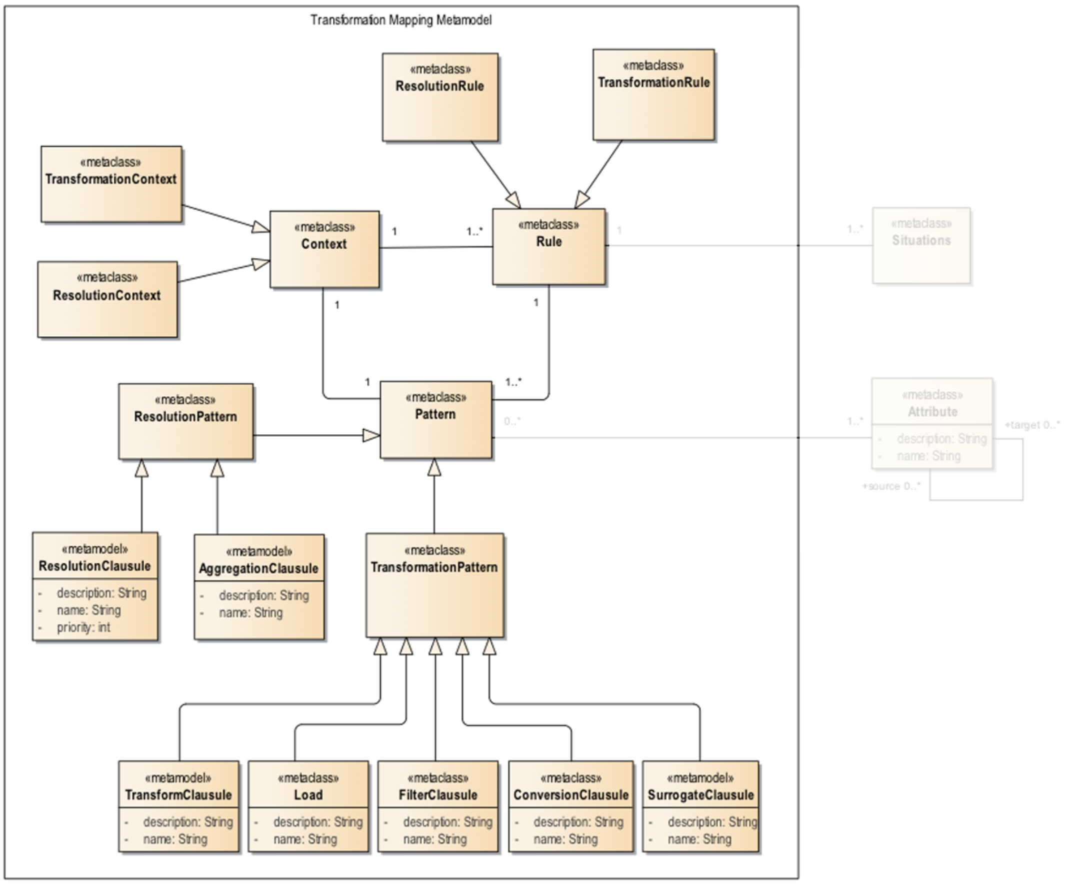

Transformation Mapping Metamodel
«metaclass» ResolutionRule
«metaclass» TransformationRule
«metaclass» TransformationContext
«metaclass» Context
«metaclass» Rule
«metaclass» Situations
«metaclass» ResolutionContext
«metaclass» ResolutionPattern
«metaclass» Pattern
«metaclass» Attribute
description: String
name: String
+target 0..*
+source 0..*
1
1..*
0..*
«metamodel» ResolutionClausule
description: String
name: String
priority: int
«metamodel» AggregationClausule
«metaclass» TransformationPattern
«metamodel» TransformClausule
«metaclass» Load
«metaclass» FilterClausule
«metaclass» ConversionClausule
«metamodel» SurrogateClausule

Figure V-5. Transformation Metamodel

This metamodel is composed of sixteen metaclasses: «Context», «Pattern», «Rule», «TransformationContext», «TransformationRule», «ResolutionContext», «ResolutionRule», «ResolutionPatter», «ResolutionClausule», «AggregationClausule», «TransformationPattern», «TransformationClausule», «Load», «FilterClausule», «ConversionClausule» and «SurrogateClausule». Now, it will be defined each metaclass in detail, for that, it will be exposed their descriptions, generalizations, attributes, operations, associations and constraints. Some of the selected operations have been chosen taking the criteria presented by Trujillo & Luján-Mora, (2003).

4.1.«CONTEXT» METACLASS

Description: This metaclass allows to represent the connections between the types of entities of the «DataSource» metaclass instances and the types of entities of the «VirtualGraph» metaclass instances. These connections impose conditions to be fulfilled to project the entities of the data sources to the entities of the reconciled solution.

Generalization: N/A

Attributes: N/A

Operations: N/A

Associations: This metaclass is defined by five associations:

- *has: Rule [1..*]*
  This relation represents that an instance of the «Context» metaclass may be part of one or more instances of the «Rule» metaclass.
- *has: Pattern [1]*
  This relation represents that an instance of the «Context» metaclass may only be part of an instance of the «Pattern» metaclass.
- *has: DataSourceEntity [1..*]*
  This relation represents the dependency of this block of the metamodel with the data source metamodel, being able to generate business rules trough derivations later.
- *has: EntityVertex [1..*]*
  This relation represents the dependency of this block of the metamodel with the virtual graph metamodel, being able to generate business rules trough derivations later.
- *has: Attribute [1..*]*
  This relation represents the dependency of this block of the metamodel with the virtual graph metamodel, being able to generate business rules trough derivations later.

Constraints: N/A

4.2.«RESOLUTIONCONTEXT» METACLASS

Description: This metaclass allows to represent the connections between the types of entities of the «DataSource» metaclass instances and the types of entities of the «VirtualGraph» metaclass instances for the resolution process.

Generalization: «Context» metaclass

Attributes: N/A

Operations: N/A

Associations: N/A

Constraints: N/A

4.3.«TRANSFORMATIONCONTEXT» METACLASS

Description: This metaclass allows to represent the connections between the types of entities of the «DataSource» metaclass instances and the types of entities of the «VirtualGraph» metaclass instances for the transformation process.

Generalization: «Context» metaclass

Attributes: N/A

Operations: N/A

Associations: N/A

Constraints: N/A

4.4.«PATTERN» METACLASS

Description: The instantiation of this metaclass allows to impose the conditions that lead the actions of the Entity Reconciliation process.

Generalization: N/A

Attributes: N/A

Operations: N/A

Associations: This metaclass is defined by four associations:

- *has: Rule [1]*
  This relation represents that an instance of the «Pattern» metaclass may only be part of an instance of the «Rule» metaclass.
- *has: Context [1]*
  This relation represents that an instance of the «Pattern» metaclass may only be part of an instance of the «Context» metaclass.
- *has: Attribute [0..*]*
  This relation represents the dependency of this block of the metamodel with the data source metamodel, being able to generate business rules trough derivations later.

Constraints: N/A

4.5.«RESOLUTIONPATTERN» METACLASS

Description: The instantiation of this metaclass allows to impose the conditions that lead the actions of the Entity Reconciliation process in the resolution step.

Generalization: «Pattern» metaclass

Attributes: N/A

Operations: N/A

Associations: N/A

Constraints: N/A

4.6.«RESOLUTIONCLAUSULE» METACLASS

Description: The instantiation of this metaclass allows to impose the conditions that lead the actions of the Entity Reconciliation process in the resolution step, stablishing the level of priority between data sources.

Generalization: «ResolutionPattern» metaclass

Attributes: This metaclass is defined by three attributes:

- *name: String [1]*
  Short and concise description with which users can uniquely identify this operation.
- *description*: *String [1]*
  Detailed description about the process that this operation must follow.
- *priority: int [1]*
  This attribute defines the level of priority between data sources.

Operations: N/A

Associations: N/A

Constraints: N/A

4.7.«AGGREGATIONCLAUSULE» METACLASS

Description: The instantiation of this metaclass allows to impose the conditions that lead the actions of the Entity Reconciliation process in the resolution step, making a concatenation of all the attributes related.

Generalization: «ResolutionPattern» metaclass

Attributes: This metaclass is defined by two attributes:

- *name: String [1]*
  Short and concise description with which users can uniquely identify this operation.
- *description*: *String [1]*
  Detailed description about the process that this operation must follow.

Operations: N/A

Associations: N/A

Constraints: N/A

4.8.«TransformationPattern» Metaclass

Description: The instantiation of this metaclass allows to impose the conditions that lead the actions of the Entity Reconciliation process in the transformation step.

Generalization: «Pattern» metaclass

Attributes: N/A

Operations: N/A

Associations: N/A

Constraints: N/A

4.9.«TransformClausule» Metaclass

Description: The instantiation of this metaclass allows to impose the conditions that lead the actions of the Entity Reconciliation process in the resolution step, making a transformation of a source attribute to another one.

Generalization: «TransformationPattern» metaclass

Attributes: This metaclass is defined by two attributes:

- *name: String [1]*
  Short and concise description with which users can uniquely identify this operation.
- *description*: *String [1]*
  Detailed description about the process that this operation must follow.

Operations: N/A

Associations: N/A

4.10. «LOADCLAUSULE» METACLASS

Description: The instantiation of this metaclass allows to impose the conditions that lead the actions of the Entity Reconciliation process in the resolution step, transferring the content of an attribute to another one.

Generalization: «TransformationPattern» metaclass

Attributes: This metaclass is defined by two attributes:

- *name: String [1]*
  Short and concise description with which users can uniquely identify this operation.
- *description*: *String [1]*
  Detailed description about the process that this operation must follow.

Operations: N/A

Associations: N/A

4.11. «FILTERCLAUSULE» METACLASS

Description: The instantiation of this metaclass allows to impose the conditions that lead the actions of the Entity Reconciliation process in the resolution step, making a filter of the content of one or more attributes.

Generalization: «TransformationPattern» metaclass

Attributes: This metaclass is defined by two attributes:

- *name: String [1]*
  Short and concise description with which users can uniquely identify this operation.
- *description*: *String [1]*
  Detailed description about the process that this operation must follow.

Operations: N/A

Associations: N/A

4.12. «CONVERSIONCLAUSULE» METACLASS

Description: The instantiation of this metaclass allows to impose the conditions that lead the actions of the Entity Reconciliation process in the resolution step, making a conversion of a source attribute two more than one attributes.

Generalization: «TransformationPattern» metaclass

Attributes: This metaclass is defined by two attributes:

- *name: String [1]*
  Short and concise description with which users can uniquely identify this operation.
- *description*: *String [1]*
  Detailed description about the process that this operation must follow.

Operations: N/A

Associations: N/A

4.13. «SURROGATECLAUSULE» METACLASS

Description: The instantiation of this metaclass allows to impose the conditions that lead the actions of the Entity Reconciliation process in the resolution step, generating an unique surrogate key.

Generalization: «TransformationPattern» metaclass

Attributes: This metaclass is defined by two attributes:

- *name: String [1]*
  Short and concise description with which users can uniquely identify this operation.
- *description*: *String [1]*
  Detailed description about the process that this operation must follow.

Operations: N/A

Associations: N/A

4.14. «RULE» METACLASS

Description: The instantiation of this metaclass represents the elements of the rules that constitute this metamodel. It is divided in three mail types: transformation, resolution rules and integration rules (the last ones, defined in the section 5 of this chapter).

Generalization: «Context» metaclass

Attributes: N/A

Operations: N/A

Associations: N/A

Constraints: N/A

4.15. «TRANSFORMATIONRULE» METACLASS

Description: The instantiation of this metaclass represents the elements of the rules that constitute this metamodel in the transformation step.

Generalization: «Rule» metaclass

Attributes: N/A

Operations: N/A

Associations: N/A

Constraints: N/A

4.16. «RESOLUTIONRULE» METACLASS

Description: The instantiation of this metaclass represents the elements of the rules that constitute this metamodel in the resolution step.

Generalization: «Rule» metaclass

Attributes: N/A

Operations: N/A

Associations: N/A

Constraints: N/A

5. TESTING METAMODEL

This section describes the Virtual Metamodel and all its components. Figure V-6 shows the metaclases that compose this metamodel and all their relationships. Also, with a low transparency level, it is shown the relationships with metaclases that belong to other blocks of the metamodel.

As mentioned before, this block of the global metamodel allows the user to represent the testing objectives for the entity reconciliation application in the early stages of the development. The test models can be focused on different level, as unit testing or integration testing.

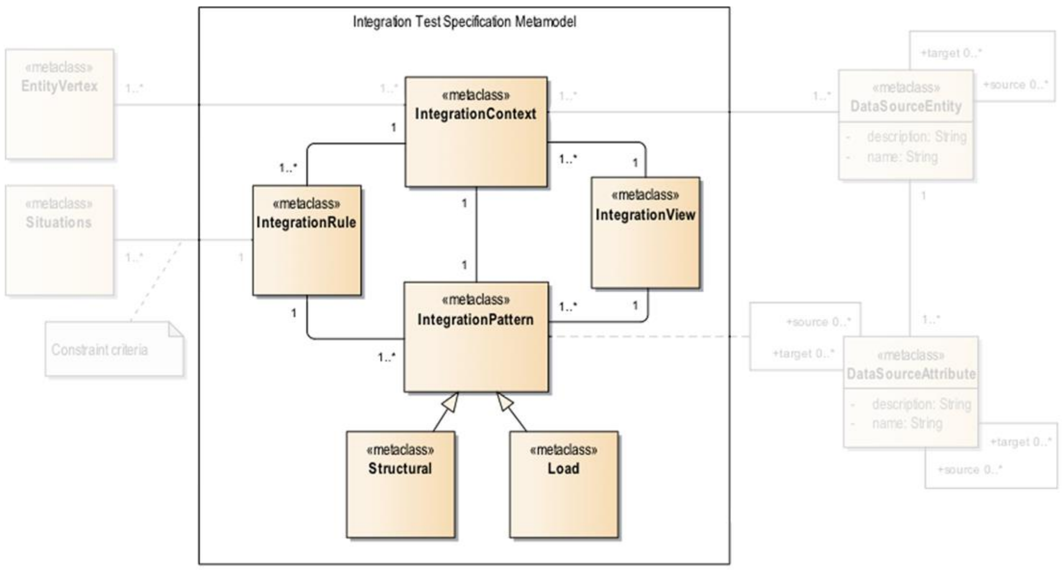

Integration Test Specification Metamodel
«metaclass» EntityVertex
«metaclass» Situations
Constraint criteria
«metaclass» IntegrationContext
«metaclass» IntegrationRule
«metaclass» IntegrationView
«metaclass» IntegrationPattern
«metaclass» Structural
«metaclass» Load
«metaclass» DataSourceEntity
- description: String
- name: String
«metaclass» DataSourceAttribute
- description: String
- name: String
+target 0..*
+source 0..*
1..*
1

Figure V-6. Testing Metamodel

This metamodel is composed of six main metaclasses: «IntegrationContext», «IntegrationView», «IntegrationPattern», «Structural», «Load» and «IntegrationRule». Now, it will be defined each metaclass in detail, for that, it will be exposed their descriptions, generalizations, attributes, operations, associations and constraints. It is important to note that «IntegrationContext», «IntegrationPattern» and «IntegrationRule» herigate from the «Context», «Pattern» and «Rule» metaclasses of Transformation Metamodel, although for getting a better clarity of the vision of the metamodel, they have not been added to the figure V-6.

5.1.«IntegrationContext» Metaclass

Description: This metaclass allows to represent the connections between the types of entities of the «DataSource» metaclass instances and the types of entities of the «VirtualGraph» metaclass instances. These connections impose conditions to be fulfilled to project the entities of the data sources to the entities of the reconciled solution.

Generalization: «Context» metaclass

Attributes: N/A

Operations: N/A

Associations: This metaclass is defined by five associations:

- *has: IntegrationRule [1..*]*
  This relation represents that an instance of the «IntegrationContext» metaclass may be part of one or more instances of the «IntegrationRule» metaclass.

- *has: IntegrationPattern [1]*
  This relation represents that an instance of the «IntegrationContext» metaclass may only be part of an instance of the «IntegrationPattern» metaclass.
- *use: IntegrationView [1]*
  This relation represents that an instance of the «IntegrationContext» metaclass may only use one instance of the «IntegrationView» metaclass.
- *has: DataSourceEntity [1..*]*
  This relation represents the dependency of this block of the metamodel with the data source metamodel, being able to generate business rules trough derivations later.
- *has: EntityVertex [1..*]*
  This relation represents the dependency of this block of the metamodel with the virtual graph metamodel, being able to generate business rules trough derivations later.

Constraints: N/A

5.2.«INTEGRATIONVIEW» METACLASS

Description: This metaclass allows to connects a subset of instances of the «EntityVertex» metaclass involved in an instance of the «IntegrationContext» metaclass. A view is focused on a part of the projection defined by means of an instance of the «IntegrationContext» metaclass.

Generalization: N/A

Attributes: N/A

Operations: N/A

Associations: This metaclass is defined by two associations:

- *has: IntegrationContext [1]*
  This relation represents that an instance of the «IntegrationView» metaclass may only be part of an instance of the «IntegrationContext» metaclass.
- *has: IntegrationPattern [1]*
  This relation represents that an instance of the «IntegrationView» metaclass may only be part of an instance of the «IntegrationPattern» metaclass.

Constraints: N/A

5.3.«INTEGRATIONPATTERN» METACLASS

Description: The instantiation of this metaclass allows to impose the conditions that lead the actions of the Entity Reconciliation process.

Generalization: «Pattern» metaclasses

Attributes: N/A

Operations: N/A

Associations: This metaclass is defined by four associations:

- *has: IntegrationRule [1]*
  This relation represents that an instance of the «IntegrationPattern» metaclass may only be part of an instance of the «IntegrationRule» metaclass.
- *has: IntegrationView [1]*
  This relation represents that an instance of the «IntegrationPattern» metaclass may only be part of an instance of the «IntegrationView» metaclass.
- *has: IntegrationContext [1]*
  This relation represents that an instance of the «IntegrationPattern» metaclass may only be part of an instance of the «IntegrationContext» metaclass.
- *has: DataSourceAttribute [0..*]*
  This relation represents the dependency of this block of the metamodel with the data source metamodel, being able to generate business rules trough derivations later.

Constraints: N/A

5.4.«STRUCTURAL» METACLASS

Description: This metaclass allows to impose conditions to be fulfilled in order to create new entities and relationship into the reconciled solution. It is a type of Integration Pattern.

Generalization: «IntegrationPattern» metaclass.

Attributes: N/A

Operations: N/A

Associations: N/A

Constraints: N/A

5.5.«LOAD» METACLASS

Description: This metaclass allows to establish conditions to be fulfilled to derive the value of the attributes of the entities that belong to the reconciled solution from the data sources. It is a type of Integration Pattern.

Generalization: «IntegrationPattern» metaclass.

Attributes: N/A

Operations: N/A

Associations: N/A

Constraints: N/A

5.6.«INTEGRATIONRULE» METACLASS

Description: The instantiation of this metaclass represents the elements of the integration rules that constitute this metamodel.

Generalization: «Rule» metaclasses

Attributes: N/A

Operations: N/A

Associations: This metaclass is defined by three associations:

- *has: IntegrationContext [1]*
  This relation represents that an instance of the «IntegrationRule» metaclass may only be part of an instance of the «IntegrationContext» metaclass.
- *has: IntegrationPattern [1..*]*
  This relation represents that an instance of the «IntegrationRule» metaclass may be part of one or more instances of the «IntegrationPattern» metaclass.
- *has: Situations [1..*]*
  This relation represents that an instance of the «IntegrationRule» metaclass may be part of one or more instances of the «Situations» metaclass. This metaclass represents the test requirements that will be generated from an instance of the «IntegrationRule» metaclass.

Constraints: This metaclass is defined by one constratint:

- *Constraint criteria.*
  It will represent the criteria that will be applied to an instance of the «IntegrationRule» metaclass for obtaining the test cases, for example, MCDC (Blanco et al., 2012a).

6. CONCLUSIONS

Throughout this chapter, the metamodels necessary to address the definition of the Entity Reconciliation model have been presented in a formal way within the MaRIA Framework. This metamodel is composed of four main blocks: virtual graph metamodel, data source metamodel, transformation metamodel and the testing metamodel, all related between them.

This formal definition and representation, materialized through the UML class diagrams notation, allow to establish derivations that will let to be entity reconciliation problem modeled with the MaRIA metamodel systematically tested. These derivation rules will be defined in Chapter VI.

CHAPTER VI
DERIVATIONS

CHAPTER VI. DERIVATIONS

Chapter IV defined the framework on which we have worked in this doctoral thesis composed of three main pillars: MaRIA Process, Model-Driven Approach and the MaRIA tool. In this context, Chapter IV, described the set of processes that the MaRIA Process defines to be added to any software methodology in its requirement, analysis and testing phases. Chapter V described the MaRIA metamodel, one of the elements that compose the second pillar of MaRIA Framework, the Model-Driven Approach.

This chapter, presents the rules that have been defined to make the instantiation of the metamodel defined in Chapter IV testable and also, the derivation mechanisms used to automate the process of creation of these rules. Finally, this chapter synthesizes a set of conclusions.

1. INTRODUCTION

Derivation mechanisms, conforms the second element that completes the second pillar of the MaRIA Framework, the Model-Driven Approach. Figure VI-1 shows the position that this chapter is focusing inside the Framework.

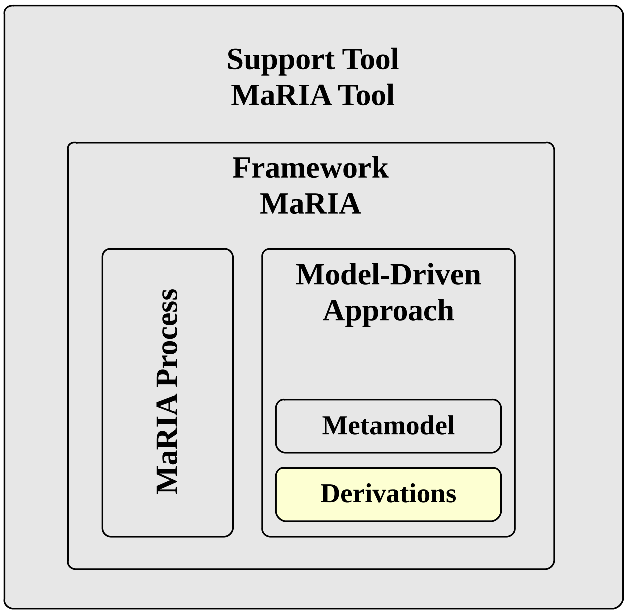


Figure VI-1. Framework MaRIA Metamodel

Once developed this second pillar, all the elements for the modeling of an entity reconciliation problem during the development of a software product in any software methodology are covered.

The proposal presented in this doctoral thesis proposes a set of Model to Text (M2T) transformations to automatically, generate the test requirements of the application that will consume the data that has been generated after the reconciliation process.

There are great variety of languages that allow the automatic generation of code from models. Some of the most extended are: Epsilon (Frankel, 2003), a model management platform that provides transformation languages for model-to-model, model-to-text, update-in-place, migration and model merging transformations. MOF Model to Text Transformation Language (Mof2Text or MOFM2T) (Omg, 2008), defined by the Object Management Group (OMG) as a standard for expressing M2T transformations. Java Emitter Templates (JET) (Eclipse, 2016a) used for generating text from Eclipse Modeling Framework (EMF) based models. MOFScript (Oldevik et al., 2005) is a direct result of the OMG RFP for a M2T transformation language. It works with any Meta-Object Facility (MOF) based model and it is very influenced by QVT (Omg, 2008). Basically, it is an imperative language which supports all primitive types and abstract data types.

However, this doctoral thesis has bet for the use of Text Template Transformation Toolkit (T4) (Vasudevan and Tratt, 2011) because it is the language defined by Microsoft for M2T transformations in DSL-Tools (tool which has been used for the development of the support tool of the MaRIA methodology).

DSL-Tools offers a generator component called "TextTemplating" that allow the generation of code using templates called "T4 templates". Next sections, will describe the architecture of templates, the typical execution process of a transformation and the main elements of a T4 template following the indications of the manual for domain-specific development with Visual Studio DSL Tools presented by Cook et al., (2007).

1.1.ARCHITECTURE OF TEMPLATES

T4 templates are written in C# o Visual Basic language and they have a set of expressions that are exclusives for this technology.

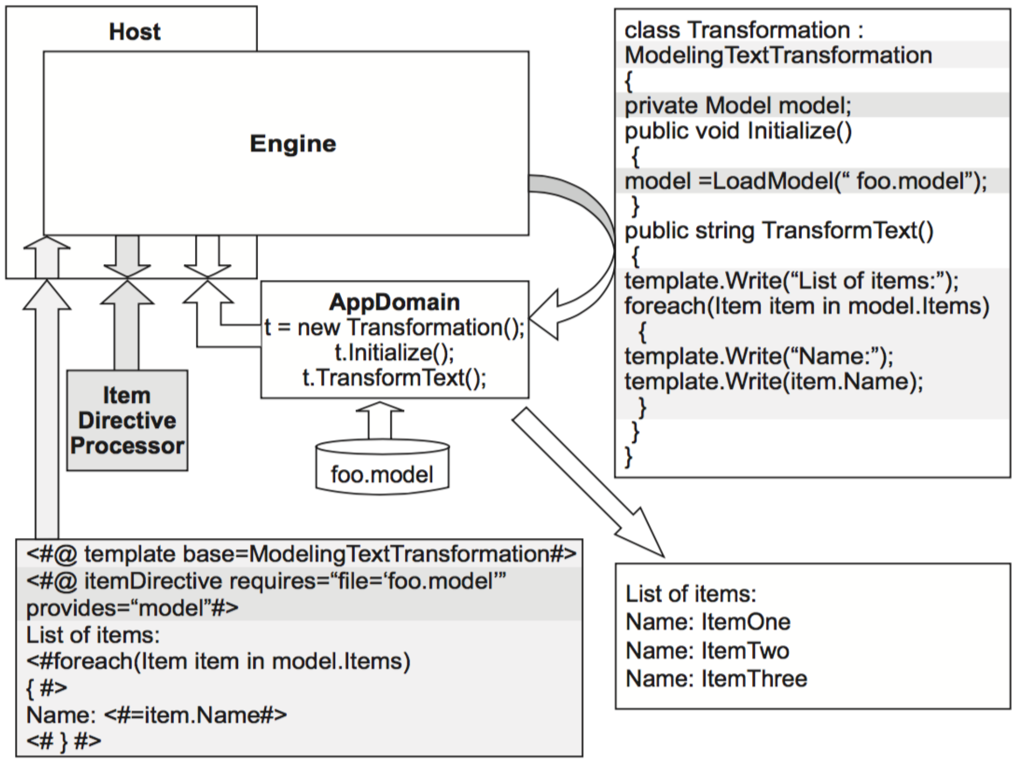
The modeling and visualization software of Visual Studio provide a set of tools that allow the generation of code of any type of files taking as a base the content defined in the templates.

The transformation of the templates is a process that is performed in two steps the first one, the engine generates a temporal class called "transformation class" that contains the code generated by the control blocks and the directives. The second one take place when the engine is going to compile and execute the transformation class previously generated for creating the output file.

The most important components of the tool for generating T4 templates (marked with blue text in Figure VI-2) of the modeling and visualization software of Visual Studio are:

- Item Directive Processor. It represents the classes that manage text template directives.
- Host. It is the interface between the engine and the user environment. Visual Studio is a host of the text transformation process.

- Motor. It controls the process of transforming text templates and core of the template system. It oversees processing the templates and creating an output.



```
t = new Transformation();
t.Initialize();
t.TransformText();
```

```
class Transformation :
ModelingTextTransformation
{
private Model model;
public void Initialize()
 {
model =LoadModel(" foo.model");
 }
public string TransformText()
 {
template.Write("List of items:");
foreach(Item item in model.Items)
  {
template.Write("Name:");
template.Write(item.Name);
  }
 }
}
```

```
<#@ template base=ModelingTextTransformation#>
<#@ itemDirective requires="file='foo.model'"
provides="model"#>
List of items:
<#foreach(Item item in model.Items)
{ #>
Name: <#=item.Name#>
<# } #>
```

```
List of items:
Name: ItemOne
Name: ItemTwo
Name: ItemThree
```

Figure VI-2. Text Template Architecture (Cook et al., 2007)

Once presented the most important elements of the process of transformation, next section will describe the typical execution sequence of the text templates.

1.2. EXECUTION SEQUENCE OF TEXT TEMPLATES

Taking as reference the architecture of Figure VI-2, the execution sequence of a text templates usually follows these steps:

- The host reads in a template file from disk.
- The host instantiates a template engine.
- The host passes the template text to the engine along with a callback.
- The engine parses the template, finding standard and custom directives and control blocks.
- The engine asks the host to load the directive processors for any custom directives it has found.
- The engine produces in memory the code for the skeleton of a Transformation class, derived ultimately from *TextTransformation*. This template has specified the derived *ModelingTextTransformation*.
- The engine gives directive processors a chance to contribute code to the *Transformation class* for both class members and to run in the body of the *Initialize()* method.

- The engine adds the content of class feature blocks as members to the *Transformation class*, thus appearing to allow methods and properties to be "added to the template."
- The engine adds boiler plate inside *Write* statements and the contents of other standard control blocks to the *TransformText()* method.
- The engine compiles the *Transformation class* into a temporary .NET assembly.
- The engine asks the host to provide an *AppDomain* in which to run the compiled code.
- The engine instantiates the *Transformation class* in the new *AppDomain* and calls the *Initialize()* and *TransformText()* methods on it via .NET remoting.
- The *Initialize()* method uses code contributed by directive processors to load data specified by the custom directives into the new *AppDomain*.
- The *TransformText()* method writes boiler plate text to its output string, interspersed with control code from regular control blocks and the values of expression control blocks.
- The output string is returned via the engine to the host, which commits the generated output to disk.

1.3. Elements of a T4 Template

The principal elements of a T4 Template may be divided in: directives, text and control blocks and utility methods. Next subsections present a brief description of these elements:

Directives

Directives provide instructions to the text template creation engine on how the transformation code and output file should be generated. The directives must be the first elements in a template. There are five types of directives that need to be defined: template, output, assembly, import and include.

```
<#@ DirectiveName [AttributeName = "AttributeValue"] ... #>
```

Figure VI-3. Declaration of a Directive

- **Template**: the most usual is to start the templates with this directive since it is the one that specifies how to process the template. It contains five main properties to define: language, compiler options, culture, debug and line pragmas.
  - **Language**: specifies the language to be used as source code in the templates. By default, it is C#.
  - **Compiler options**: are compile options for customizing compiler behavior.
  - **Culture**: is an attribute used to know the cultural reference of a file when it is generated.
  - **Debug**: if this attribute is set as true value it allows the debugging of the code and if it is false does not allow it.
  - **LinePragmas**: allows the compiler to display in debug mode the lines of errors either the template or generated code.

```
<#@ template [language="VB"] [hostspecific="true|TrueFromBase"]
[debug="true"] [inherits="templateBaseClass"] [culture="code"]
[compilerOptions="options"][visibility="internal"] [linePragmas="false"] #>
```

Figure VI-4. Declaration of a Template directive

- **Output:** this directive defines the file extension to be generated as the output of the template execution. It also allows to change the encoding of the output file. It contains two main properties: extension and encoding
  - Extension: represents the extension of the generated file, by default, is represented as ".cs" but may be any type of needed file (".json", ".java" or ".html" among others).
  - Encoding: represents the encoding that the output file will have when it is generated ("utf-8" or "us-ascii" among others)

```
<#@ output extension="fileNameExtension" [encoding="encoding"] #>
```

Figure VI-5. Declaration of an Output directive

- **Assembly:** represents a reference to an assembly code so that the template can use the types.

```
<#@ assembly name="[assembly strong name|assembly file name]" #>
```

Figure VI-6. Declaration of an Assembly directive

- **Import:** represents the equivalent translation in C# to "using" or "import" in Java language.

```
<#@ import namespace="System.IO" #>
```

Figure VI-7. Declaration of an Import directive

- **Include:** allows access from the template that contains the include directive to the templates referenced by this directive. It is used to be able to reuse code between templates.

```
<#@ include file="filePath" [once="true"] #>
```

Figure VI-8. Declaration of an Include directive

TEXT BLOCKS

It allows inserting text in the output file. It does not require to have a certain format or make use of functions, what is written will be inserted as plain text in the output of the template.

CONTROL BLOCKS

The control blocks allow writing the template and being able to change the application context of the instructions. This will be very helpfully to create any type of template.

There are three type of control blocks. There are differentiated by the opening brackets as well as by the functionality they allow to perform. Those are: standard control block, expression control block and functions control block.

- **Standard control block (<# .... #>):** contain the instructions and the blocks can be opened and closed in the middle of sentences (if or for structures among others).

```
<# if (test) { #>

      // Do something

<# } #>
```

Figure VI-9. Standard Control Block

- **Expression control block (<# = #>):** is used for code that returns a string that we want to be in the output file.

```
<# string imports = “using System.Collections.Generic”;

      imports += “\nusing System”;

<#= imports #>
```

Figure VI-10. Expression Control Block

- **Class Feature control block (<#+ ... #>):** is used to define auxiliary functions or functions that will be reuse within a template. There can only be one block of this type in each template. Everything in this block will be static.

```
<#+ public void GenerateEmptyClass(string name) { #>
   public partial class <#= name #> {
      // Some class content
   }
<#+ } #>
```

Figure VI-11. Class Feature Control Block

UTILITY METHODS

Utility methods allow to have a minimum of tools for writing T4 templates. These methods will always be accessible from the templates and will not have to be imported. They can be briefly classified in: write, bleeding and warning and error methods.

- **Write Methods:** they are Write() y WriteLine() methdos that allow write text inside the code.

```
<#
   int i = 10;
   while (i-- > 0) {
      writeLine((i.toString()));
   }
#>
```

Figure VI-12. Write Method

- **Bleeding Methods:** these methods are used to format the output generated by the template. It can also be done with the "\t" sentence.
    - CurrentIndent: Shows the bleeding that is currently being carried.
    - IndentLenghts: list of bleedings that have been added.
    - PushIndents: adds a bleeding.
    - PopIndents: removes a bleeding.
    - ClearIndents: Cleans the stack of bleedings.

- **Warning and Error Methods:** are methods that allow to display errors in the Visual Studio bug list if there were any.

```
<#
   try {
      string str = null;
      writeLine(str.Length.toString());
   } catch (Exception e) {
      Error(e.Message);
   }
#>
```

Figure VI-13. Error Method

Once a clear vision of how the transformations must defined and all the components of the DSL-Tools for this purpose have been presented, the next section will present how the transformations have been defined for the proposal of this doctoral thesis.

2. INTEGRATION RULES DEFINITION

The integration rules, which are statements that define or constraint the business structure or the business behavior (Hay and Healy, 2000), have been used in other approaches focused on testing database applications, such as (Blanco et al., 2012) and (Willmor et al., 2006). On the other hand, as the integration rules are based on the system specification, they could also be used to generate some implementation of the ER application.

These integration rules are specially focused on the subsequent derivation of test coverage items that guide the creation of the test data sources and the test reconciled solution.

As stated in Chapter IV, integration rules may be defined by two ways: load and structural rules. In this sense and to delimit the scope of this Doctoral Thesis, it will be only covered the integration testing of the model, what means, the load transformations of data and the data structure where the solution will be stored. This concept has been taken from the black box testing method (Beizer, 1995).

The following subsections aims to present the patterns that allow expressing the integration context, the integration context view, as well as the integration patterns of each type of integration rule (load and structural).

2.1. INTEGRATION CONTEXTS AND INTEGRATION CONTEXT VIEWS

To describe the integration context and the integration context views of an integration rule, it is necessary to define the concept path that is used in their construction.

A *path (P)* is a set of one or more types of entities (instances of the metaclasses *DataSourceEntity* and *EntityVertex*) and/or types of relationships (instances of the metaclasses *AssociateEdge* and *DataSourceEntityLink*) R_1, R_2, …, R_n, where each pair (R_i, R_{i+1}) is directly connected via some attributes in the predicate $q_{i,i+1}()$:

$$\textit{Path P is } R_1\ [q_{1,2}()]\ R_2\ [q_{2,3}()]\ \ldots\ [q_{n-1,n}()]\ R_n$$

Each $q_{i,i+1}()$ can contain arithmetic and logical expressions and functions, which involve attributes of R1, R2, …, R_{i+1}.

The definition of the concept path suggests the redefinition of the *integration context*, the *integration context view*, as well as the *context entities*, *context relationships* and *context attributes* in terms of this concept, as explained below:

An *integration context (IC)* is a set of one or more *paths* P_1, P_2, …, P_m that define the connections between the data source models and the reconciled solution model that are involved in a test condition:

$$\textit{Integration context IC is } P_1, P_2, \ldots, P_m$$

If an *integration context* is formed by only one *path*, it can be defined directly by:

Integration context IC is $R_1\ [q_{1,2}()]\ R_2\ [q_{2,3}()]\ \ldots\ [q_{n-1,n}()]\ R_n$

An *integration context view* or *view*, for short, *(V_{IC})* of an integration context *IC* is a subset $R_j, R_{j+1}, R_{j+2}, \ldots, R_k$ of a path *P* of *IC*, where each pair (R_i, R_{i+1}) (i=j..k-1) is directly connected via the predicate defined in *P*:

Integration context view V_{IC} *is* $R_j\ []\ R_{j+1}\ []\ \ldots\ []\ R_k$ *of IC.P*

A *context entity* is a type of entity *R* of a *path P* of an *integration context IC* denoted by *IC.R*. If *R* is not unique in *IC* it is denoted by *IC.P.R*, where *P* is a path of *IC* that contains *R*. A *context entity* of a *view* V_{IC} of an *integration context IC* is denoted by $V_{IC}.R$.

A *context relationship* is a type of relationship *R* of a *path P* of an *integration context IC* denoted by *IC.R*. If *R* is not unique in *IC* it is denoted by *IC.P.R*, where *P* is a *path* of *IC* that contains *R*. A *context relationship* of a *view* V_{IC} of an *integration context IC* is denoted by $V_{IC}.R$.

A *context attribute* is an attribute *A* of a *context entity* or a *context relationship* of an *integration context IC* denoted by *IC.A*. If *A* is not unique in *IC* it is denoted by *IC.P.R.A* or *IC.R.A*, where *P* is a *path* of *IC* and *R* is a *context entity* of *P* that contains *A*. A *context attribute* of a *view* V_{IC} of an *integration context IC* is denoted by $V_{IC}.A$ or $V_{IC}.R.A$.

2.2. SPECIFICATION OF STRUCTURAL RULES

A structural rule establishes the projection from a *context entity IC.R* (or $V_{IC}.R$) that belongs to a *data source model* to one or several *context entities* and *context relationships* $IC.S_i$ (or $V_{IC}.S_i$) that belong to the *reconciled solution model*. It also establishes one or several conditions on the *context attributes* $IC.S_i.A_j$ (or $V_{IC}.S_i.A_j$) that constrain their values when the new *entities* and *relationships* are created into the current *reconciled solution*.

The projection imposed by the structural rule must be fulfilled by each instance of *IC.R* (or $V_{IC}.R$) that belongs to the unreconciled context domain of *IC* (or the unreconciled view domain of V_{IC}). The integration pattern of a structural rule is described below, using the EBNF notation (Horrocks et al., 2004). The integration pattern of a structural rule is defined as:

structural_rule = *"Each unreconciled" (IC.R |* $V_{IC}.R$*) "generates" gen_cond {"and" gen_cond};*

gen_cond = *"exactly one"* ($IC.S_i$ | $V_{IC}.S_i$) *"with" att_cond;*

att_cond = ($IC.S_i.A_j$ | $V_{IC}.S_i.A_j$) *"="* p_j *{"and"* ($IC.S_i.A_j$ | $V_{IC}.S_i.A_j$) *"="* p_j*};*
where each p_j is a predicate over context *attributes* of *IC.R* (or $V_{IC}.R$) and/or $IC.S_i$ (or $V_{IC}.S_i$).

2.3.SPECIFICATION OF LOAD RULES

A load rule imposes one on several conditions that constrain the value of a *context attribute IC.S.A* that belongs to the *reconciled solution model*, according to one or several *context attributes IC.Ri.Bj* that belong to the *data source models*. The conditions must be fulfilled by each tuple of the *reconciled context domain* of *IC*.

The load rules are classified according to two dimensions. The first dimension indicates whether a load rule establishes preconditions that must be fulfilled before constraining the value of a context attribute (conditional rules), or it does not establish any precondition (non-conditional rules).

The second dimension indicates the types of conditions that constrain the value of the context attributes according to one or several predicates (*IS*, *OR*, *AND*, *XOR* rules). These predicates can be either arithmetical or logical expressions or functions over *context attributes* of the *integration context IC*, as well as constants or *context attributes* of *IC*. The evaluation of the predicates returns a value that fits the type of the *context attribute* constrained or a null value, which indicates that the predicate was not able to reach a concrete value. The following definitions describe the patterns of each category, using the EBNF notation.

A *conditional rule* is a load rule whose *integration pattern* is defined as:

conditional_rule = "If" p "then" rule_pattern;

where *p* is a predicate over *context attributes IC.S.A* and/or *IC.Ri.Bj* whose evaluation returns a boolean value. This predicate defines the preconditions to be fulfilled before constraining the value of the *context attribute IC.S.A* by means of *rule_pattern* (described next).

An *IS rule* is a load rule that constrains the value of a *context attribute IC.S.A*, such that it must be equal to the evaluation of a predicate *p*. The *integration pattern* is defined as:

IS_rule = "Each" IC.S.A "is" p;

An *AND rule* is a load rule that constrains the value of a *context attribute IC.S.A*, such that it must be formed by the union of the evaluations of the predicates p_i that do not return a null value. The integration pattern is defined as:

AND_rule = "It is obligatory that" IC.S.A "is composed of " p_i {"and" p_i};

An *OR rule* is a load rule that constrains the value of a *context attribute IC.S.A*, such that it can be formed by the evaluation of one or several predicates p_i that do not return a null value. The integration pattern is defined as:

OR_rule = "It is permitted that" IC.S.A "is composed of " p_i {"or" p_i };

An *XOR rule* is a load rule that constrains the value of a *context attribute IC.S.A*, such that it must be equal to the evaluation of only one predicate p_i. Each predicate p_i has a different priority n_i that indicates the order in which they are evaluated. *IC.S.A* takes the value of the first predicate p_i that does not return a null value. The integration pattern is defined as:

$$XOR\_rule = prioritization \text{ ``Each'' } IC.S.A \text{ ``is only '' } p_i \{\text{``or'' } p_i \};$$

$$prioritization = p_i \text{ ``has priority'' } n_i \{ p_i \text{ ``has priority'' } n_i \}$$

The integration patterns of non-conditional *IS, AND, OR* and *XOR* rules are directly described by combining *conditional rules* with *IS*, *AND*, *OR* and *XOR* rules defined before respectively.

After defining the test conditions as a set of the integration rules, the test coverage items can be derived by means of applying logic criteria, as stated one of the activities defined in MaRIA process of early testing phase over the conditions imposed by these integration rules.

3. Derivation Mechanisms

This section details the main derivation mechanisms that aims to transform a model created following the MaRIA metamodel in a set of integration rules defined in section 2 of present Chapter.

These transformations take as input the elements identified in the model such as: entities, attributes, data sources or transformations connections among others and generates a business rule conforms to the described previously.

As stated in integrated rules definition section, derivation mechanisms defined will be divided in two main categories: structural rules and load rules.

3.1. STRUCTURAL RULES

The structural rules impose conditions to be fulfilled to create new entities and relationship into the reconciled solution. Thus, the expression of the transformation that will automatically generates the integration rule for structure rules is showed in Figure VI-14.

```
<#
int i = 0;
foreach (DataSource element in getDataSources()) {
      i++;
#>
      Path P<#= i #> is <#= element.Name #>
      <#
      foreach (Rule r in getRules()) {
            printRule(rule.getType())
      }
} #>

Integration context IC_substr(<#= element.Name,1 #>) is P<#= i #>
<# EntityVertex ev = getEntityVertexByEqualsRule() #>
<# TransformationPattern trans = getTransformationPatterByEntityVertex(ev) #>
<# Destination dest = getAssociationEdgeDestinationByEntityVertex(ev) #>
<# Node node = getAssociationEdgeDestinationByDestination(dest) #>
Integration context view V1_substr(<#= element.Name,1 #>) is <#= element.Name
#>[] ev[] of IC_substr(<#= element.Name,1 #>).P<#= i #>
Integration context view V2_substr(<#= element.Name,1 #>) is <#= element.Name
#>[] ev[] trans[] dest[] of IC_substr(<#= element.Name,1 #>).P<#= i #>

Each unreconciled V1_substr(<#= element.Name,1 #>).<#= element.Name #>
generates exactly one V1_substr(<#= element.Name,1 #>).Province  with
V1_substr(<#= element.Name,1 #>).element.Province=V1.getProvince()

Each unreconciled V2_substr(<#= element.Name,1 #>).<#= element.Name #>
generates exactly one V2_ substr(<#= element.Name,1 #>).Dest with V2_
substr(<#= element.Name,1 #>).element.City=V2_substr(<#= element.Name,1
#>).getLocation() and exactly one V2_substr(<#= element.Name,1 #>).Trans with
V2_substr(<#= element.Name,1 #>).Source=V2_substr(<#= element.Name,1 #>).Dest
and V2_substr(<#= element.Name,1 #>).destination=V2_substr(<#= element.Name,1
#>).getProvince()

Each unreconciled IC_substr(<#= element.Name,1 #>).element.Name generates
exactly one IC_substr(<#= element.Name,1 #>).element.Name with IC_substr(<#=
element.Name,1 #>).getMonument()=IC_substr(<#= element.Name,1 #>).getName()
and exactly one IC_(<#= element.Name,1 #>).getEntityVertexBySitedInRule()
with IC_(<#= element.Name,1 #>).
getEntityVertexBySitedInRule().source=IC_substr(<#= element.Name,1
#>).getName() and IC_(<#= element.Name,1 #>).
getEntityVertexBySitedInRule().destination=IC_substr(<#= element.Name,1
#>).getCity()
#>
```

Figure VI-14. Structural Rules Transformation Expression

A concrete example of structural rule instantiation is showed in Figure VI-15.

```
Integration context:
Path P1 is Mosaico
[Equals(province, p_name)] Province
[Belong_to.destination=p_name] Belong_to
[Belong_to.source=c_name and Equals(location, c_name)] City
[Sited_in.destination=c_name] Sited_in
[Sited_in.source=m_name and Equals(name, m_name)] Monument
Integration context IC_M is P1
Integration context views:
Integration context view V1_M is Mosaico[] Province of IC_M.P1
Integration context view V2_M is Mosaico[] Province [] Belong_to [] City of
IC_M.P1
Integration patterns:
Each unreconciled V1_M.Mosaico generates exactly one V1_M.Province
with V1_M.p_name=V1.province
Each unreconciled V2_M.Mosaico generates exactly one V2_M.City with
V2_M.c_name=V2_M.location
and exactly one V2_M.Belong_to with V2_M.source=V2_M.c_name and
V2_M.destination=V2_M.p_name
Each unreconciled IC_M.Mosaico generates exactly one IC_M.Monument with
IC_M.m_name=IC_M.name
and exactly one IC_M.Sited_in with IC_M.Sited_in.source=IC_M.m_name and
IC_M.Sited_in.destination=IC_M.c_name
```

Figure VI-15. Structural Rules Transformation Expression

3.2.Load Rules

The load rules establish conditions to be fulfilled to derive from the data sources the value of the attributes of the entities that belong to the reconciled solution. Thus, the expression of the transformation that will automatically generates the integration rules of the load rule is showed in Figure VI-16.

```
<#
int i = 0;
Map<DataSource,String> paths = new Map<DataSource,String>();
foreach (DataSource ds in getDataSources()) {
      i++;
      String path = “P”+i;
      paths.put(ds,path);
#>

      Path P<#= i #> is <#= ds.Name #>
      <#
      foreach (Rule rule in ds.getRules()) {
            printRule(rule.getType())
      }
#>
Integration context IC_substr(<#= ds.Name,1 #>) is P<#= i #>
}
<# getPathCombinations(paths) #>
#>
```

Figure VI-16. Load Rules Transformation Expression

A concrete example of load rule instantiation is showed in Figure VI-17.

```
Path P1 is Mosaico
[Equals(province, p_name)] Province
[Belong_to.destination=p_name] Belong_to
[Belong_to.source=c_name and Equals(location, c_name)] City
[Sited_in.destination=c_name] Sited_in
[Sited_in.source=m_name and Equals(name, m_name)] Monument
Path P2 is DBPedia
[Equals(GetProvince(latitude, longitude), p_name)] Province
[Belong_to.destination=p_name] Belong_to
[Belong_to.source=c_name and Equals(location, c_name)] City
[Sited_in.destination=c_name] Sited_in
[Sited_in.source=m_name and Equals(name, m_name)] Monument
Path P3 is Yelp
[Equals(GetProvince(GetZipCode(display_address)), p_name)] Province
[Belong_to.destination=p_name] Belong_to
[Belong_to.source=c_name and Equals(city, c_name)] City
[Sited_in.destination=c_name] Sited_in
[Sited_in.source=m_name and Equals(name, m_name)] Monument
Integration context IC_M is P1
Integration context IC_D is P2
Integration context IC_Y is P3
Integration context IC_D_Y is P2, P3
Integration context IC_M_D is P1, P2
```

Figure VI-17. Load Rules Transformation Expression

4. Conclusions

This Chapter has presented and studied the transformations to be applied to the metamodel defined in Chapter V to automatically can test the instances of such metamodel. These transformations have been specified trough Text Template Transformation Toolkit (T4) provided by DSL-Tools.

The first section of this Chapter has presented how the transformations must be specified in T4 language. In this sense, it has been presented the architecture of the T4 template, it has been explained the typical execution sequence of a T4 template and finishing the section, it has been described the most common elements that compose a T4 template such as: directives, text blocks, control blocks and utility methods.

Next, section 2 and 3 of this Chapter, has presented the expressions of the transformations divided in: structural rules and load rules. The structural rules impose conditions to be fulfilled to create new entities and relationship into the reconciled solution. The load rules establish conditions to be fulfilled to derive from the data sources the value of the attributes of the entities that belong to the reconciled solution.

With the conclusion of this chapter, the definition of the two first pillars of MaRIA Framework have been covered at all. In this sense, next chapter will present the support tool that has been developed to give support to the methodology defined.

CHAPTER VII
SUPPORT TOOL

CHAPTER VII. SUPPORT TOOL

Chapters IV, V and VI have presented the two first pillars of MaRIA Framework, focusing in its requirement, analysis and testing set of processes, presenting the required metamodels for modelling an entity reconciliation problem and defining the systematic transformation protocol which it is possible to test the instances of this metamodel.

However, to make possible the practical use of this theoretical environment for the construction of entity reconciliation problems, it has been necessary to develop a tool that supports the definition of the entity reconciliation and, that allows the automation of the transformation rules defined in the previous chapter. The definition of this tool represents the third objective defined in Section 2 of Chapter III.

Finally, this chapter synthesizes a set of conclusions.

1. INTRODUCTION

MaRIA Tool is the last pillar that conforms MaRIA Framework. Figure VII-1 shows the position that this chapter is focusing inside the Framework.

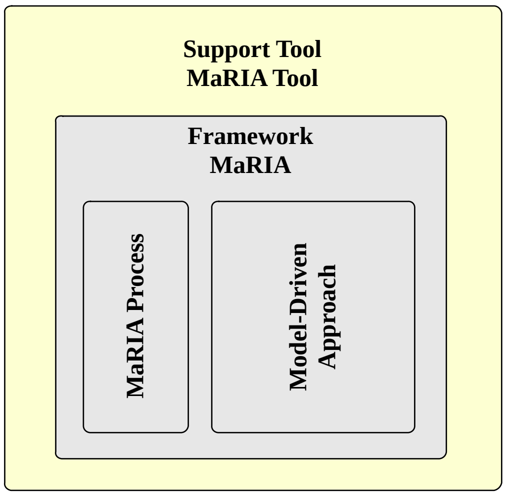


Figure VII-1. MaRIA Framework – MaRIA Tool

With the use of the Model-Driven Engineering paradigm and its direct and implicit translation with the use of the metamodels and transformation protocols between them, it is possible to guarantee uniformity, formality through a common terminology and a correct and complete definition of a concrete domain.

However, if this theoretical definition of metamodels and derivation rules are not accompanied by effective, usable and attractive software tools for the end user, the maintenance and management of models built based on such metamodels may become a too complex and expensive task.

In this sense, this chapter aims to address how to carry out the definition of the concrete syntax of the metamodels presented in Chapter V, how to automate the transformation rules specified in Chapter VI, and finally how to integrate both solutions into the support tool that makes possible its applicability in real environments.

There are a lot of tools that allow to the users to create DSLs, the more extended ones are usually presented as a framework or plugin ready to be installed in an IDE. Some of the more popular in the community are: Eclipse with Eclipse Modeling Tools with, IntelliJ with Meta Programming System (MPS) or Visual Studio with Domain-Specific Language Tools (DSL Tools).

A comparative study presented by Vilaine et al., (2012) has been taken as criteria in order to choose a tool for creating the DSL proposed. The main requirements that the selected tools needed to have for the development of this comparative were: graphical modeling, model to text (M2T) transformations and the creation of templates. The two more popular tools that agree with these requirements are: Visual Studio with Domain-Specific Language Tools (DSL Tools) and Eclipse with Eclipse Modeling Tools. For performing the comparative, the following aspects have been considered: meta-metamodel, metamodeling, visual editor components, transformations between models and visual editor deployment.

1.1.META-METAMODEL

For a to perform the operations of metamodeling, it is necessary to be based on a meta-metamodel that serves as the basis for creating the metamodel. In this sense, Figures VII-2 and VII-3 illustrates the meta-metamodels of both visual studio tools and eclipse respectively.

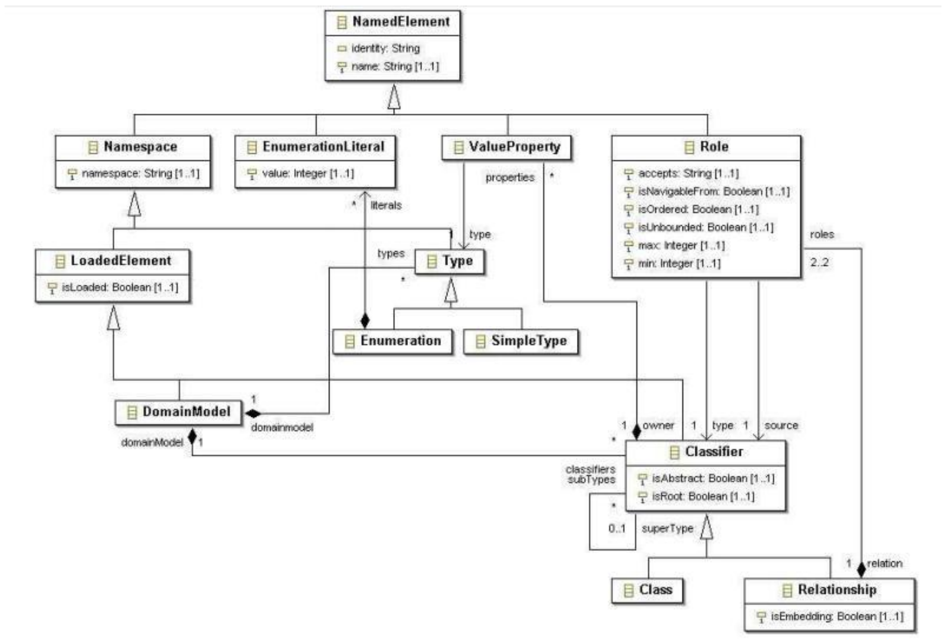


Figure VII-2. DSL-Tools Meta-Metamodel (Bézivin et al., 2005)

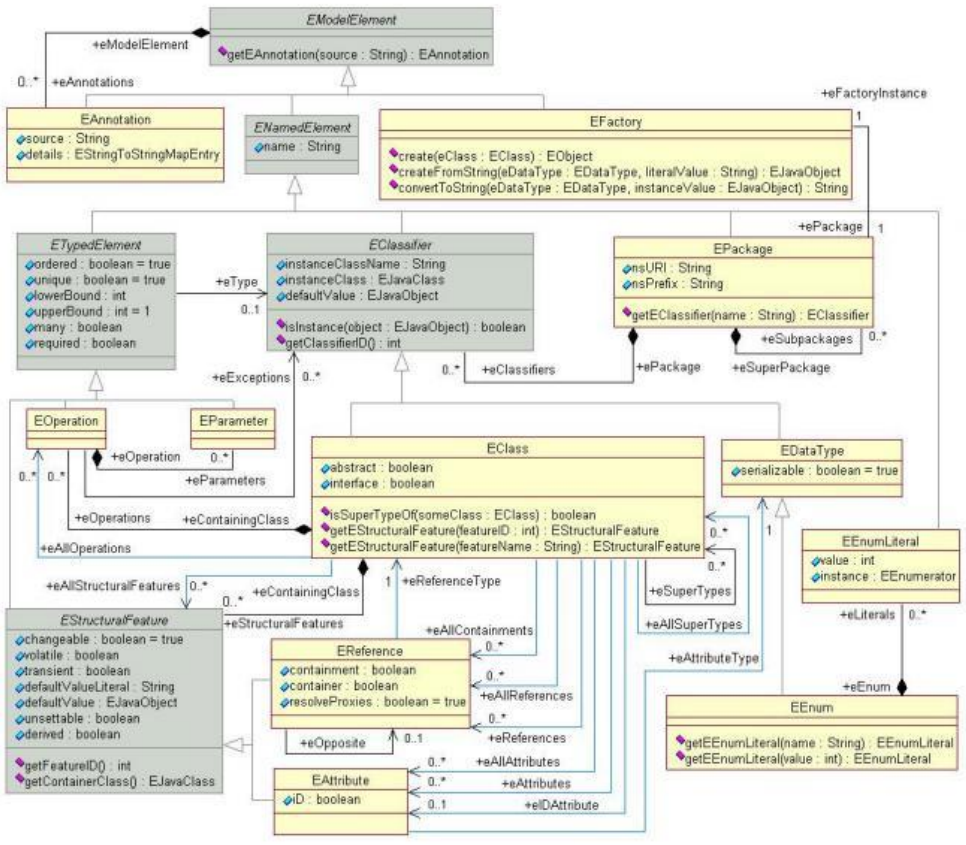


Figure VII-3. Eclipse Modeling Meta-Metamodel (Kolovos, 2016)

1.2.METAMODELING

The characteristics that each one of the tools offers described in Table VII-1.

Modeling	Characteristics
DSL-Tools	Serialization of XML files with .dsl extension
	Visual diagram of the metamodel in the .dsl.diagram file
	Multiple inheritance between metaclases
	Metaclasses with meta-attributes y meta-associations
	Meta-associations, with roles, multiplicities, navigability and types (assignment or composition)
Eclipse Modeling Tools	Serialization of XML files with .ecore extension
	Visual diagram of the metamodel in the .ecore.diagram file
	Multiple inheritance between metaclases
	Metaclasses with meta-attributes y meta-associations
	Meta-associations, with roles, multiplicities, navigability and types (assignment or composition)

Table VII-1. Characteristics of DSL and Eclipse Modeling Tools

1.3.VISUAL EDITION COMPONENTS

Both tools offer a graphical edition interface. In the case of Eclipse (Figure VII-4) the user must derive the graphical definition model, generate graphics and adjust the definition. In the case of DSL-Tools (Figure VII-5), all the changes may be performed from the properties panel.

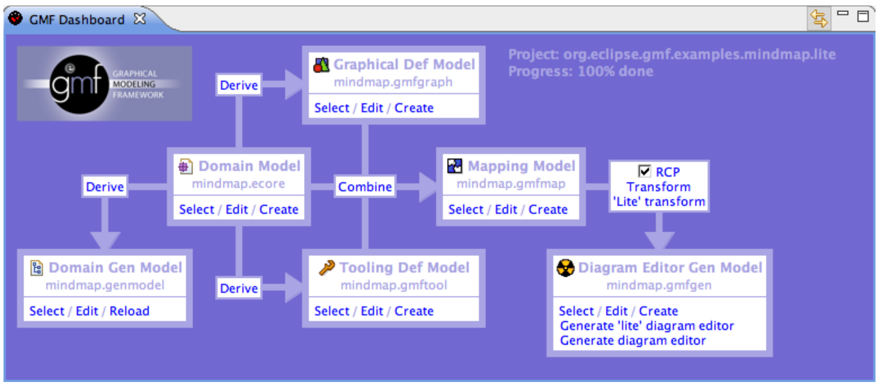


Figure VII-4. Graphical Edition Eclipse Modeling Tools (Eclipse, 2016b)

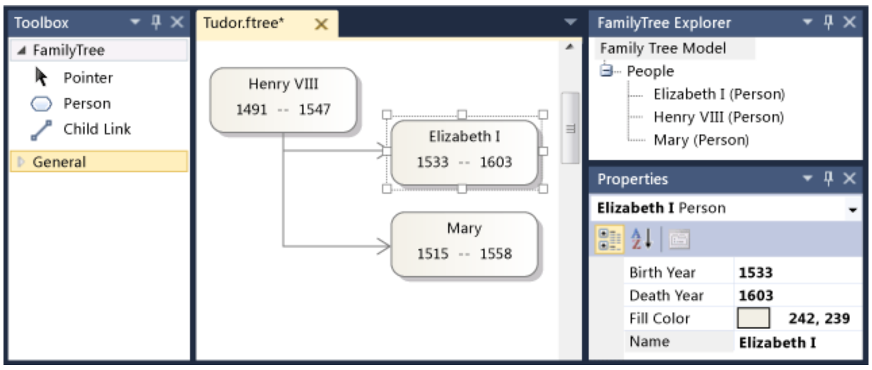


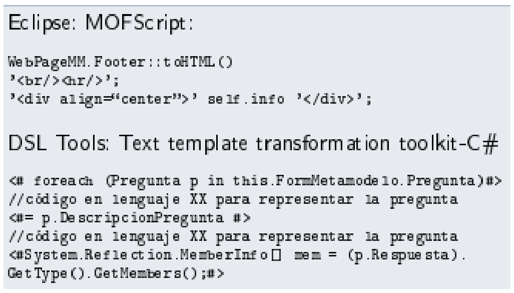
Figure VII-5. Graphical Edition DSL Tools (MSDN, 2016)

Similarly, for the mapping of components. On one hand, the visual editor of Eclipse we must derive the mapping model, generate the mapping model and adjust them and on the other and, in Visual Studio all the elements to customize the mappings are placed in the DSL Details window.

1.4.M2T TRANSFORMATIONS

DSL Tools uses the “Text template transformation toolkit” and Eclipse uses the “MOFScript” (Figure VII-6). Both environments use similar processes for this kind of transformations. Roughly, the steps that user must take are:

- Creation of the model.
- Creation of the file for performing the transformation.
- Processing of the modeled file.
- Obtaining the result.


Eclipse: MOFScript:

```
WebPageMM.Footer::toHTML()
'<br/><hr/>';
'<div align="center">' self.info '</div>';
```

DSL Tools: Text template transformation toolkit-C#

```
<# foreach (Pregunta p in this.FormMetamodelo.Pregunta)#>
//código en lenguaje XX para representar la pregunta
<#= p.DescripcionPregunta #>
//código en lenguaje XX para representar la pregunta
<#System.Reflection.MemberInfo[] mem = (p.Respuesta).
GetType().GetMembers();#>
```

Figure VII-6. M2T Transformations

1.5. VISUAL EDITOR DEPLOYMENT

Visual Studio allows to generate a VSIX plugin that will allow to use the DSL from an instance of Visual Studio. In Eclipse the user has two options: (i) generate a plugin or (ii) generate a desktop application.

In conclusion, both tools seem to be very similar and offer practically the same characteristics. However, for the development of the support tool of this doctoral thesis, it has been selected the Microsoft DSL-Tools for two main reasons: easy to use, it uses a well-known language (C#) by the doctoral student for generating the templates and transformations.

Finally, an introduction to Microsoft DSL-Tools and a small guide of how to install it, how to start a new project and the components offered for the creation of a new domain-specific language, is presented in Appendix A.

2. DEFINING THE CONCRETE SYNTAX OF METAMODELS

Models and metamodels alone do not explicitly require the use of any particular notation for their representation. The concrete syntax of a metamodel specifies how to visually represent the models through diagrams. It may be represented by two ways: textual or graphical (Fondement and Baar, 2007). As mentioned before, this proposal has been based on the graphical way, supported by the DSL-Tools. In this sense, the concrete syntax will consist of a set of templates, where each template specifies the visual representation of each class of a metamodel.

Table VII-1 represents all the elements that have been defined in the concrete syntax that represent the different instances that a metaclass of the metamodel defined in Chapter V may has.

Image Tool	Reprsentation
⛁	Data Source
☁	Wrapper
⟋	Data Source Link
▭	Entity
⬭	Attribute
⟋	Has Link
⟋	Transformation Link
⚙	Transform
▽	Filter
⬇	Load
fx	Conversion
⟋	Resolution Link
☰	Resolution
✣	Aggregation

Table VII-1. Concrete Syntax

- Data Source: represents each data source to be reconciled.
- Wrapper: represents the way of extracting information from one data source.
- Data Source Link: represent a connection between the Data Source and the Wrapper.
- Entity: represents each the entity of the model.
- Attribute: represents the attributes that define the entity.
- Has Link: represents the connection between entities.
- Transformation Link: represents the transformation that data must undergo to carry out the entity reconciliation problem. It is composed by four type of operations:
    - Transform: an attribute is transformed into another one.
    - Filter: it applies a filter between some attributes.
    - Load: it takes a data from a source and moves it to a target.
    - Conversion: it applies an operation to different attributes.
- Resolution Link: represents the way in which the reconciliation will be carried out. It is composed by two types:
    - Resolution: it represents the preferred attribute depending on a priority.
    - Aggregation: it represents the concatenation of all the values of the attributes involved in the operation.

For creating the metamodel presented in Chapter V, using the UML language, to the offered by Visual Studio with DSL Tools, the first step that the doctoral student performed was generating a “Minimal Language” project (see Annex A).

DSL Tools contain a set of tools for creating DSLs. Due to the operation of the DSL tools it was necessary to define a “DomainClass” to which must associated a “Diagram” type visual interface (see Figure VII-7). Since all the metaclasses of the metamodel contain an embedded relation with this root element with the same characteristics, it will we removed from the figures for two reasons: (i) improve the quality of the figure and (ii) Do not repeat elements already described.

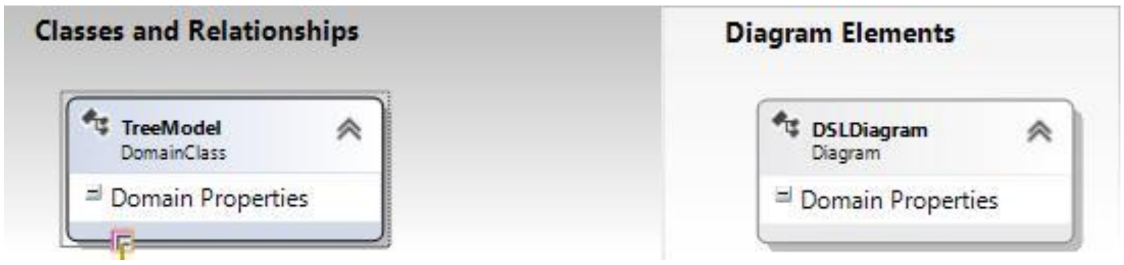


Figure VII-7. Root Element (DomainClass + Diagram)

2.1.DATA SOURCE MODEL

This section describes how the data source block of the metamodel presented in Chapter V has been modeled in DSL-Tools.

The “DataSource” (Figure VII-8) metaclass has been modeled as a “NamedDomainClass” type So it contains “Name” property by default and it will be unique in all the definition of the metamodel. Also, it has been defined the “URL” property to know the directory path of the data source from which the information wants to be extracted.

There is a “Embedding Relationship” type meta-association between the “TreeModel” and the “DataSource” metaclass that indicates that one “TreeModel” may contain zero or more “DataSource”. Also, it is needed another meta-association, in this time of “Reference Relantionship” type that associates the “DataSource” and the “Wrapper” metaclass because it is a reference from one metaclass to the other one.

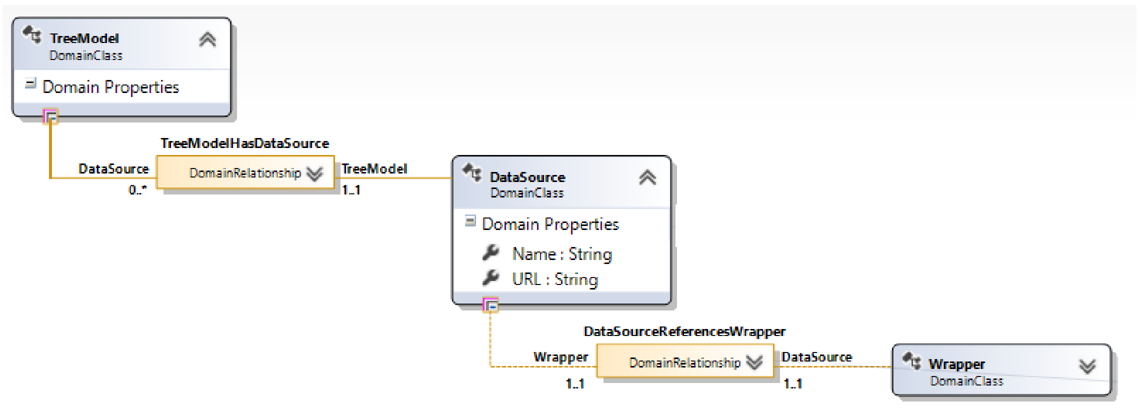


Figure VII-8. Root Element (DomainClass + Diagram)

The “Wrapper” (Figure VII-9) metaclass has been defined as a “DomainClass” class because it does not have the necessity of any property for creating the abstraction shape between the “DataSourceEntity” and “DataSource” metaclasses. It is possible to see that it is related with the “DataSourceEntity” metaclass through a “Reference Relationship” meta-association type.

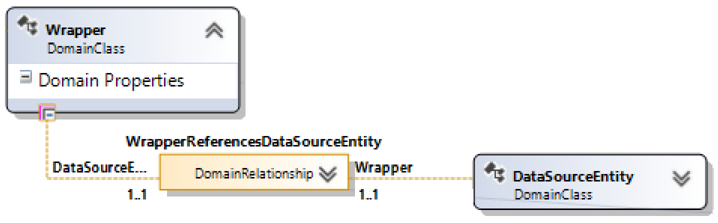


Figure VII-9. Wrapper Metaclass

The “DataSourceEntity” (Figure VII-10) metaclass is defined as a “NamedDomainClass” type metaclass, it means that, once it has been defined a name, it will not be a other one with the same description in all the diagram. This metaclass has a reference relationship with the “DataSourceAttribute” metaclass.

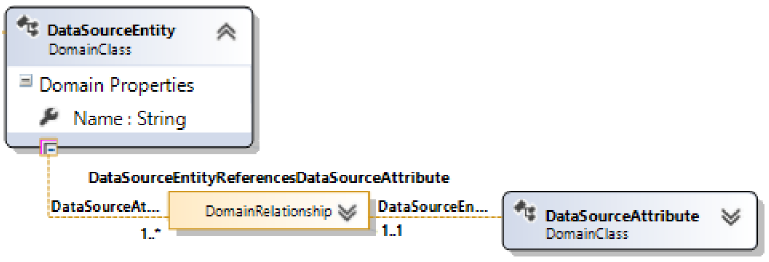


Figure VII-10. DataSourceEntity Metaclass

“DataSourceAttribute” (Figure VII-11) metaclass has been defined as a “DomainClass” because the attributes may be duplicated in the diagram. It must be modeled like that because it is necessary for using the same attribute (for example: “descripton”) in more than one instance of the “DataSourceEntity” metaclass. This metaclass contains the “Name” and “Type” string properties, one for specifying the name of the attribute and the other one for specifying the type of the attribute.

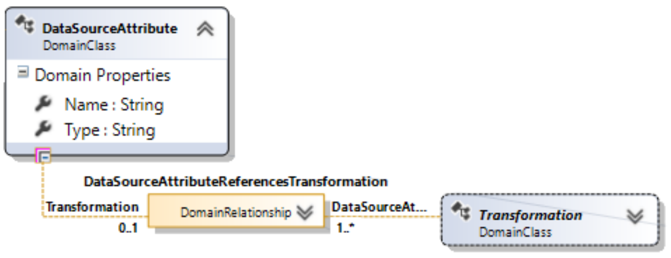


Figure VII-11. DataSourceAttribute Metaclass

2.2. Virtual Graph Model

The “EntityVertex” metaclass (Figure VII-12) is defined as a “DomainClass” metaclass. In this sense, in the final model it could exist more than one instance of the “EntityVertex” metaclass.

This metaclass has two properties: “Name” specifying the name of the attributewith a string type “Result” that will stablish if a concrete instance of the “EntityVertex” that has been created as result of a single or a set or operations.

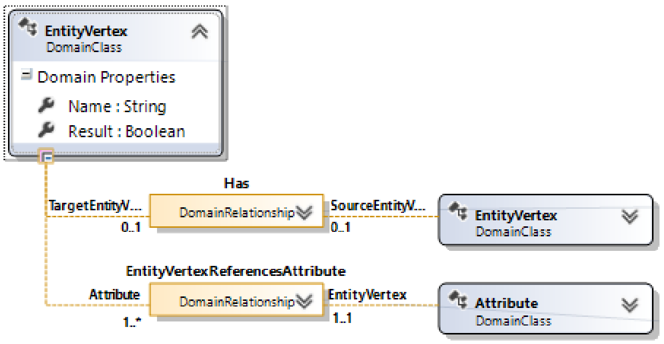


Figure VII-12. EntityVertex Metaclass

The “Attribute” metaclass (Figure VII-13) follows the same pattern than the “DataSourceAttribute” but with the difference that instead of dealing with transformations it does so with transformations. Otherwise the behavior is similar.

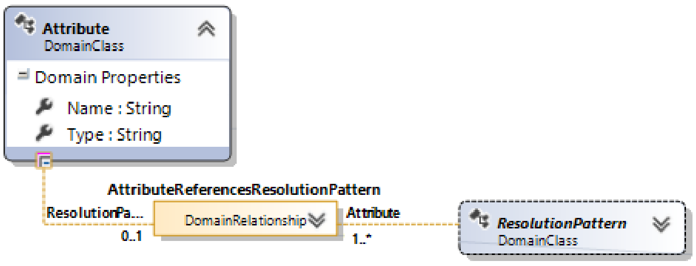


Figure VII-13. Attribute Metaclass

2.3. Transformation Model

The “Transformation” metaclass (Figure VII-14) is defined as a “DomainClass” metaclass and it is the metaclass from which all specific transformations inherit, each transformation has a determined behavior so it will behave differently (Figure VII-15).

It is possible to see that the multiplicities represent that whenever there is a transformation must have a relation with the attribute, but for an attribute it is not always mandatory to have an associated transformation.

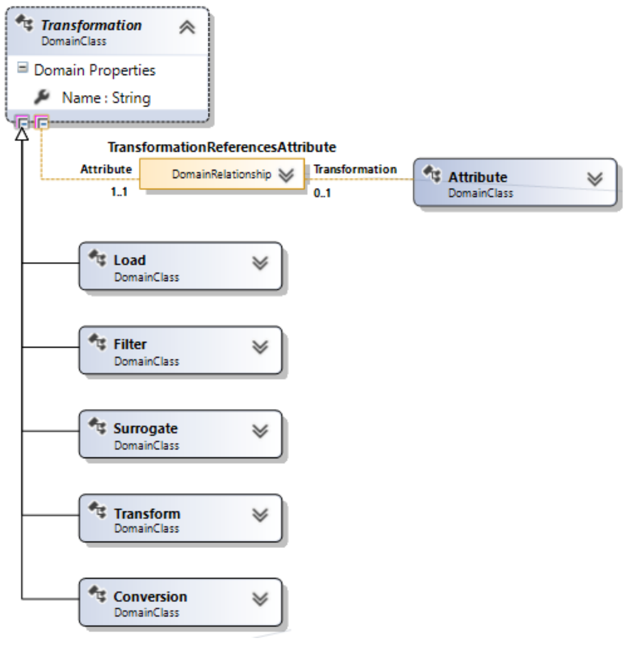


Figure VII-14. Transformation Metaclass

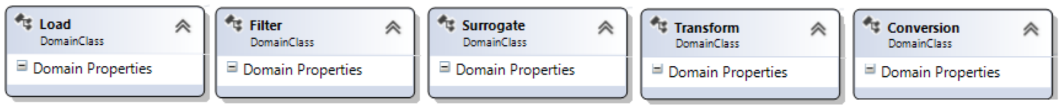


Figure VII-15. Transformation Operations Metaclasses

The “ResolutionPattern” metaclass (Figure VII-16) is defined as a “DomainClass” metaclass and it has a similar representation corresponding to the “Transformation” metaclass, but, in this case, the operations will be carried out between attributes. (Figure VII-15). From this metaclass all specific resolution transformations inherit (Figure VII-17), each operation has a determined behavior so it will behave differently.

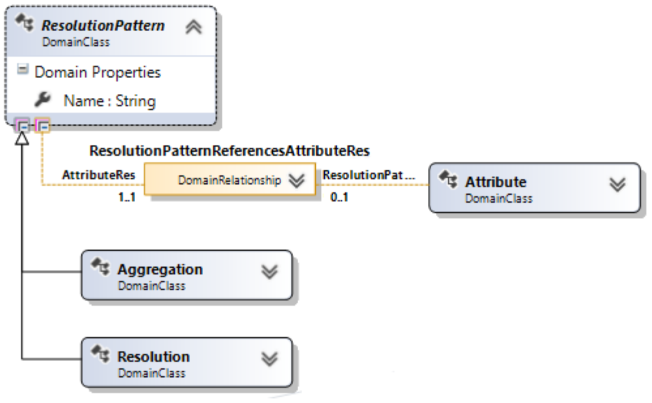


Figure VII-16. Transformation Metaclass

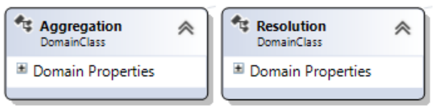


Figure VII-17. Resolution Operations Metaclasses

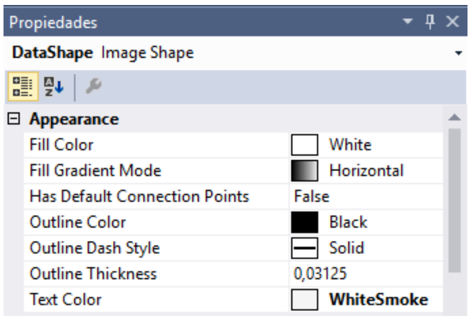
3. Editor Design

The editor is defined by two types of elements, on one hand, the appearance of the tools that are going to be generated in the tool box and on the hand, the appearance of this tool in the diagram.

Firstly, it is necessary to define the appearance that the metaclasses and the meta-associations will have when the execution of the DSL is running. This operation can be easily done adding "Shape" element types for the metaclasses and "Connector" element types for the meta-associations.

There are some types of shapes or forms for the metaclasses. In this implementation are used two of them: "Geometry Shape" and "Image Shape" (Figure VII-18).

- **Geometry Shape**: shows the metaclass as a form that may be rectangular, oval or rounded.
- **Image shape**: it is used when a metaclass behaves like an image but it is necessary that this metaclass still has the logical load that a metaclass has.



Appearance	
Fill Color	⬜ White
Fill Gradient Mode	▨ Horizontal
Has Default Connection Points	False
Outline Color	⬛ Black
Outline Dash Style	Solid
Outline Thickness	0,03125
Text Color	⬜ **WhiteSmoke**

Figure VII-18. Types of Used Shapes

To all the forms can be added son decorators that let change the behavior when the domain classes are in execution. There are three types of decorators (Figure VII-19):

- **Expansion and Collapse Decorator**: it will allow that once running the metaclasses instances have a "+" and "–" symbol which will show or hide the properties of the class.
- **Icons Decorator**: it will allow add an icon in a concrete position of the instances of the metaclasses in execution time.
- **Text Decorator**: it will allow to add the visualization of the properties of the instances of the metaclasses in execution time.

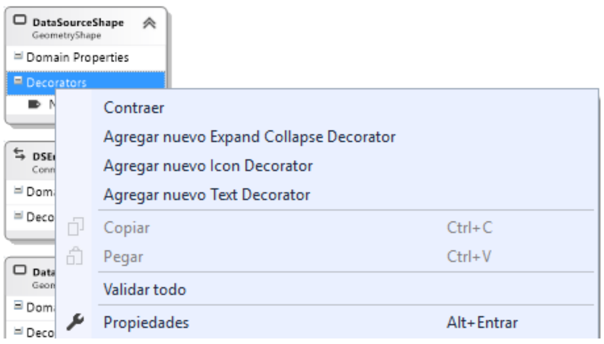


Figure VII-19. DataSource Shape Decorador

Once created the forms for the metaclasses and the meta-associations, for stablishing the relationship between the metaclasses and meta-associations and the elements of the diagram it will be used the "Diagram Element Map" (Figure VII-20) element. Once selected this tool, for relate a shape to a metaclass the only necessary thing to do is click on the metaclass and click on the shape that wants to be related to. Making this step, both metaclass and shape will be related.

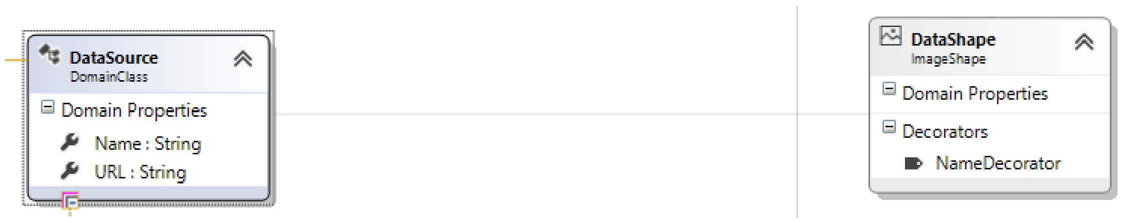
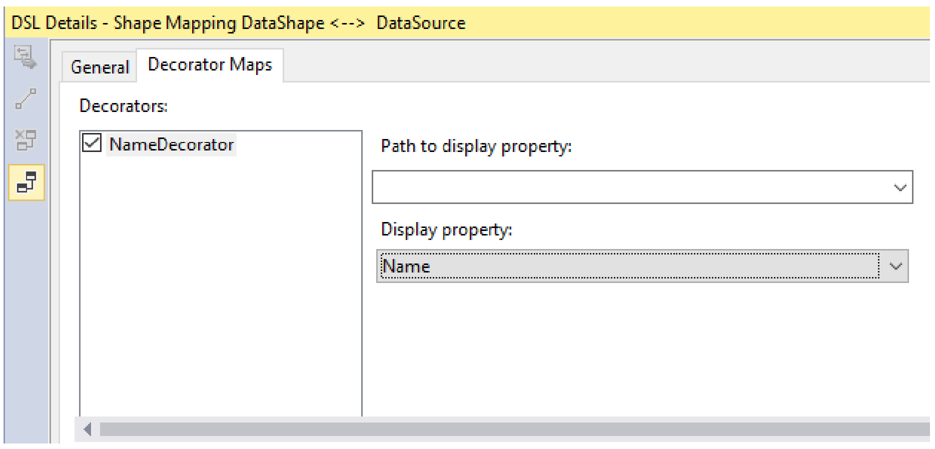


Figure VII-20. Diagram Element Map Example

Once the relationship has been created the next step is configuring it. For that, clicking on the relationship, it is possible to win the DSL Details windows all the information related with the mapping divided in two tabs: "General" shows all the information autocompleted according to the relationship created and "Decorator Maps" (Figure VII-21) where it is necessary to select the decorator that wants to be used. The same process must be applied to the connectors.



DSL Details - Shape Mapping DataShape <--> DataSource

General | Decorator Maps

Decorators:

☑ NameDecorator

Path to display property:

Display property: Name

Figure VII-21. Decorator Maps

Having all the metaclasses and meta-associations defined, it is necessary to create a new toolbox. For that, it is necessary to DSL Tools option located in the path showed in Figure VII-22 of the in the DSL Explorer window.

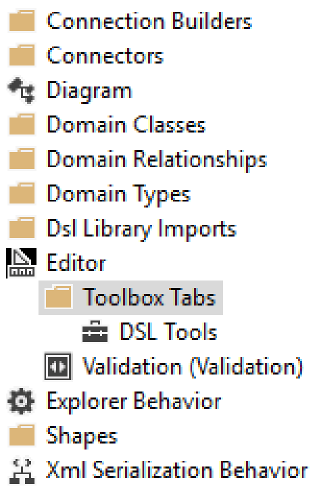


Figure VII-22. Decorator Maps

Making right click on this tool, it will appear a menu where it can be added both element or connection tools. The Element Tools refer to the tools that allow to create the metaclasses and the Connection Tools refer to the tools that allow to create the meta-associations.

For the Element Tools, the properties that must be configured are:

- **Class**: name of the metaclass.
- **Name**: name of the tool.
- **Notes**: informative notes.
- **Caption**: name that will appear in the toolbox.
- **Help Keyword**: keyword that allow to find this tool if a search is performed.
- **Toolbox Icon**: icon of the tool that will appear in the toolbox.
- **ToolTip**: name that will be the suggestion of the tool.

For the Connection Tools, the properties that must be configured are:

- **Connection Builder**: this parameter refers to the constructor of the reference.
- **Name**: name of the tool.
- **Reverses Direction**: direction of the relationship.
- **Notes**: informative notes.
- **Caption**: name that will appear in the toolbox.
- **Help Keyword**: keyword that allow to find this tool if a search is performed.
- **Source Cursor Icon**: icon that the relationship will have in the source element.

- **Target Cursor Icon**: icon that the relationship will have in the target element.
- **Toolbox Icon**: icon of the tool that will appear in the toolbox.
- **ToolTip**: name that will be the suggestion of the tool.

Once the process of creation of these elements have been defined, from Figure VII-23 to Figure VII-24 it is shown how the form of all the connectors and metaclasses has been defined. It is important to highlight that if there is a special icon defined for an element, it is sited in top of each figure.

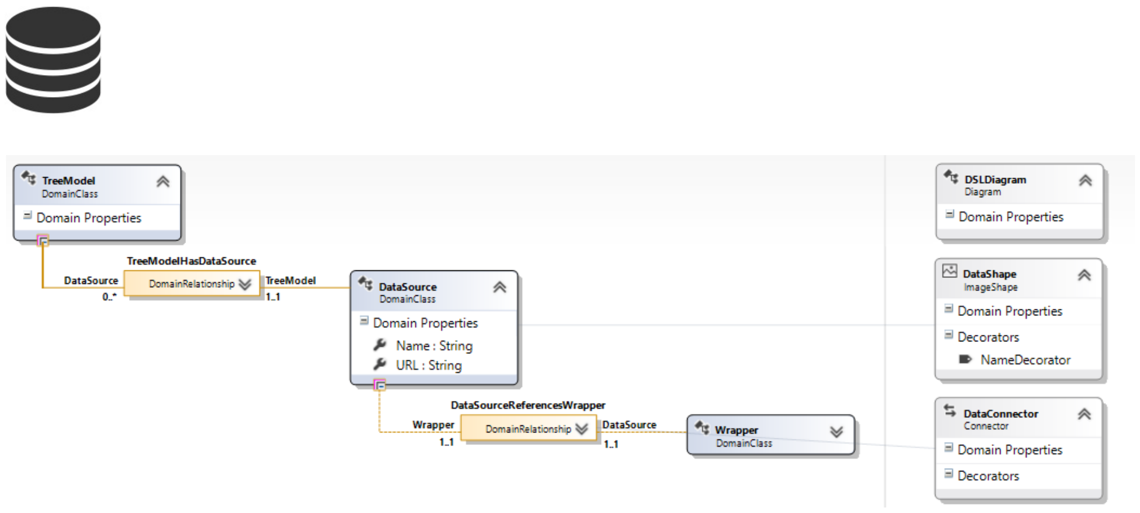


Figure VII-23. Data Source Element Definition

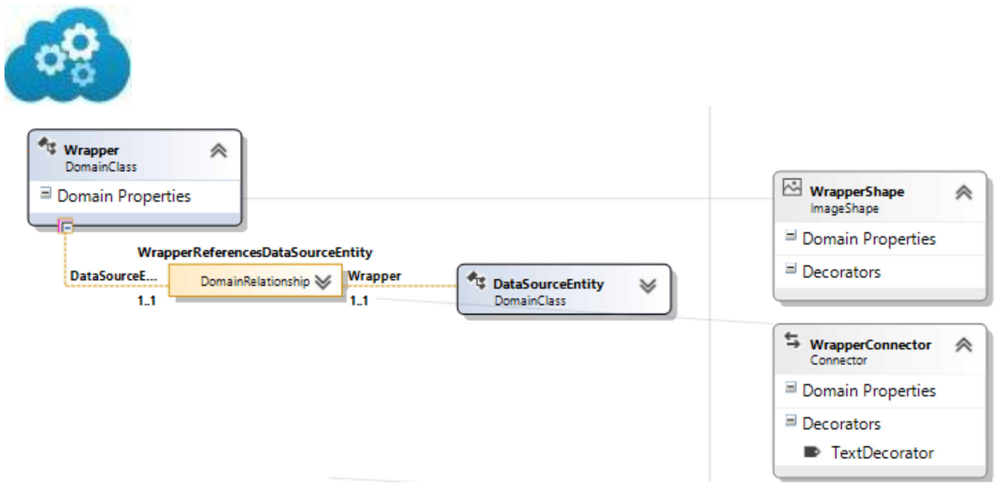


Figure VII-24. Wrapper Element Definition

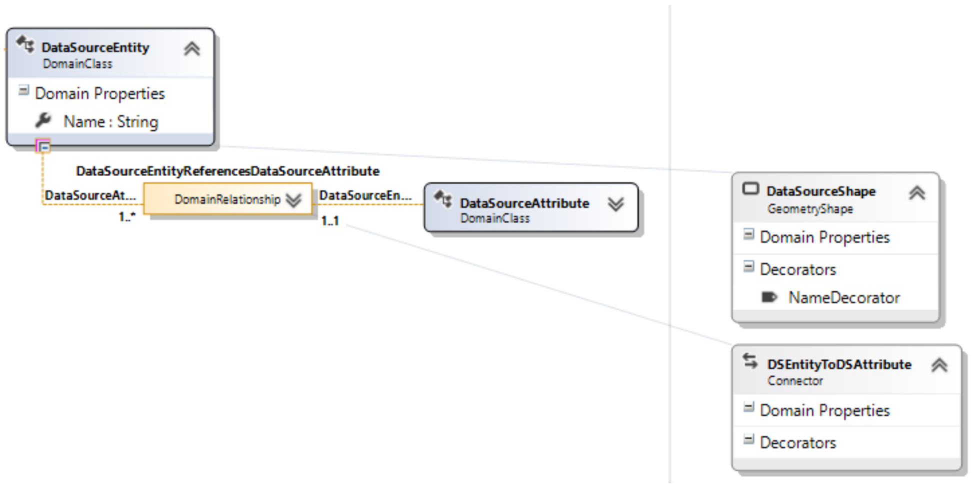


Figure VII-25. DataSourceEntity Element Definition

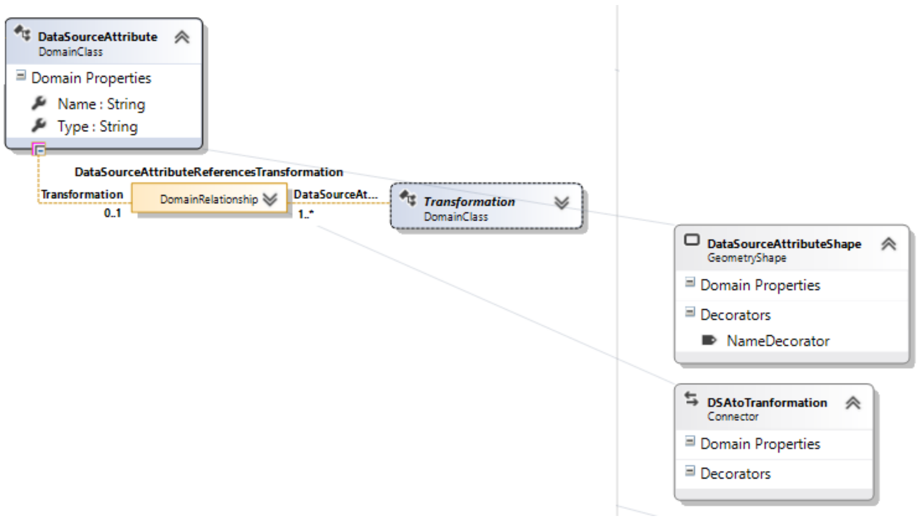


Figure VII-26. DataSourceAttribute Element Definition

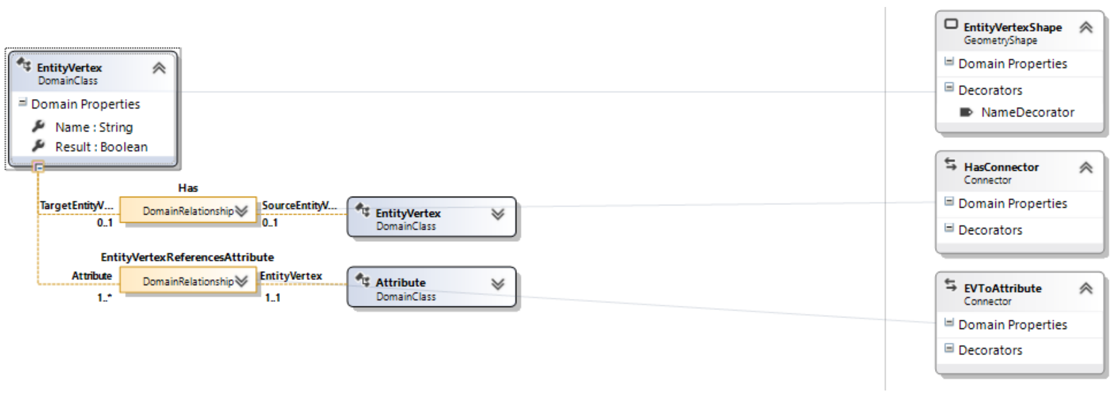


Figure VII-27. EntityVertex Element Definition

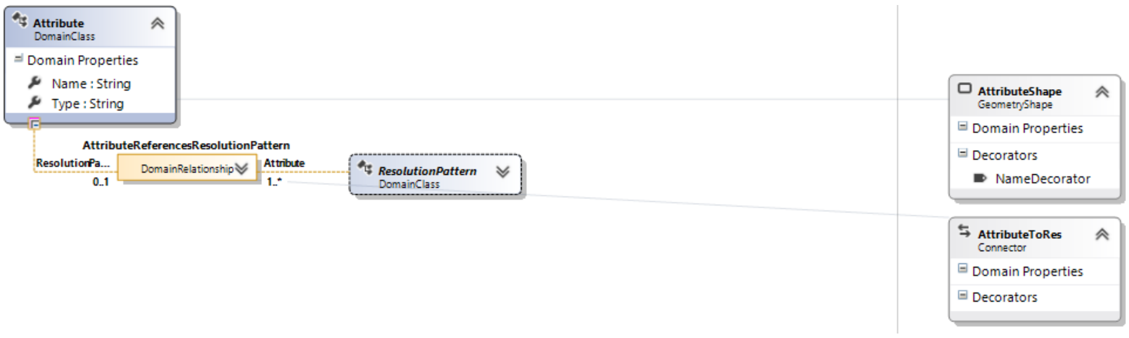


Figure VII-28. Attribute Element Definition

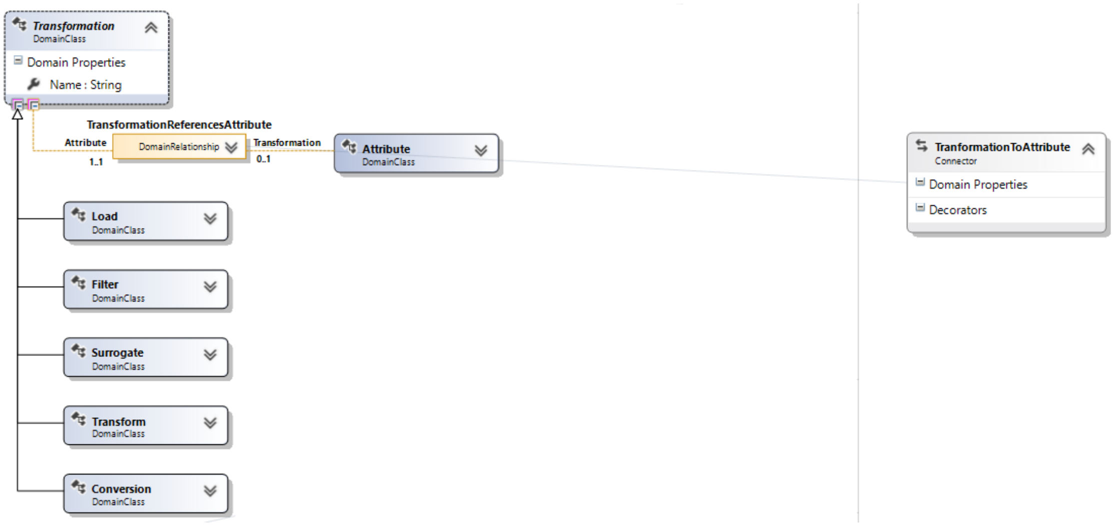


Figure VII-29. Transformation Element Definition

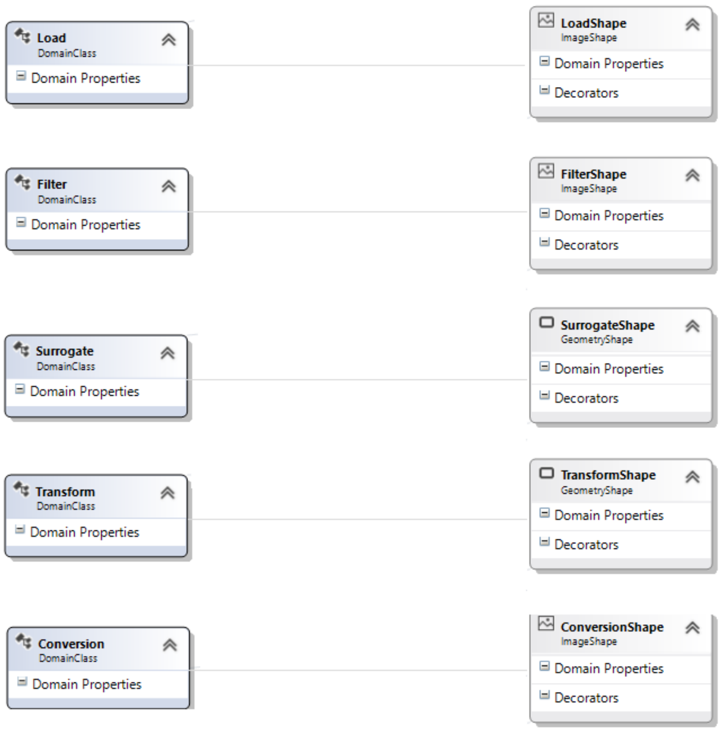


Figure VII-30. Transformation Types Element Definition

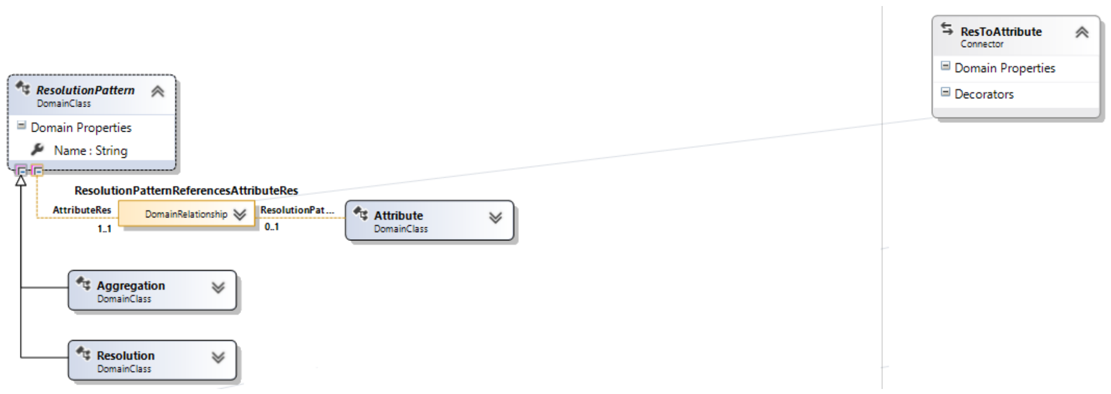


Figure VII-31. Main Resolution Element Definition

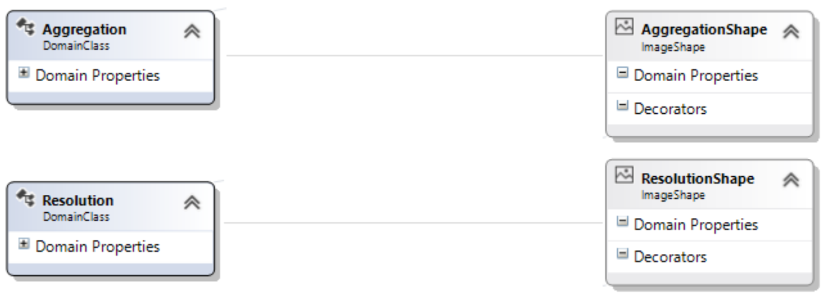


Figure VII-32. Resolution Operation Element Definition

4. RESULT

Once all the metaclasses, meta-associations, shapes and forms have been defined. The execution of the DSL created will open a new instance of Visual Studio 2015. In this instance, it will be displayed the toolbox that has been defined (Figure VII-33) and a window where a new diagram may be defined.

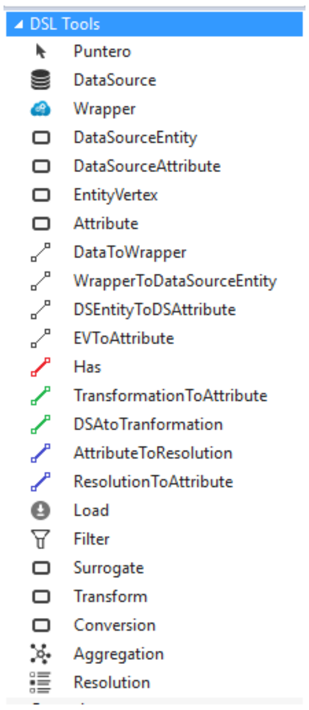


Figure VII-33. Toolbox

Finally, an example of model designed with the developed DSL is shown in Figure VII-34.

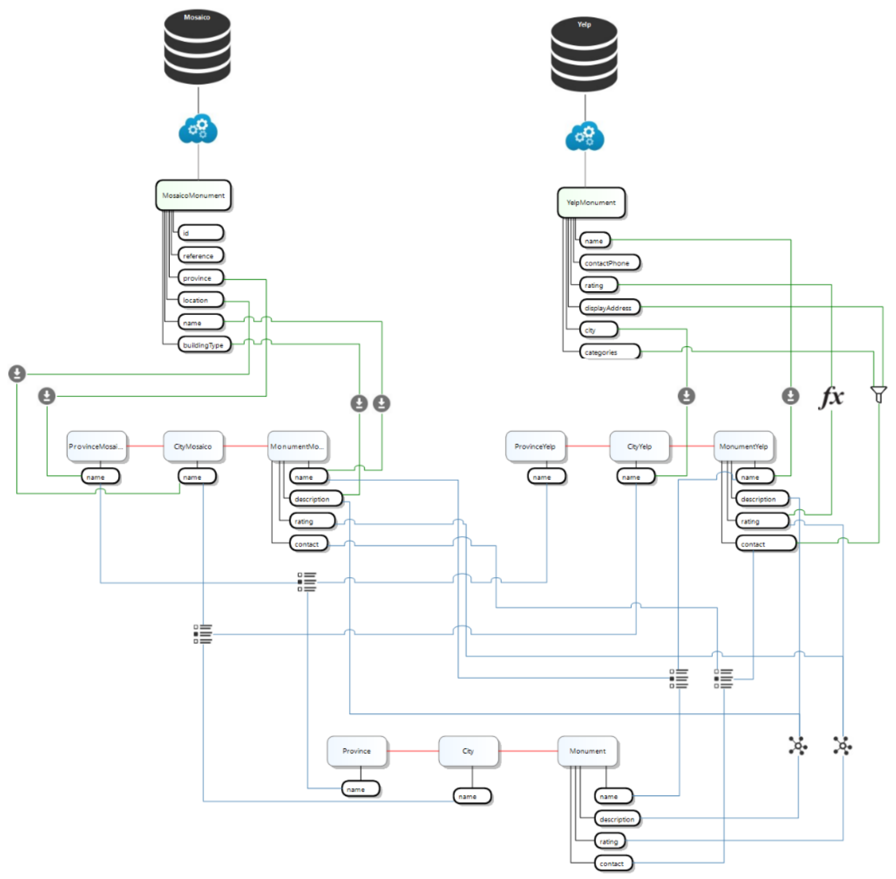


Figure VII-34. Example of Modeling

5. Conclusions

During Chapters IV, V and VI, it has been defined by a formal way all the theoretical framework in which the proposal specified in this doctoral thesis is based on.

However, to ensure the feasibility and applicability of this theoretical framework within practical environments and a production context, it is convenient and necessary to develop a CASE tool that supports it to make easier the maintenance and improve the quality of the results. This has been the purpose of this chapter and to that end, different aspects have been addressed.

At first, the section 2 of this Chapter defines how the concrete syntax of the metamodel proposed in chapter V has been defined. For this purpose and after a comparative study between Eclipse Modeling Tools of Eclipse and DSL Tools of Microsoft, it was decided to use the one provided by Microsoft.

Once defined the profile in DSL Tools, section 3 presented the editor design phase. There, it was described how to configure the forms and shapes of the elements (that represent instances of metaclasses) and relationships (that represent instances of meta-associations).

Section 4 present how the transformations have been defined to automatically generate the business rules that represent the test requirements of the application to develop and finally, section 5 has present how the final DSL created looks like and an example of model defined with this support tool.

CHAPTER VIII VALIDATION

CHAPTER VIII. VALIDATION

This doctoral thesis has presented the MaRIA Framewok and its three main pillars: the MaRIA Process, a set of activities to be added in any software development methodology that allows prepare the software system to be developed, to guarantee a suitable entity reconciliation and having the possibility to be systematically tested, the Model-Driven Approach, composed of the MaRIA Metamodel and a set of Derivations that allows the software engineer to model an test the entity reconciliation problem and finally, the MaRIA Tool, a domain-specific language supported by Microsoft Visual Studio that let some software engineer model entity reconciliation problems in a simple way.

Last chapter has presented the more important characteristics of each of the elements that compose the MaRIA Tool. Now it is time to validate the tool.

This chapter presents the results of a real proof of concept, based on a problem related to the data management of the cultural heritage of the fixed monuments of the region of Andalusia (Spain).

1. TWO REAL-WORLD CASE STUDIES

This section aims to develop to real-world case studies based on different projects that are being carried out in the research group where this Doctoral Thesis has been developed: DIPHDA and ADAGIO.

1.1. DIPHDA

One of the main objectives of the "Instituo Andaluz de Patrimonio Histórico (IAPH, Andalusian institute of historical patrimony) in the management of cultural heritage information is concerned with the design and implementation of an effective and operational information system capable of integrating all information resulting from research, documentation, conservation, protection, dissemination, etc. Of the cultural heritage, so that the information reaches the person who needs it at the right time for decision making (Instituto Andaluz de Patrimonio Histórico, 2010).

In this context, the IAPH, at the beginning of the 90s, the "Sistema de Información del Patrimonio Histórico de Andalucía" (SIPHA, Andalusian Historical Heritage Information System) was launched, which included, among others, the following advances:

- Creation of standardized, integrated and computerized standards on the different patrimonial entities.
- Creation of a standardized documentary language, the Andalusian Historical Heritage Thesaurus, an international pioneer for its multidisciplinary.
- Incorporation of Geographic Information Systems (GIS).

- Photographic and/or audiovisual documentation of patrimony or entities included in the system.
- Transfer of information through Information Services.
- Online consultation of the different databases.

In recent years, SIPHA sytem has been intetrated with the “Sistema para la Gestión Integral del Patrimonio Cultural” (MOSAICO, System for the Integral Management of Cultural Heritage), project of the Ministry of Culture of the Andalusia region (MOSAICO, 2016).

MOSAICO (Figure VIII-1) is a horizontal and global system that aims: (i) offer the technological resources and tools for the management of historical patrimony, (ii) offer a global information system that will store information about all the cultural patrimony and (iii), bring the public and government closer and more specifically, the information related to patrimony. This system was developed by the IAPH to meet their own objectives such as: managing cultural heritage information, protecting cultural heritage information of Andalusia, preserving the cultural heritage of Andalusia, disseminating the values of cultural assets or bringing government to citizen (Ponce et al., 2010).

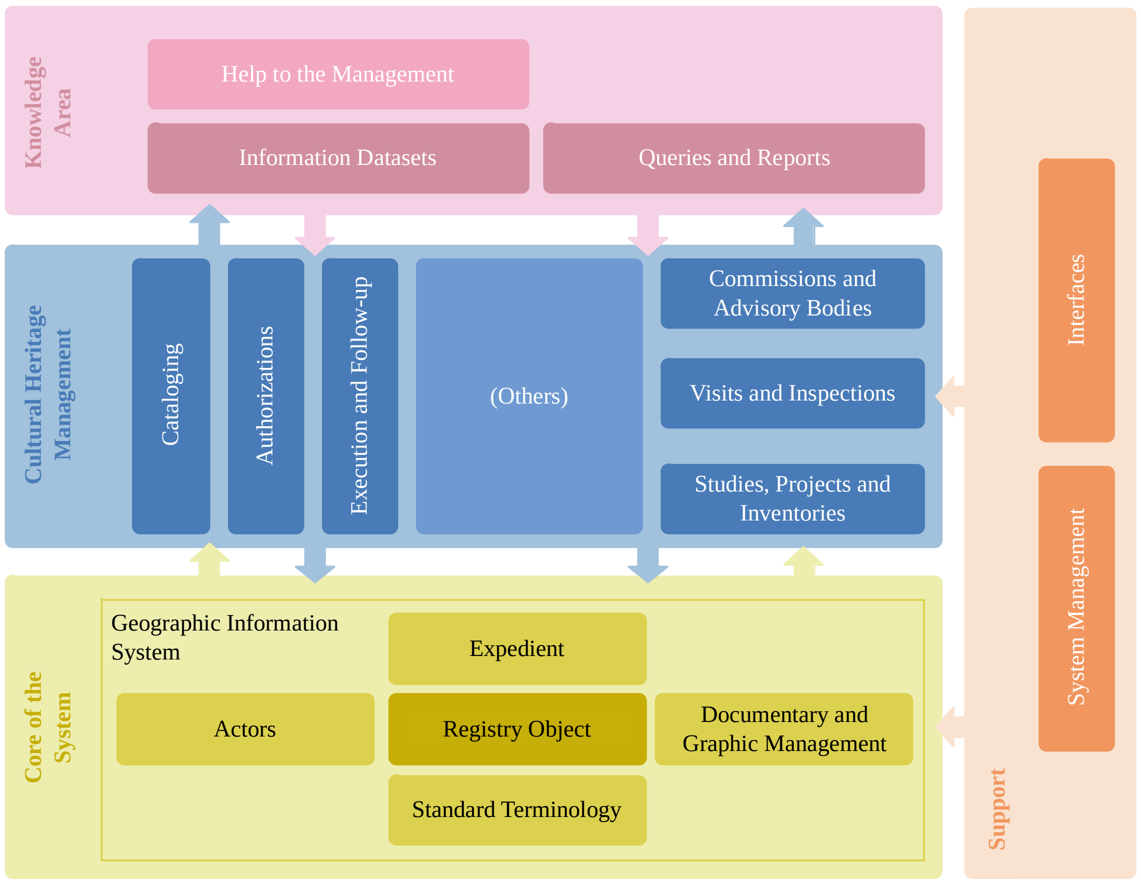


Figure VIII-1. MOSAICO Architecture (MOSAICO, 2016)

There are lots of monuments and several data sources where the information is stored so for the IAPH, keep in control all information published about patrimony in the worldwide suppose a very difficult task. In addition, the size and complexity of these data sources make complicated the management of these systems due to the large amount of information stored on them (for example, just MOISAICO, stores terabytes of information). Then, it is necessary to reconcile the existing information about monuments from all data sources.

Furthermore, the process in which the information of historical patrimony mentioned is managed, is carried out in a very rudimentary way. When a campaign is done at any point, either externally or internally, it is performed offline, it means, the team in charge of this process perform the campaign and once the delivery finishes, the administrators make the validation of the patrimony found one by one.

For example, if there is a reservoir that was studied in the 80’s and now in 2017, a revision is requested the new process requires the following steps

- The IAPH exports the data that is already in MOSAICO system and it is given to the team in charge of the revision of the monument.
- The team load, at their discretion, the new things that have been found.
- Once this process is finished, the team give the report to the IAPH and they perform the integration in the system.

In this context, the variabiliy is very high and also, the integrity of data is questioned.

Trying to offer a solution to these problems, the project “DIPHDA” (Dynamic Integration for Patrimonial Heritage Data in Andalucía) is being developed in collaboration with the Fujitsu Laboratories of Europe (FLE). The objective of DHIPDA is to achieve significantly improved accuracy and data management efficiency, based on reconciliation logic applied to open data information, as opposed to simple string matching reconciliation. This solution will can integrate management different systems. For this case the “MOSAICO”, Wikipedia and Yelp systems were used.

The information that DIPHDA manages is retrieved from the process of reconciliation done in one of its functionalities where the user must define the data structure where the results of the reconciliation process will be stored. This functionality is covered with the domain-specific language developed as support tool of this doctoral thesis presented in Chapter VII.

In this context, DIPHDA project provides a DSL (MaRIA) for designing the concrete entity reconciliation problem to be addressed. The software engineer must design the data structure with all the necessary attributes and operations for carrying out the entity reconciliation process. Figure VIII-2, illustrates a brief example of how the instantiation should be made by the software engineer.

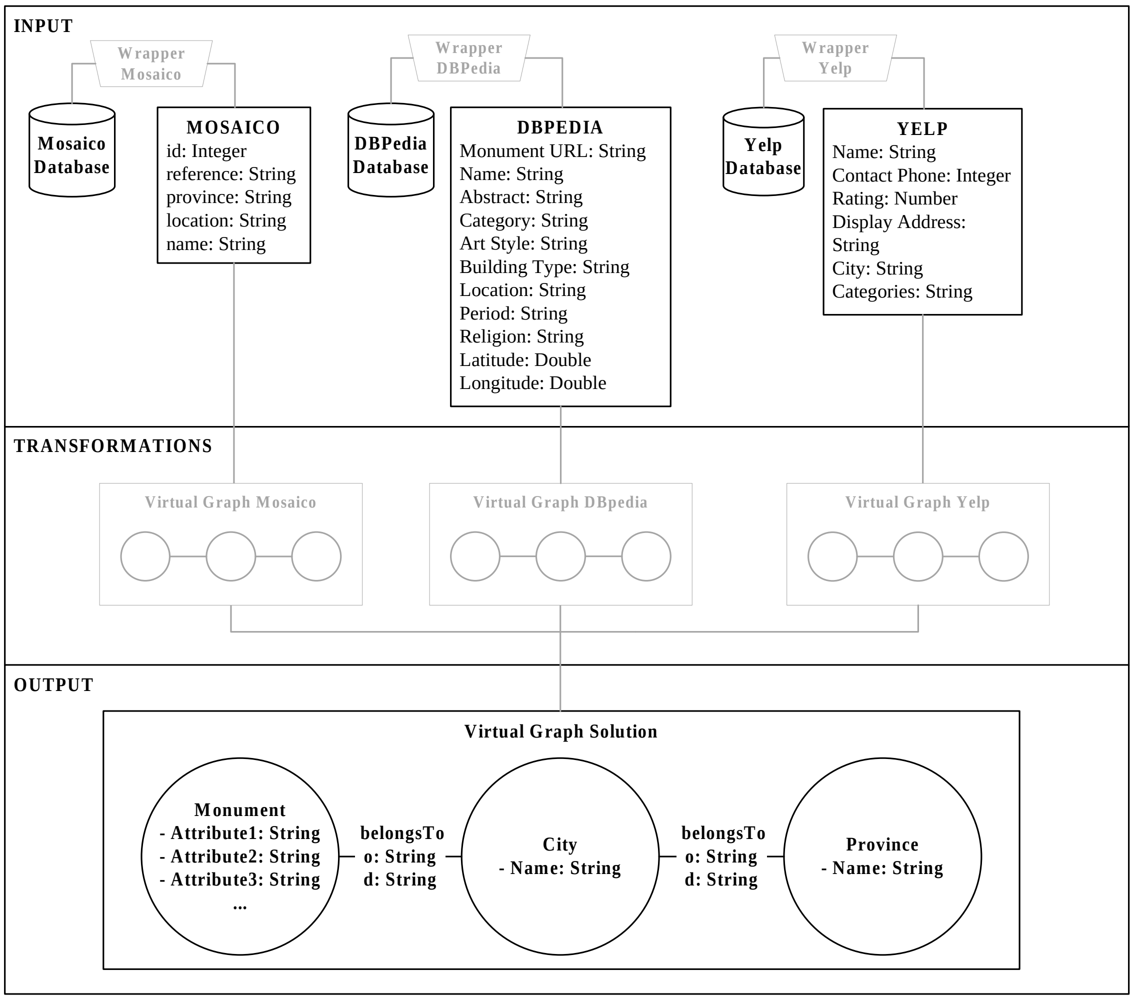
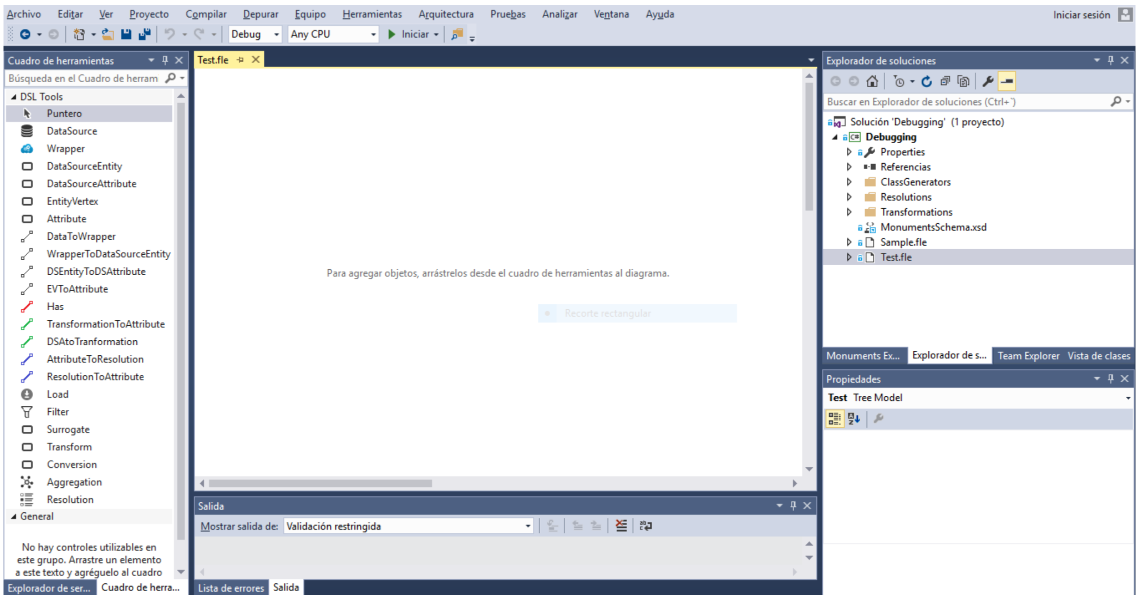


Figure VIII-2. Draft Example of Modeling

To materialize this example using the MaRIA Tool, it was decided to make a proof of concept designing the entity reconciliation problem between two heterogeneous data sources: the first one with the information stored in mosaic and the second one, with the information stored in Yelp.



Para agregar objetos, arrástrelos desde el cuadro de herramientas al diagrama.

No hay controles utilizables en este grupo. Arrastre un elemento a este texto y agréguelo al cuadro

Figure VIII-3. MaRIA Tool

Following the example showed in Figure VIII-2 and considering how is the visual aspect of the MaRIA Tool illustrated in Figure VIII-3, the result of the real model can be observed in Figure VIII-4.

For achieving to this model and following the MaRIA Process described in Chapter IV, the first step that that software engineer had to design was the data sources from which the information was going to be extracted. As it is possible to see, there are the two different heterogeneous data sources mentioned before: “Mosaico” and “Yelp”.

Next, the software engineer had to define the entities where the information was going to be stored during the process of the entity reconciliation. There are two entities, one for each data source. In addition, the user had to define the attributes that defined each data source. In the case of “Mosaico” these were: “id”, “reference”, “province”, “location”, “name” and “buildingType”. In the case of “Yelp” these were: “name”, “contactPhone”, “rating”, “displayAddress”, “city” and “categories”.

Once these elements were defined, the user modeled the “wrappers”, one for each data source, thus making it possible the information transfer from a data source to a defined entity.

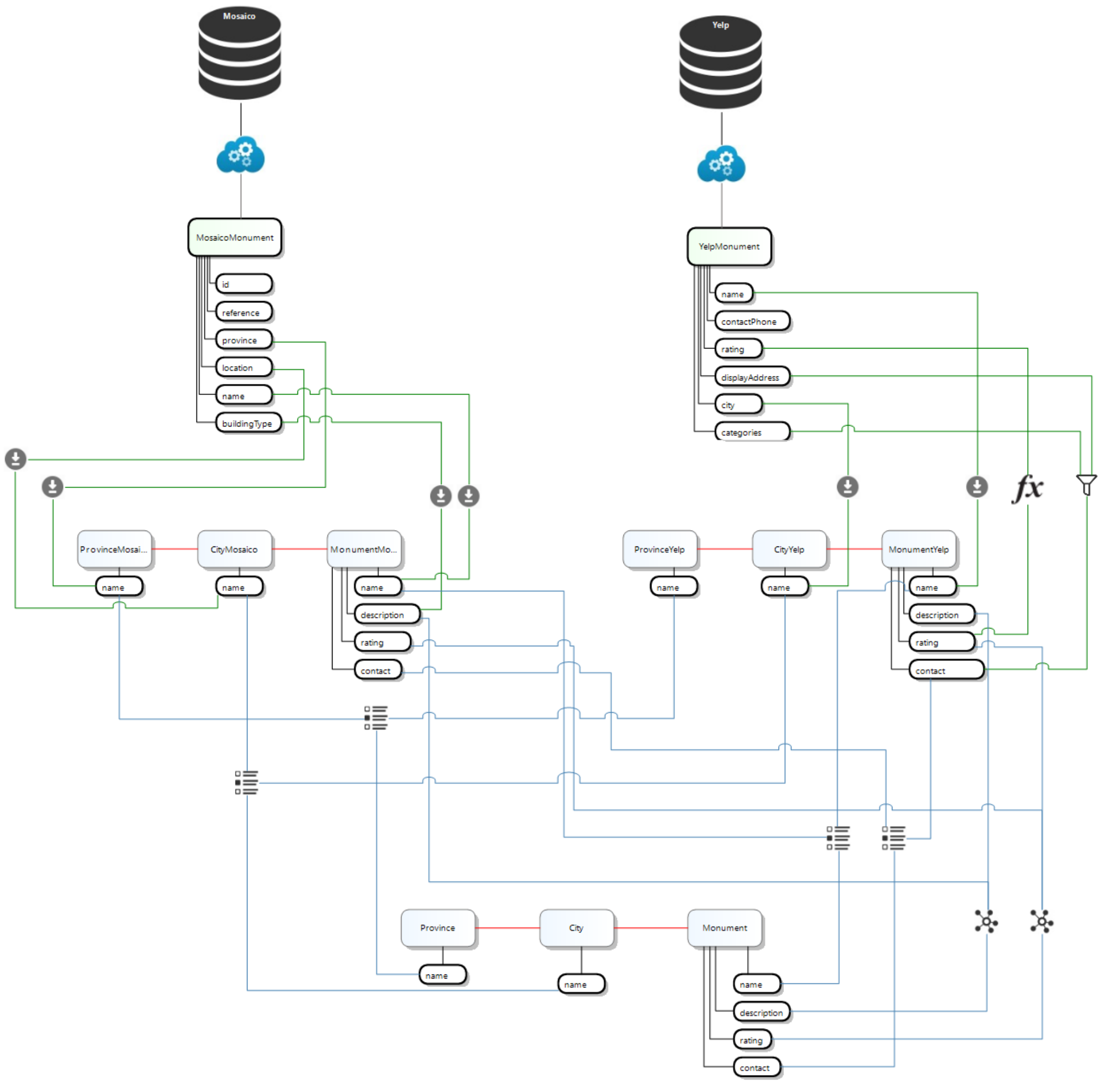


Figure VIII-4. Real Entity Reconciliation Problem modeled with MaRIA

The next step that the user had to perform was the designing of the structures where the solution was going to be stored. In this sense, it was defined three equals graph structures, two that were connected to each data source and another one that was going to be the final structure where the data generated from the reconciliation process would be stored. As it is possible to see in Figure VIII-4, the graphs structure was composed of three vertexes and two relationships between them. One of the vertexes ("CityMosaico", "CityYelp" and "City"), stores information about the city where the monument is located. Other of the vertexes ("ProvinceMosaico", "ProvinceYelp" and "Province"), stores information about the province where the monument is located. These two vertexes have a "name" property and are related between them with a "isProvinceOf" relationship, indicating that one city belongs to a province. The final ("MonumentMosaico", "MonumentYelp" and "Monument") vertex represents the monument itself. They have four attributes that define the monument: "name", "description", "rating" and "contact".

Performing the last step of the analysis process of the MaRIA Process, the software engineer had to define the transformation between the different attributes of the entities and graph structures. As it is possible to note, the information transfer between the data sources and the entities were defined trough three types of operations: “load”, where the information is retrieved and loaded into the attribute with any other transformation, “average”, where the information of the rating of the monument was going to be the average of all the rates in case of existing more than one and “filter”, applied to the display address and the category of the monument. Finally, the operations between the data structures were defined. It was used two types of operations: “priority”, giving a priority to the data source and taking the information of this one as priority and the “concatenate” operation, where the information of the description attribute of each data structure must be concatenated and stored in the description attribute of the final structure.

1.2. ADAGIO

Data of interest for the field of medical, clinical and pharmacological research are generally located through diverse institutions in diverse and heterogeneous systems. These systems implement generally heterogeneous data structures and data semantics.

For today research groups, the selection of populations per socio-demographic, documentary, traditional and/or chronic variables that define and identify them, either as study objectives or as contrast populations, is very complex given the variety of data sources, formats and levels of aggregation of existing information.

In addition, the need to combine additional information with data from other non-health sources, for example for trend analysis, correlation data, decision making, etc., is often raised.

The extension of all these combined data from different sources represents the global knowledge put into play. On the other hand, the problem exposed does not only affect health and research groups, but also affects other sectors and actors that require massive data analysis from diverse and heterogeneous sources such as government agencies, insurance companies, pharmaceutical companies, agencies, etc.

Ensuring access to this global knowledge is crucial for achieving the achievement of research in the health, pharmaceuticals or insurance sectors among others, and achieving relevant successes in the launching of health and social policies, Clinical treatments, risk analysis and qualification or urban planning among others.

In this context, the ADAGIO (Analytics Data Aggregated Geolocation Open) project is born. The vision of the ADAGIO project is to combine strategies of Big Data and Machine Learning in areas of treatment of geolocated data of diverse sources and linked to the individual, to generate knowledge of value, starting from open data already available.

The main objective of the ADAGIO project is the development of an information system or platform that allows:

1. The aggregation, consolidation and standardization of data from different semantic fields: demographic, socio-economic or health among others
2. contextualized by environmental and manning data,
3. referenced by the geolocation of the target data,
4. where the data are obtained from heterogeneous sources and with different levels of aggregation,
5. and allowing the subsequent consultation based on specific variables: age, geographic area or proximity to an assistance center for a pathology among others,
6. so that specific algorithms facilitate geolocation obtaining and the desired aggregation level of target populations and contrast for research groups, companies, agencies, and other target audience.

Figure VIII-5 represents an overview of the Adagio subsystems.

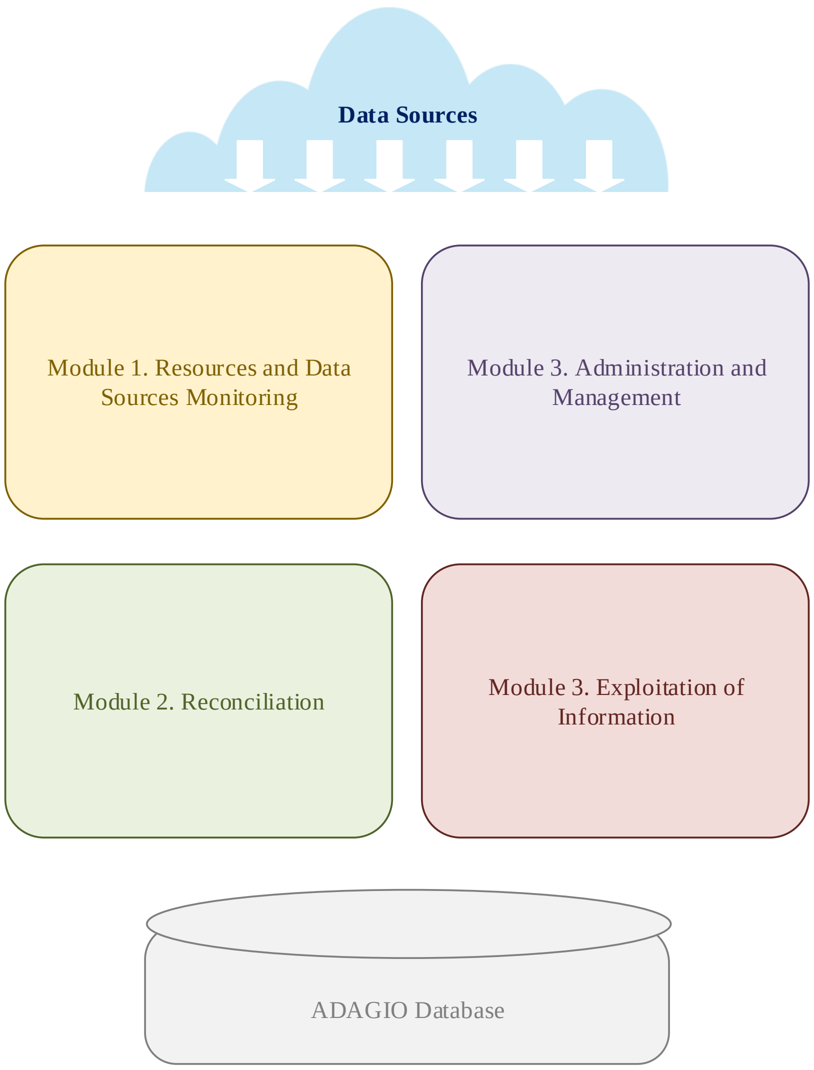

Data Sources
Module 1. Resources and Data Sources Monitoring
Module 3. Administration and Management
Module 2. Reconciliation
Module 3. Exploitation of Information
ADAGIO Database

Figure VIII-5. ADAGIO Architecture

The main function of this module 1 is based on the periodic downloading and treatment of the different data sources, accessing them and providing them to module 2 for their treatment.

The main purpose of the source reconciliation module is to perform an entity reconciliation process between all sources that are available and registered in ADAGIO. This process is based on a process of reconciliation, normalization of terms and extraction of geolocalized information from all available sources. All this will be done with the support of the database of the ADAGIO platform as well as the thesauri and metatesauros of the same. Once all the information has been normalized, the resulting information will be loaded into the databases defined in the ADAGIO platform for this purpose. The approach presented by this Doctoral Thesis, is being considered as one of the alternatives to cover the functionalities of the entity reconciliation process of Module 2.

The main objective of module 3 is based on the promotion of queries by users of the system, in a language as natural and high level as possible, applying supervised machine learning to the processing of written text when expressing the query, with text mining techniques.

Finally, the main functionalities of module 4 are: manage data sources, the modification of data source update periods and system reconciliation periods, the management of system users and the manual execution of the reconciliation module, allowing the user to launch the entity reconciliation process at any time.

2. CONCLUSIONS

This chapter has presented two real-world cases of study where the validation of the MaRIA tool has been performed.

In DIPHDA project, the problem covered is related to the data information management of the cultural heritage the community of the region of Andalusia, to the maintenance, updating and assurance of the quality of the data offered by their MOSAICO system in charge of the integral management of cultural heritage information. Although the system manages more information, the first solution has addressed the fixed monuments of the system.

The advantages that MaRIA tool offers to this problem are:

- Easy to use
- Ease of learning
- Information update at low cost
- Ensure the quality of the data generated
- Highly scalable with little effort

It was described how this real case study has been developed with the tool following the activities defined in the MaRIA Process described in Chapter IV.

In ADAGIO project, the objective is to combine strategies of Big Data and Machine Learning in areas of treatment of geolocated data of diverse sources and linked to the individual, to generate knowledge of value, starting from open data already available. The architecture of this project contains an entity reconciliation module where the proposal that this Doctoral Thesis presents, is being considered among one of the alternatives to carry out this process.

CHAPTER IX
CONTRIBUTION, FUTURE WORK AND CONCLUSIONS

CHAPTER IX. CONTRIBUTION, FUTURE WORK AND CONCLUSIONS

During the development of this Doctoral Thesis all the necessary elements to complete the MaRIA framework, which facilitates the definition and analysis of solutions to solve entity reconciliation problems in heterogeneous data sources within the information systems, have been addressed.

Initially, a comprehensive study of the state of the art was carried out to determine the initial situation and, once the results obtained were analyzed, work began on this framework, detailed in the present Doctoral Thesis.

To solve the problem we opted for a solution based on the MDE paradigm, whose framework is supported by 4 main pillars: (i) the MaRIA process, which defines a set of activities that should be included in the software development methodology of the organization that uses it and, on the other hand, offers the software engineer a procedure to solve the problem of reconciliation of entities of the software system to be developed, (ii) a modeling language that allows the definition, analysis and testing of the solution, (iii) derivation mechanisms that make it possible to transform defined models in business rules so that the designed model is testable and (iv) as a practical feasibility test of the defined framework, the MaRIA tool. This tool has been validated in a real scenario of the projects DIPHDA, in collaboration with Fujitsu Laboratories of Europe (FLE) and the culture council of the Junta de Andalucía and ADAGIO, a CDTI project of the Ministry of Economy and Competitiveness in collaboration with the Servinform organization.

In summary, all this material constitutes the work done in this Doctoral Thesis. This chapter concludes by describing the strategic research framework in which this Doctoral Thesis has been developed (section 1), which has influenced the results obtained, describing the specific contributions regarding (Section 2), and proposing a set of future lines of research in which to continue working on this line of research (section 3).

1. RESEARCH FRAMEWORK OF THIS PHD THESIS

IWT2 research group presents, as one of its main strategic lines, the successful combination of the Model-Driven Engineering paradigm with the management of information in multiple areas, with the objective of solving needs that have been identified. The fact of having a line on the application of the MDE paradigm allows to harmonize proposals and approaches, provoking that the results of a research work support the hypotheses of the following, advancing together.

The objectives of this line of research include the need to offer solutions within the framework of data management, facilitating the software engineer to support the entity reconciliation in the requirements and analysis phases of a software development preparing the system to guarantee a suitable entity reconciliation with, besides could be systematically tested., using the MDE paradigm.

In addition, as result of this Doctoral Thesis, within the framework of the IWT2 group, an end of degree works has been performed in the field of the development of a system that let a software engineer to design and analyze an entity reconciliation problem in heterogeneous data sources using the MDE paradigm.

Finally, this Doctoral Thesis has been partially sponsored by Fujitsu Laboratories of Europe (FLE). Also, it has been carried out a project that takes this proposal as a basis for its development in collaboration with this organization.

2. CONTRIBUTIONS

This second section recapitulates the main contributions of the present Doctoral Thesis to the scientific community, referring to the objectives that were raised at the beginning of the research, proving that there is at least one correspondence for each objective and concluding that all the objectives have been cutlery.

In the development of the research carried out in this Doctoral Thesis, it was fundamental to know the current situation regarding the existing solutions to solve entity reconciliation problems in heterogeneous data sources. To know the current situation, a Systematic Mapping Study (SMS) was carried out to first, trying to (i) understand the state-of-the-art of the problem and (ii), identifying any gaps in current research. This SMS was published in (Enríquez, J.G. et al., 2017)

A characterization scheme (Table IX-1) was created to achieve these goals. This characterization was divided in three groups: UML, ER challenges and type of datasets.

Following Unified Modeling Language (UML) specification (Group, 2017) that classifies diagrams in two categories: (i) structure-based diagrams, which show the static structure of the system and its parts on different abstraction and implementation levels and how they are related to each other, (ii) and behavior-based diagrams, which show the dynamic behavior of the objects in a system extrapolating it to our problem, it was decided to categorize the proposals found in two big groups: design and operation.

Challenges for ER proposed in Getoor and Machanavajjhala, (2013), multi-relational, dealing with structure of entities, multi-domain, dealing with customizable methods that span across domains and multi-applications, dealing with systems that serve diverse application with different accuracy requirements, level of automation of the proposal.

Finally, the types of dataset that were used for the validation of the proposals, understanding them as heterogeneous or non-heterogeneous.

	Objective
UML	Design
	Operation
ER Challenges	Automation
	Multi-Relational
	Multi-Domain
	Multi-Applications
Type of Dataset	Heterogeneous
	Non-heterogeneous

Table IX-1. Characterization scheme

The analysis of the results showed that the heterogeneity of the datasets is acceptable knowing that more than a half proposals use heterogeneous data sources to test them. Most of the research work has been focused on the operation phase of the reconciliation and not in the design phase. Finally, the efforts made to automate the process of reconciliation, and consider the multi-relational, multi-domain and multi-applications challenges, have been very limited.

This covers the objective set first in Chapter III, which was "*Perform a study of the state of the art of the different existing solutions for the entity reconciliation of heterogeneous data sources, checking if they are being used in real environments*".

2.1.MARIA FRAMEWORK

The main contribution made with the development of the present Doctoral Thesis is the Framework that allows software engineers to design, analyze and test entity reconciliation problems into any software development methodology. It is worth highlighting the great influence that the MDE paradigm had on the development of this reference framework, which has largely guided the solutions adopted.

Figure IX-1, shows a global view of the MaRIA Framework. Thanks to the proposed solution, the software engineers will be able to model their presumable entity reconciliation models for solving any type of entity reconciliation problem, understanding presumable, as the capacity of test the final solutions for checking the coverage level of the new generated dataset without more efforts than the design of the problem, all this, thanks to the integration of Early Testing.

MaRIA Framework has been designed so that it can be integrated in any methodology of software development, whether they have classic or agile life cycles. Considering that any software development methodology is composed of several phases, the scope of this Doctoral Thesis has been set up to cover the requirements and analysis phases. Also, the testing phase has been considered adding Early Testing to allow the models to be systematically tested.

MaRIA Framework is composed of three fundamental pillars: “MaRIA process”, “Model-Driven Approach” and “MaRIA Tool”.

“MaRIA process” defines the set of activities that must be added and performed in any software development methodology to develop a solution to an entity reconciliation problem. “Model-Driven Approach” is defined by the metamodel that allows the software engineer that use this framework to model the entity reconciliation problem and derivation mechanisms that to allow the models defined to be systematically tested. “MaRIA Tool” is the support tool developed to give support to the MaRIA Framework.

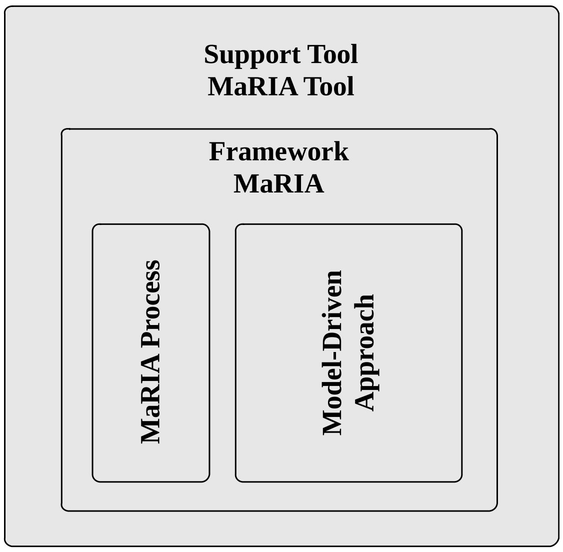


Figure IX-1. MaRIA Framework.

This covers the objective set second in Chapter III, which was “*Define and develop a Framework for designing the entity reconciliation models by a systematic way for the requirements and analysis phases*”.

2.2. MARIA TOOL

One of the main pretensions since the beginning of the development of this Doctoral Thesis was that MaRIA Framework could be put into practice to be able to use it in real scenarios. To achieve this objective, the need to develop a support tool that implements the MaRIA Framework encompassing the 3 fundamental pillars that comprise it was raised.

The resulting tool, named MaRIA Tool (Figure IX-2), was developed based on a Domain Specific Language that implements the metamodel defined in the Model-Driven pillar of MaRIA Framework.

This covers the objective set third in Chapter III, which was *“Provide a support tool for the framework”.* This objective is covered in Chapter VII. The support will allow to a software engineer to define the analysis model of an entity reconciliation problem between different and heterogeneous data sources. The tool will be represented as a Domain Specific Language (DSL)”.

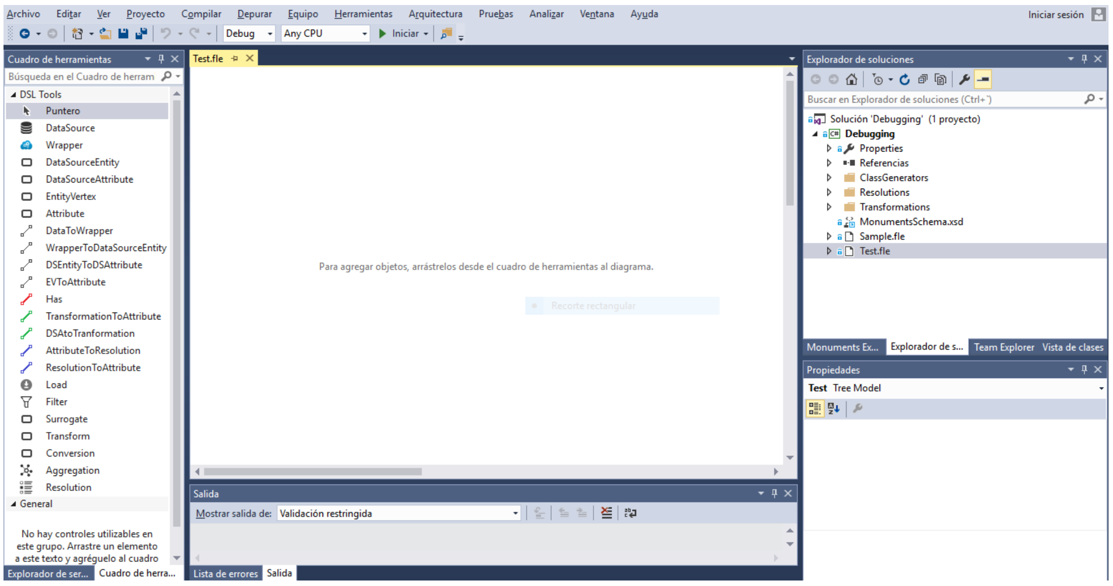


Figure IX-2. MaRIA Tool

To evaluate MaRIA tool, two real-world use cases have been taken as reference: (i) DIPHDA project, in collaboration with the Government of Andalusia and Fujitsu Laboratories of Europe that aimed to reconcile historical patrimony information related to the monuments of the region, and (ii) ADAGIO, a CDTI project in collaboration with Servinform company that aimed to develop an information system or platform that allows: the aggregation, consolidation and standardization of data from different semantic fields, contextualized by environmental and manning data.

This covers the last objective set forth in Chapter III, which was "*Evaluate the results obtained of the application of the proposal in a real-world case study*".

Finally, two more contributions have been achieved: the real validation and the transfer of the knowledge. As mentioned before, this Doctoral Thesis has been applied in two real world case studies (DIPHDA and ADAGIO) transferring the knowledge generated during the development of this work to the companies.

3. Future Work and New Research Lines

After recapitulating the main contributions made during the development of this Doctoral Thesis, it is important to emphasize that its results lead to the opening of new research lines that will allow us to advance in the selected topic. Considering the objectives defined in Chapter III for this Doctoral Thesis, future work encompasses several avenues.

Table IX-2 describes the relation between the objectives proposed for this Doctoral Thesis and the solution proposed each one and the future work and new research lines that emerges from this relation.

Objective	Solution	Future Work and New Research Lines
Perform a study of the state of the art of the different existing solutions for the entity reconciliation of heterogeneous data sources, checking if they are being used in real environments	*Entity Reconciliation in Big Data Sources: a Systematic Mapping Study (Enríquez, J.G. et al., 2017)*	**New Research Lines** • Increase level of automation in solution for solving entity reconciliation problems in heterogeneous data sources • Multi-applications problems, where different applications with different requirements need to be served with the results of the reconciliation process
		Future Work • Include new searches to increase the domains included in the developed SMS • Continue developing this king of studies to keep it updated and do not let it become obsolete
Define and develop a Framework for designing the entity reconciliation models by a systematic way for the requirements and analysis phases	*MaRIA Framework*	**New Research Lines** • Apply MaRIA Framework in other software development methodology such as SCRUM (Szalvay, 2004) or BDD (Lazăr et al., 2010) among others.
		Future Work • Extend NDT-Methodology to cover, besides the requirements, analysis and testing processes, all the set of processes that define it • Extend Testing set of processes of NDT-Methodology to: o Cover the unit testing of the transformations applied over the data to carry out the entity reconciliation o Generate automatically the insert queries that cover test requirements
Provide a support tool for the Framework	*MaRIA Tool*	**New Research Lines** • Extend the tool to cover the new set of processes that gives support to the extended MaRIA Framework
		Future Work • Encompass more transformation operations • Offer a big level of abstraction
Evaluate the results obtained of the application of the proposal in a real-world case study	*DIPHDA and ADAGIO projects*	**Future Work** • Work closely to the companies to keep improving this approach and solve new problems of different domains

Table IX-2. Future work and new research lines

4. CONCLUSIONS

Previous chapters of this PhD thesis hold up the work where we present and define a theoretical and practical Framework. They also describe a work based on a real need in organizations (Chapter I), that later has turned into a specific problem (Chapter III) derived from the results and conclusions obtained after studying the state-of-the-art (Chapter II).

Once the context has been specified, the remaining PhD thesis introduces a Framework composed three main pillars: (i) the MaRIA Process (Chapter IV), (ii) the Model-Driven approach (Chapter V) to support the entity reconciliation modeling preparing the system to be developed to be systematically tested defining a set of transformation mechanisms (Chapter VI) and (iii), a support tool to cover the previous two pillars called MaRIA Tool (Chapter VII).

To apply the theoretical framework to real environments. In this sense this approach has been put in practice in two real world case studies presented in Chapter VIII.

This Doctoral Thesis propose a suitable environment to support the entity reconciliation in the requirements and analysis phases. This environment allows the development team to prepare their future system to guarantee a suitable entity reconciliation with, besides could be systematically tested.

REFERENCES

REFERENCES

Ali, O., Cristianini, N., 2010. Information fusion for entity matching in unstructured data. IFIP Adv. Inf. Commun. Technol. 339 AICT, 162–169. doi:10.1007/978-3-642-16239-8_23

Axelos, 2016. PRINCE2 Certification | AXELOS [WWW Document]. PRINCE2 Qualif. URL https://www.axelos.com/qualifications/prince2-qualifications

Ayat, N., Akbarinia, R., Afsarmanesh, H., Valduriez, P., 2014. Entity resolution for probabilistic data. Inf. Sci. (Ny). 277, 492–511. doi:10.1016/j.ins.2014.02.135

Ayat, N., Akbarinia, R., Afsarmanesh, H., Valduriez, P., 2013. Entity resolution for distributed probabilistic data. Distrib. Parallel Databases 31, 509–542.

Balaji, J., Javed, F., Kejriwal, M., Min, C., Sander, S., Ozturk, O., 2016. An Ensemble Blocking Scheme for Entity Resolution of Large and Sparse Datasets.

Beheshti, S.-M.-R., Benatallah, B., Venugopal, S., Ryu, S.H., Motahari-Nezhad, H.R., Wang, W., 2016. A systematic review and comparative analysis of cross-document coreference resolution methods and tools. Computing 1–37. doi:10.1007/s00607-016-0490-0

Beizer, B., 1995. Black-box testing: techniques for functional testing of software and systems, Software Testing Verification and Reliability.

Bézivin, J., 2005. On the unification power of models. Softw. Syst. Model. 4, 171–188. doi:10.1007/s10270-005-0079-0

Bézivin, J., Hillairet, G., Jouault, F., Kurtev, I., Piers, W., 2005. Bridging the MS/DSL Tools and the Eclipse Modeling Framework. Meta San Diego,.

Bhattacharya, I., Getoor, L., 2005. A latent dirichlet allocation model for entity resolution. Proc. 2005 SIAM Int. Conf. Data Min. 47–58.

Blanco, R., Tuya, J., Seco, R. V., 2012a. Test adequacy evaluation for the user-database interaction: A specification-based approach, in: Proceedings - IEEE 5th International Conference on Software Testing, Verification and Validation, ICST 2012. pp. 71–80. doi:10.1109/ICST.2012.87

Blanco, R., Tuya, J., Seco, R. V., 2012b. Test adequacy evaluation for the user-database interaction: A specification-based approach, in: Proceedings - IEEE 5th International Conference on Software Testing, Verification and Validation, ICST 2012. pp. 71–80. doi:10.1109/ICST.2012.87

Bortoli, S., Bouquet, P., Bazzanella, B., 2014. An Identification Ontology for Entity Matching. Move to Meaningful Internet Syst. Otm 2014 Work. 8842, 587–596.

Brambilla, M., Cabot, J., Wimmer, M., 2012. Model-Driven Software Engineering in Practice, Synthesis Lectures on Software Engineering.

doi:10.2200/S00441ED1V01Y201208SWE001

Brando, C.., Frontini, F.. b, Ganascia, J.-G.., 2015. Disambiguation of named entities in cultural heritage texts using linked data sets. Commun. Comput. Inf. Sci. 539, 505–514. doi:10.1007/978-3-319-23201-0_51

Bratus, S., Rumshisky, A., Khrabrov, A., Magar, R., Thompson, P., 2011. Domain-specific entity extraction from noisy, unstructured data using ontology-guided search. Int. J. Doc. Anal. Recognit. 14, 201–211. doi:10.1007/s10032-011-0149-5

Brizan, D.G., Tansel, A.U., 2006. A Survey of Entity Resolution and Record Linkage Methodologies. Commun. IIMA 6, 41–50.

Calvanese, D., Keet, C.M., Nutt, W., Rodr'\iguez-Muro, M., Stefanoni, G., 2010. Web-based graphical querying of databases through an ontology: the Wonder system. Proc. 2010 ACM Symp. Appl. Comput. 1388–1395. doi:10.1145/1774088.1774384

Cartlidge, A., Hanna, A., Rudd, C., Ivor, M., Stuart, R., 2007. An introductory overview of ITIL V3, The UK Chapter of the itSMF. doi:10.1080/13642818708208530

Carvalho, L.F.M., Laender, A.H.F., Meira Jr, W., 2015. Entity Matching: A Case Study in the Medical Domain, in: Alberto Mendelzon International Workshop on Foundations of Data Management. p. 57.

Ceri, S., Fraternali, P., Bongio, A., 2000. Web modeling language (WebML): a modeling language for designing Web sites. Comput. Networks 33, 137–157. doi:10.1016/S1389-1286(00)00040-2

Certification Europe, 2014. ISO 27001 Information Security Certification. ISO 27001 Inf. Secur.

Chilenski, J.J., 2001. An investigation of three forms of the modified condition decision coverage (MCDC) criterion. Security.

Cohen, W.W., Richman, J., 2002. Learning to match and cluster large high-dimensional data sets for data integration. Proc. eighth ACM SIGKDD Int. Conf. Knowl. Discov. data Min. 475–480. doi:10.1145/775107.775116

Cook, S., Jones, G., Kent, S., Wills, A.C., 2007. Domain Specific Development with Visual Studio DSL Tools, Library.

Costa, G., Cuzzocrea, A., Manco, G., Ortale, R., 2011. Data de-duplication: A review. Stud. Comput. Intell. 375, 385–412. doi:10.1007/978-3-642-22913-8_18

Costa, G.D.A., 2016. Large-Scale Entity Resolution for Semantic Web Data Integration LARGE-SCALE ENTITY RESOLUTION FOR SEMANTIC.

de Assis Costa, G., de Oliveira, J.M.P., 2016. A Blocking Scheme for Entity Resolution in the Semantic Web, in: Advanced Information Networking and Applications (AINA), 2016 IEEE 30th International Conference on. pp. 1138–1145.

Dharavath, R., Kumar, C., 2015. The Journal of Systems and Software Entity resolution based EM for integrating heterogeneous distributed probabilistic data. J. Syst. Softw. 107, 93–109. doi:10.1016/j.jss.2015.05.035

Dharavath, R., Singh, A.K., 2016. Entity Resolution-Based Jaccard Similarity Coefficient for Heterogeneous Distributed Databases, in: Proceedings of the Second International Conference on Computer and Communication Technologies. pp. 497–507.

Domínguez-Mayo, F.J., Escalona, M.J., Mejías, M., Ross, M., Staples, G., 2014. Towards a homogeneous characterization of the model-driven web development methodologies.

Dorneles, C.F., Gonçalves, R., dos Santos Mello, R., 2011. Approximate data instance matching: A survey. Knowl. Inf. Syst. 27, 1–21. doi:10.1007/s10115-010-0285-0

Eclipse, 2016a. Eclipse Modeling Framework and Java Emitter Templates [WWW Document]. http//www.eclipse.org/emf, Last Accesed December 201. URL http://www.eclipse.org/emf

Eclipse, 2016b. Eclipse Graphical Modeling Framework.

Efthymiou, V., Efthymiou, V., Papadakis, G., Papastefanatos, G., Stefanidis, K., Palpanas, T., 2016a. Parallel Meta-blocking for Scaling Entity Resolution over Big Heterogeneous Data. Inf. Syst. 65, 137–157. doi:10.1016/j.is.2016.12.001

Efthymiou, V., Stefanidis, K., Vassilis, C., 2016b. Minoan ER : Progressive Entity Resolution in the Web of Data, in: 19th International Conference on Extending Database Tech- Nology, EDBT 2016. pp. 670–671. doi:10.5441/002/edbt.2016.79

Enríquez, J.G., Domínguez-Mayo, F.J., Escalona, M.J., García-García, J.A., Lee, V., Masatomo, G., 2015. Entity Identity Reconciliation based Big Data Federation-A MDE approach, in: International Conference on Information Systems Development (ISD2015).

Escalona, M.J.., Gutiérrez, J.J.., Ortega, J.A.., Ramos, I.., Aragón, G.., 2008. {NDT} & Metrica V3 an approach for public organizations based on model driven engineering, in: {WEBIST} 2008 - 4th International Conference on Web Information Systems and Technologies, Proceedings. pp. 224–227.

Escalona, M.J., Aragón, G., 2008. NDT. A model-driven approach for web requirements. IEEE Trans. Softw. Eng. 34, 377–394. doi:10.1109/TSE.2008.27

Fan, W., Geerts, F., Tang, N., Yu, W., 2014. Conflict Resolution with Data Currency and Consistency. J. Data Inf. Qual. 5, 6:1--6:37. doi:10.1145/2631923

Fan, W., Li, J., Ma, S., Tang, N., Yu, W., 2011. Interaction between record matching and data repairing, Proceedings of the 2011 international conference on Management of data - SIGMOD ’11. doi:10.1145/1989323.1989373

Fellegi, I.P., Sunter, A.B., 1969. A Theory for Record Linkage. Source J. Am. Stat. Assoc. 64, 1183–1210. doi:10.1080/01621459.1969.10501049

Firmani, D., Saha, B., Srivastava, D., 2016. Online entity resolution using an oracle. Proc. VLDB Endow. 9, 384–395.

Fisher, J., Christen, P., Wang, Q., Rahm, E., 2015. A Clustering-Based Framework to Control Block Sizes for Entity Resolution. Proc. 21th ACM SIGKDD Int. Conf. Knowl. Discov. Data Min. - KDD ’15 279–288. doi:10.1145/2783258.2783396

Fondement, F., Baar, T., 2007. Concrete syntax definition for modeling languages. ISIM-LGL. doi:10.5075/epfl-thesis-3927

Frankel, D.S., 2003. Model Driven Architecture: Applying MDA to Enterprise Computing, Journal of Chemical Information and Modeling. doi:10.1017/CBO9781107415324.004

Frontini, F., Brando, C., Ganascia, J., Frontini, F., Brando, C., Domain-adapted, J.G., 2015. Domain-adapted named-entity linker using Linked Data To cite this version :

Gaikwad, S., 2015. A Survey Analysis On Duplicate Detection in Hierarchical Data 0, 1–6.

Gal, A., 2014. Uncertain Entity Resolution: Re-evaluating Entity Resolution in the Big Data Era: Tutorial. Proc. VLDB Endow. 7, 1711–1712. doi:10.14778/2733004.2733068

Galhardas, H., Florescu, D., Shasha, D., Simon, E., Saita, C.-A., 2001. Declarative Data Cleaning: Language, Model, and Algorithms. Proc. 27th Int. Conf. Very Large Data Bases 371–380.

García-Borgoñón, L., 2015. Un marco de referencia para facilitar la interoperabilidad y mantenibilidad de los modelos de procesos de software.

Genero, M., Cruz-Lemus, J.A., Piattini, M., 2014. Métodos de Investigación en Ingeniería del Software., RA-MA Editorial.

Getoor, L., Machanavajjhala, A., 2013a. Entity resolution for big data, in: Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining - KDD ’13. p. 1527. doi:10.1145/2487575.2506179

Getoor, L., Machanavajjhala, A., 2013b. Entity resolution for big data. KDD ’13 Proc. 19th ACM SIGKDD Int. Conf. Knowl. Discov. data Min. 4503. doi:10.1145/2487575.2506179

Getoor, L., Machanavajjhala, A., 2012. Entity resolution: Theory, practice & open challenges. Proc. VLDB Endow. 5, 2018–2019. doi:10.14778/2367502.2367564

Group, O.M., 2017. OMG Unified Modeling Language TM (OMG UML), Version 2.5. InformatikSpektrum 21, 758. doi:10.1007/s002870050092

Gu, Q., Zhang, Y., Cao, J., Xu, G., Cuzzocrea, A., 2014. A confidence-based entity resolution approach with incomplete information, in: Data Science and Advanced Analytics (DSAA), 2014 International Conference on. pp. 97–103. doi:10.1109/DSAA.2014.7058058

Hay, D., Healy, K.A., 2000. Defining business rules: what are they really? Final Rep. 34. doi:10.1016/j.ijinfomgt.2003.12.007

Hermansson, L., Kerola, T., Johansson, F., Jethava, V., Dubhashi, D., 2013. Entity disambiguation in anonymized graphs using graph kernels. Proc. 22nd ACM Int. Conf. Conf. Inf. Knowl. Manag. - CIKM ’13 1037–1046. doi:10.1145/2505515.2505565

Hernández, M., Koutrika, G., 2013. HIL: a high-level scripting language for entity integration. Proc. 16th … 549–560. doi:10.1145/2452376.2452440

Horrocks, I., Patel-schneider, P.F., Boley, H., Tabet, S., Grosof, B., Dean, M., 2004. SWRL : A Semantic Web Rule Language Combining OWL and RuleML. W3C Memb. Submiss. 21 1–20.

Hsueh, S.C., Lin, M.Y., Chiu, Y.C., 2014. A load-balanced mapreduce algorithm for blocking-based entity-resolution with multiple keys. Conf. Res. Pract. Inf. Technol. Ser. 152, 3–9.

Instituto Andaluz de Patrimonio Histórico, 2010. La Gestión de la Información del Patrimonio Cultural de Andalucía.

Ioannou, E., Nejdl, W., Niedere’e, C., Velegrakis, Y., Nieder, C., 2010. On-the-Fly Entity-Aware Query Processing in the Presence of Linkage. Proc. VLDB Endow. 3, 429–438. doi:10.14778/1920841.1920898

Irmak, U., Kraft, R., 2010. A scalable machine-learning approach for semi-structured named entity recognition. Proc. 19th Int. Conf. World wide web WWW 10 461. doi:10.1145/1772690.1772738

Iso, 2008. BS EN ISO 9001:2008, Text.

ISO/IEC/IEEE, 2013. INTERNATIONAL STANDARD ISO/IEC/IEEE 29119 2013.

ISO/IEC JTC 1, 2014. ISO/IEC CD 20546 - Big data report 31, 498–513. doi:10.1007/978-0-387-35081-3

J.G. Enríquez, F.J. Domínguez-Mayo, M.J. Escalona, M. Ross, G. Staples, 2017. Entity Reconciliation in Big Data Sources: a Systematic Mapping Study. Expert Syst. Appl. 80, 14–27. doi:http://dx.doi.org/10.1016/j.eswa.2017.03.010

Kejriwal, M., 2014. Populating Entity Name Systems for Big Data. Proc. 13th Int. Semant. Web Conf. 521–528.

Kitchenham, B.A., Budgen, D., Pearl Brereton, O., 2011. Using mapping studies as the basis for further research - A participant-observer case study. Inf. Softw. Technol. 53, 638–651. doi:10.1016/j.infsof.2010.12.011

Kitchenham, B., Brereton, P., 2013. A systematic review of systematic review process research in software engineering. Inf. Softw. Technol. 55, 2049–2075. doi:10.1016/j.infsof.2013.07.010

Kitchenham, B., Charters, S., 2007. Guidelines for performing Systematic Literature Reviews in Software Engineering. Engineering 2, 1051. doi:10.1145/1134285.1134500

Koch, N., Knapp, A., Zhang, G., Baumeister, H., 2008. UML-based web engineering: An Approach Based on Standards. Web Eng. Model. Implement. Web Appl. 157–191.

Kolb, L., Köpcke, H., Thor, A., Databases, H.S.D., Systems, H., 2011. Learning-based Entity Resolution with MapReduce. Cikm 1–6. doi:http://doi.acm.org/10.1145/2064085.2064087

Kolovos, D., 2016. The Eclipse Modeling (EMF) and Graphical Modeling (GMF) Frameworks. York.

Kong, C., Gao, M., Xu, C., Quian, W., Zhou, A., 2016. Entity Matching Across Multiple Heterogeneous Data Sources, in: Proceedings of the 21st International Conference on Database Systems for Advanced Applications (DASFAA 2016). pp. 133–146. doi:10.1007/978-3-319-32025-0

Lazăr, I., Motogna, S., Pârv, B., 2010. Behaviour-driven development of foundational UML components, in: Electronic Notes in Theoretical Computer Science. pp. 91–105. doi:10.1016/j.entcs.2010.07.007

Lee, H., Chang, A., Peirsman, Y., Chambers, N., Surdeanu, M., Jurafsky, D., 2013. Deterministic Coreference Resolution Based on Entity-Centric, Precision-Ranked Rules. Comput. Linguist. 39, 885–916. doi:10.1162/COLI

Leitão, L., Calado, P., Herschel, M., 2013. Efficient and Effective Duplicate Detection in Hierarchical Data. IEEE Trans. Knowl. Data Eng. 25, 1028–1041. doi:10.1109/TKDE.2012.60

Li, L., Li, J., Gao, H., 2015. Rule-based method for entity resolution. IEEE Trans. Knowl. Data Eng. 27, 250–263. doi:10.1109/TKDE.2014.2320713

Li, L., Li, J., Wang, H., Gao, H., 2011. Context-based Entity Description Rule for Entity Resolution. Cikm 1725–1730. doi:10.1145/2063576.2063825

Lin, Y., Wang, H., Li, J., Gao, H., 2016. Efficient Entity Resolution on Heterogeneous Records. doi:1610.09500

Liu, H., Kumar, T.K.A., Thomas, J.P., 2015. Cleaning Framework for Big Data - Object Identification and Linkage. 2015 IEEE Int. Congr. Big Data 215–221. doi:10.1109/BigDataCongress.2015.38

Liu, Y., Wang, H., Gao, H., 2011. A fast entity resolution method based on wave of records. 2011 Int. Conf. Consum. Electron. Commun. Networks, CECNet 2011 - Proc. 4642–4645. doi:10.1109/CECNET.2011.5768200

Maddodi, S., Attigeri, G. V., Karunakar, A.K., 2010. Data Deduplication techniques and analysis, in: Proceedings - 3rd International Conference on Emerging Trends in Engineering and Technology, ICETET 2010. pp. 664–668. doi:10.1109/ICETET.2010.42

Mazanek, S., 2011. HelloWorld! An Instructive Case for the Transformation Tool Contest, in: Gorp, P. Van, Mazanek, S., Rose, L.M. (Eds.), Proceedings Fifth Transformation Tool Contest, {TTC} 2011, Z{ü}rich, Switzerland, June 29-30 2011., EPTCS. pp. 22–26. doi:10.4204/EPTCS.74.4

McCallum, A., Nigam, K., Ungar, L.L.H., 2000. Efficient clustering of high-dimensional data sets with application to reference matching. Proc. sixth ACM SIGKDD Int. Conf. Knowl. Discov. data Min. 169–178. doi:10.1145/347090.347123

Meliá, S., Gómez, J., Pérez, S., Díaz, O., 2008. A model-driven development for GWT-based rich internet applications with OOH4RIA, in: Proceedings - 8th International Conference on Web Engineering, ICWE 2008. pp. 13–23. doi:10.1109/ICWE.2008.36

Mellor, S., Scott, K., Uhl, A., Weise, D., 2004. MDA Distilled - Principles of Model Driven Architecture. Addison Wesley.

Mendeley Support Team, 2011. Mendeley Guide. Mendeley Deskt. 1–16. doi:10.1063/1.476396

Metzger, A., 2008. A Systematic Look at Model Transformations. Nature 451, 644–647. doi:10.1038/451644a

Mondal, J., Deshpande, A., 2012. Managing large dynamic graphs efficiently. Proc. 2012 Int. Conf. Manag. Data - SIGMOD '12 145. doi:10.1145/2213836.2213854

MOSAICO, 2016. MOSAICO: Sistema de Información para la Gestión del Patrimonio Cultural en Andalucía (http://www.juntadeandalucia.es/cultura/web/areas/bbcc/sites/consejeria/areas/bbcc/contenidos/Mosaico/sistema_gestion_bienes_culturales). Last Access December 2016.

MSDN, 2016. Domain-Specific Languages (DSL-Tools).

Ngai, E.W.T., Hu, Y., Wong, Y.H., Chen, Y., Sun, X., 2011. The application of data mining techniques in financial fraud detection: A classification framework and an academic review of literature. Decis. Support Syst. 50, 559–569. doi:10.1016/j.dss.2010.08.006

Nguyen, K., Ichise, R., 2016. Linked data entity resolution system enhanced by configuration learning algorithm. IEICE Trans. Inf. Syst. 99, 1521–1530.

Nuray-turan, R., Kalashnikov, D. V, Mehrotra, S., 2013. Adaptive Connection Strength Models for Relationship-Based Entity Resolution. Jdiq 4, 22. doi:10.1145/2435221.2435224

Oldevik, J., Neple, T., Grønmo, R., Aagedal, J., Berre, A.J., 2005. Toward standardised model to text transformations, in: Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics). pp. 239–253. doi:10.1007/11581741_18

Omg, Q., 2008. Meta Object Facility (MOF) 2 . 0 Query / View / Transformation

Specification. Transformation 1–230. doi:citeulike-article-id:1557124

Otero-Cerdeira, L., Rodríguez-Martínez, F.J., Gómez-Rodríguez, A., 2015. Ontology matching: A literature review. Expert Syst. Appl. 42, 949–971. doi:10.1016/j.eswa.2014.08.032

Papadakis, G., Ioannou, E., Niederée, C., Fankhauser, P., 2011a. Efficient entity resolution for large heterogeneous information spaces. Proc. fourth ACM Int. Conf. Web search data Min. - WSDM '11 535. doi:10.1145/1935826.1935903

Papadakis, G., Ioannou, E., Niederée, C., Palpanas, T., Nejdl, W., 2012. Beyond 100 million entities: large-scale blocking-based resolution for heterogeneous data. Proc. fifth ACM Int. Conf. Web search data Min. 53–62. doi:10.1145/2124295.2124305

Papadakis, G., Ioannou, E., Niederée, C., Palpanas, T., Nejdl, W., 2011b. To Compare or Not to Compare: Making Entity Resolution More Efficient. Proc. Int. Work. Semant. Web Inf. Manag. 3:1--3:7. doi:10.1145/1999299.1999302

Papadakis, G., Svirsky, J., Gal, A., Palpanas, T., 2016. Comparative Analysis of Approximate Blocking Techniques for Entity Resolution. Pvldb 9, 684–695. doi:10.14778/2947618.2947624

PMI, 2008. A Guide to the Project Management Body of Knowledge (PMBOK Guide), Management. doi:10.1002/pmj.20125

Ponce, J., Escalona, M.J., Gómez, A., Luque, M., Molina, A., 2010. Definición de una política de pruebas en la gestión cultural : aplicación al desarrollo del proyecto Mosaico 6, 25–43.

Priya, P.A., Prabhakar, S., Vasavi, S., 2015. Entity resolution for high velocity streams using semantic measures. Adv. Comput. Conf. (IACC), 2015 IEEE Int. 35–40.

Rahmani, H., Ranjbar-Sahraei, B., Weiss, G., Tuyls, K., 2016. Entity resolution in disjoint graphs: an application on genealogical data. Intell. Data Anal. 20, 455–475.

Ramadan, B., Christen, P., Liang, H., 2014. Dynamic Sorted Neighborhood Indexing for Real-Time Entity Resolution. Databases Theory Appl. Adc 2014 8506, 1–12. doi:10.1145/2816821

Rossi, G., Schwabe, D., Cowan, D.D., 1995. An Object-Oriented Model for Designing the Human-Computer Interface Of Hypermedia Applications, in: In International Workshop on Hypermedia Design. pp. 131–152. doi:10.1.1.48.1222

Sarawagi, S., Bhamidipaty, A., 2002. Interactive Deduplication using Active Learning. Proc. eighth ACM SIGKDD Int. Conf. Knowl. Discov. data Min. 269–278. doi:10.1145/775047.775087

Schmidt, D.C., 2006. Guest Editor's Introduction : Model-Driven Engineering. IEEE Comput. 39, 25–31. doi:http://doi.ieeecomputersociety.org/10.1109/MC.2006.58

Schwaber, K., Beedle, M., 2001. Agile Software Development with Scrum, cdswebcernch. doi:10.1109/2.947100

Selenium, 2017. Selenium website documentation, last access February. RA-MA Editor.

Shen, W., Han, J., Wang, J., 2014. A Probabilistic Model for Linking Named Entities in Web Text with Heterogeneous Information Networks. Proc. 2014 ACM SIGMOD Int. Conf. Manag. data 1199–1210. doi:10.1145/2588555.2593676

Shin, K., Jung, J., Lee, S., Kang, U., 2015. BEAR: Block Elimination Approach for Random Walk with Restart on Large Graphs, in: Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data, SIGMOD '15. ACM, New York, NY, USA, pp. 1571–1585. doi:10.1145/2723372.2723716

Shu, L., Chen, A., Xiong, M., Meng, W., 2011. Efficient Spectral neighborhood blocking for entity resolution. Proc. - Int. Conf. Data Eng. 1067–1078. doi:10.1109/ICDE.2011.5767835

Shu, L., Lin, C., Meng, W., Han, Y., Yu, C.T., Smalheiser, N.R., 2012. A framework for entity resolution with efficient blocking, in: Information Reuse and Integration (IRI), 2012 IEEE 13th International Conference on. pp. 431–440. doi:10.1109/IRI.2012.6303041

Sleeman, J., Finin, T., 2013. Entity Type Recognition for Heterogeneous Semantic Graphs. 2013 AAAI Fall Symp. Ser. 63–67. doi:10.1109/ICSC.2013.22

Software Engineering Institute, 2010. CMMI for Development, Version 1.3. Carnegie Mellon Univ. 482. doi:CMU/SEI-2010-TR-033 ESC-TR-2010-033

Song, D., Luo, Y., Heflin, J., 2016. Linking Heterogeneous Data in the Semantic Web Using Scalable and Domain-Independent Candidate Selection. IEEE Trans. Knowl. Data Eng. PP, 143–156. doi:10.1109/TKDE.2016.2606399

Sun, C., Shen, D., Kou, Y., Nie, T., Yu, G., 2014. ERGP: A Combined Entity Resolution Approach with Genetic Programming. 2014 11th Web Inf. Syst. Appl. Conf. 215–220. doi:10.1109/WISA.2014.46

Sun, Z., Xiao, N., Liu, F., Fu, Y., 2014. DS-Dedupe: A scalable, low network overhead data routing algorithm for inline cluster deduplication system. 2014 Int. Conf. Comput. Netw. Commun. ICNC 2014 895–899. doi:10.1109/ICCNC.2014.6785456

Szalvay, V., 2004. An introduction to agile software development. Danube Technol.

Thiry, L., Thirion, B., 2009. Functional metamodels for systems and software. J. Syst. Softw. 82, 1125–1136. doi:10.1016/j.jss.2009.01.042

Trujillo, J., Luján-Mora, S., 2003. A UML based approach for modeling ETL processes in data warehouses. Concept. Model. 2003 2813, 307–320. doi:ETL processes, Data warehouses, conceptual modeling, UML

Tuya, J., Suárez-Cabal, M.J., De La Riva, C., 2010. Full predicate coverage for testing SQL database queries. Softw. Test. Verif. Reliab. 20, 237–288. doi:10.1002/stvr.424

Vasudevan, N., Tratt, L., 2011. Comparative study of DSL tools. Electron. Notes Theor. Comput. Sci. 264, 103–121. doi:10.1016/j.entcs.2011.06.007

Verykios, V.S., Moustakides, G. V., Elfeky, M.G., 2003. A Bayesian decision model for cost optimal record matching. VLDB J. 12, 28–40. doi:10.1007/s00778-002-0072-y

Vilaine, M., Ramos-Collado, F., Tejero-Fernández, J.A., Labrador del Río, I., 2012. Visual Studio DSL Tools VS. Eclipse. Cádiz.

Wang, F., Wang, H., Li, J., Gao, H., 2013. Graph-based reference table construction to facilitate entity matching. J. Syst. Softw. 86, 1679–1688. doi:10.1016/j.jss.2013.02.026

Wang, H., Li, J., Gao, H., 2016. Efficient Entity Resolution Based on Subgraph Cohesion. Knowl. Inf. Syst. 46, 285–314. doi:10.1007/s10115-015-0818-7

Wang, H., Li, J., Gao, H., 2015. Efficient entity resolution based on subgraph cohesion. Knowl. Inf. Syst. 285–314. doi:10.1007/s10115-015-0818-7

Wang, J., Oyama, S., Kurihara, M., Kashima, H., 2014. Learning an accurate entity resolution model from crowdsourced labels. Proc. 8th Int. Conf. Ubiquitous Inf. Manag. Commun. - ICUIMC '14 1–8. doi:10.1145/2557977.2558060

Wang, Q., Cui, M., Liang, H., 2016. Semantic-aware blocking for entity resolution. IEEE Trans. Knowl. Data Eng. 28, 166–180.

Whang, S.E., Garcia-Molina, H., 2014. Incremental entity resolution on rules and data. VLDB J. 23, 77–102. doi:10.1007/s00778-013-0315-0

Whang, S.E., Marmaros, D., Garcia-Molina, H., 2013. Pay-as-you-go entity resolution. IEEE Trans. Knowl. Data Eng. 25, 1111–1124. doi:10.1109/TKDE.2012.43

Willmor, D., Road, O., Embury, S.M., 2006. An Intensional Approach to the Specification of Test Cases for Database Applications, in: Proceedings of the 28th International Conference on Software Engineering. pp. 102–111. doi:10.1145/1134285.1134301

Winkler, W.E., 2002. Methods for Record Linkage and Bayesian Networks. Res. Rep. 29.

Xue, X., Wang, Y., 2016. Using Memetic Algorithm for instance coreference resolution. 2016 IEEE 32nd Int. Conf. Data Eng. ICDE 2016 28, 1450–1451. doi:10.1109/ICDE.2016.7498369

Yang, C.C., Chen, H., Hong, K., 2003. Visualization of large category map for Internet browsing. Decis. Support Syst. 35, 89–102. doi:10.1016/S0167-9236(02)00101-X

Yang, Y., Sun, Y., Tang, J., Ma, B., Li, J., 2015. Entity Matching across Heterogeneous Sources. Proc. 21th ACM SIGKDD Int. Conf. Knowl. Discov. Data Min. 1395–1404. doi:10.1145/2783258.2783353

Yu, M., 2014. Entity Linking on Graph Data. Www 21–25. doi:10.1145/2567948.2567954

Yumusak, S., Dogdu, E., Kodaz, H., 2014. A Short Survey of Linked Data Ranking 14–17. doi:10.1145/2638404.2638523

Zhu, L., Ghasemi-gol, M., Szekely, P., Galstyan, A., Knoblock, C.A., 2016. Unsupervised Entity Resolution on Multi-type Graphs, in: International Semantic Web Conference. pp. 649–667. doi:10.1007/978-3-319-46523-4

ANNEX A. INTRODUCTION TO DSL-TOOLS 2015

ANNEX A. INTRODUCTION TO DSL-TOOLS 2015

Annex A aims to introduce to the user a brief description of DSL-Tools in order to help a user to start the process of creation new project for developing a domain-specific language.

Domain-Specific Language Tools (DSL Tools), which are hosted in Visual Studio, let a user design a domain-specific language and then generate everything that users must have to create models that are based on the language. Some of the tools included in DSL-Tools are:

- A project wizard that uses different solution templates to help you start developing your domain-specific language.
- A graphical designer for creating and editing your domain-specific language definition.
- A validation engine that makes sure that the domain-specific language definition is well-formed, and displays errors and warnings if there are problems.
- A code generator that takes a domain-specific language definition as input and produces source code as output.

The Domain-Specific Designer Wizard provides the following solution templates:

- Task Flow
- Class Diagrams
- Minimal Language
- Component Models
- Minimal WPF
- Minimal Windows.
- Forms
- DSL Library

This annex is divided in three sections: section 1 describes the required software to be installed for developing a new project and it also shows how to install them. Section two defines how to create a new project and finally, section 3 describes the IDE and what it offers to the user in order to create a new domain-specific language.

1. INSTALLATION MANUAL

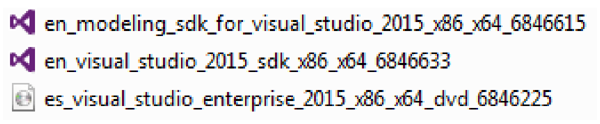
This section describes the process of installation of Visual Studio Enterprise 2015. In this sense, it will be divided in: software requirements and installation process.

1.1. SOFTWARE REQUIREMENTS

For starting with the installation of the components for the development of a DSL project, it is necessary to have downloaded the following software previously:

- Visual Studio Enterprise 2015: it must be the Enterprise version because it is the unique version that allows the DSLs design.
- Visual Studio 2015 SDK: this is the developer kit needed for developing software to be added to the Visual Studio tool.
- Modeling SDK: this is the modeling tools development kit that contains all the necessary tools for creating a DSL. once

Figure A-1 shows the elements downloaded by the doctoral student. Once all the software requirements have been collected, the installation process of Visual Studio Enterprise 2015 can start.

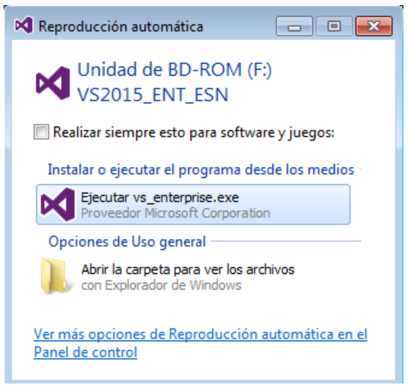


en_modeling_sdk_for_visual_studio_2015_x86_x64_6846615
en_visual_studio_2015_sdk_x86_x64_6846633
es_visual_studio_enterprise_2015_x86_x64_dvd_6846225

Figure A-1. Needed Software for Installation of Visual Studio Enterprise 2015

1.2.INSTALLATION PROCESS

The first step that the user must perform is the execution of the Visual Studio Enterprise 2015 image. This execution, produces the dialog window showed in Figure A-2. In this case the user must click on the "Execute vs_enterprise.exe" option.



Reproducción automática

Unidad de BD-ROM (F:)
VS2015_ENT_ESN

Realizar siempre esto para software y juegos:

Instalar o ejecutar el programa desde los medios

Ejecutar vs_enterprise.exe
Proveedor Microsoft Corporation

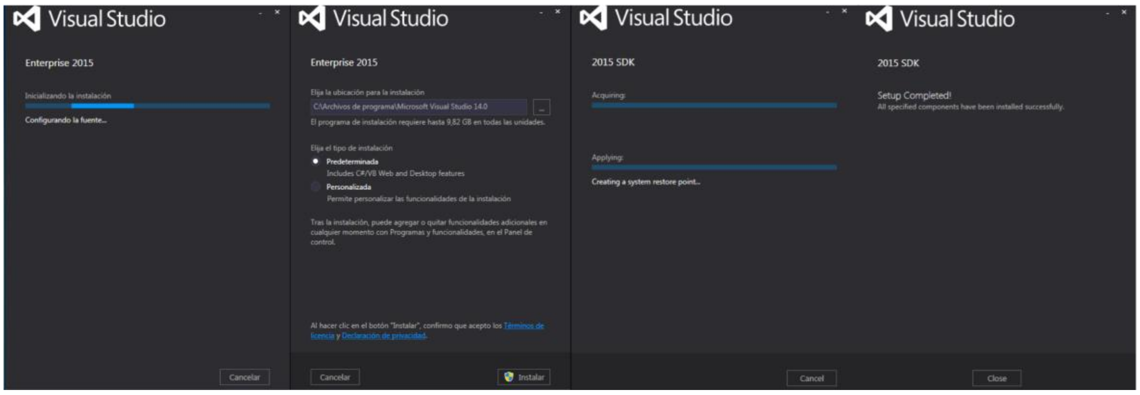
Opciones de Uso general

Abrir la carpeta para ver los archivos
con Explorador de Windows

Ver más opciones de Reproducción automática en el Panel de control

Figure A-2. Installation dialog of VS2015 Enterprise

When the first step of the installation assistant finishes of collecting all the data to proceed with the installation, it will show a new window to the in which he must choose the location of the installation, giving the possibility of choosing one by default or customize it. Once selected and the process finishes, the user will be asked for restarting the system. Once restarted, the user must run the Visual Studio 2015 SDK installer. All these steps, are shown in Figure A-3


Visual Studio

Enterprise 2015

Inicializando la instalación

Configurando la fuente...

Visual Studio

Enterprise 2015

Elija la ubicación para la instalación

C:\Archivos de programa\Microsoft Visual Studio 14.0

El programa de instalación requiere hasta 9,82 GB en todas las unidades.

Elija el tipo de instalación

● Predeterminada
Includes C#/VB Web and Desktop features

Personalizada
Permite personalizar las funcionalidades de la instalación

Tras la instalación, puede agregar o quitar funcionalidades adicionales en cualquier momento con Programas y funcionalidades, en el Panel de control.

Al hacer clic en el botón "Instalar", confirmo que acepto los Términos de licencia y Declaración de privacidad.

Visual Studio

2015 SDK

Acquiring:

Applying:

Creating a system restore point...

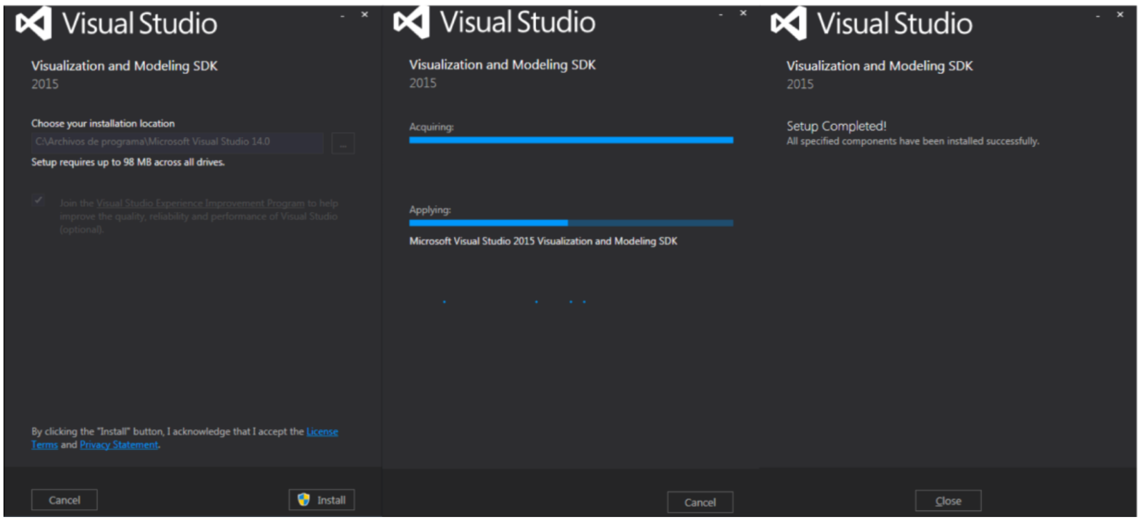
Visual Studio

2015 SDK

Setup Completed!
All specified components have been installed successfully.

Figure A-3. Installation process of VS2015 Enterprise

The second step that the user must perform is the execution of the Visual Studio Enterprise 2015 SDK installation and after that and to conclude, the execution of the Modeling SDK 2015 installation. For the installation of both tools, the user only has to follow the assistant (Figure A-4) by clicking in the "next" button. One this process has finished, the only action that the user has to do is click on the "Close" button for completing the development environment installation.


Visual Studio

Visualization and Modeling SDK
2015

Choose your installation location

C:\Archivos de programa\Microsoft Visual Studio 14.0

Setup requires up to 98 MB across all drives.

Join the Visual Studio Experience Improvement Program to help improve the quality, reliability and performance of Visual Studio (optional).

By clicking the "Install" button, I acknowledge that I accept the License Terms and Privacy Statement.

Visual Studio

Visualization and Modeling SDK
2015

Acquiring:

Applying:

Microsoft Visual Studio 2015 Visualization and Modeling SDK

Visual Studio

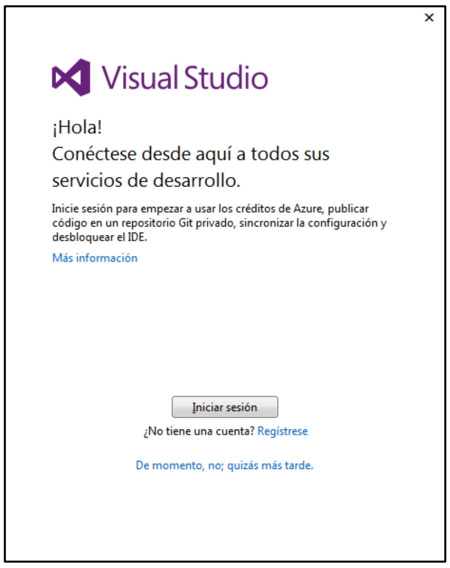
Visualization and Modeling SDK
2015

Setup Completed!
All specified components have been installed successfully.

Figure A-4. Installation dialog of Enterprise SDK 2015 and Modeling SDK 2015

2. Minimal Language Start Project

For creating a new project that allow the user develop a Domain Specific Language, once installed all the required software, the first step that the user must perform is open the Visual Studio 2015 tool. The first time that the user opens the application, it will show a dialog (Figure A-5) where the user can login with their Microsoft credentials or just use it without login.



Visual Studio

¡Hola!
Conéctese desde aquí a todos sus servicios de desarrollo.

Inicie sesión para empezar a usar los créditos de Azure, publicar código en un repositorio Git privado, sincronizar la configuración y desbloquear el IDE.

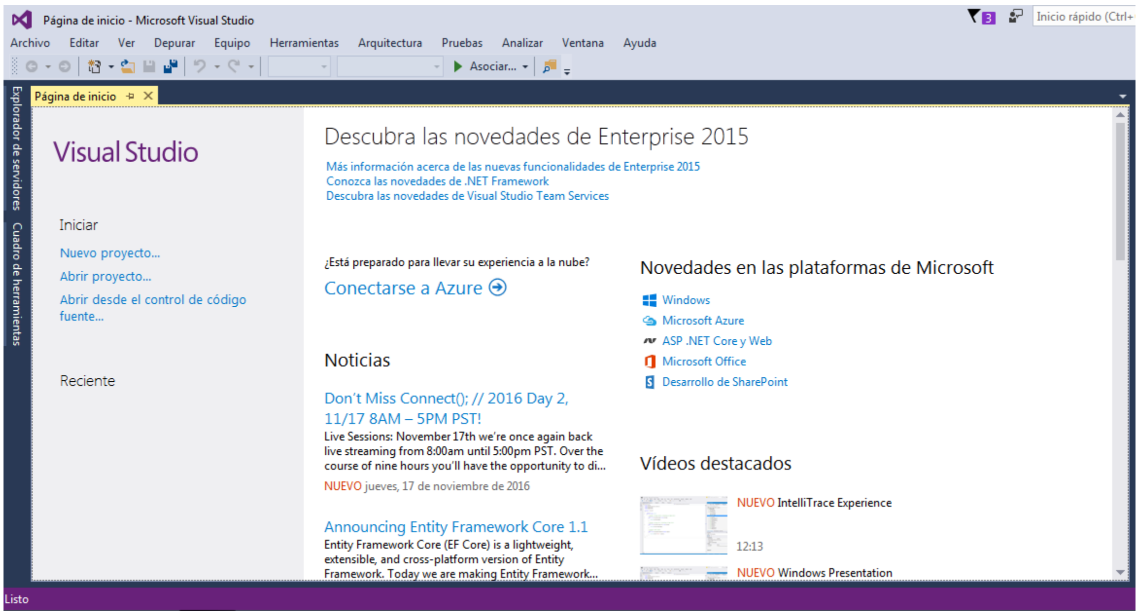
Más información

¿No tiene una cuenta? Regístrese

De momento, no; quizás más tarde.

Figure A-5. Installation dialog of Enterprise SDK 2015 and Modeling SDK 2015

In the next step, the application will show to the user a dialog (Figure A-6) where there is information about all the tools that compose Visual Studio 2015. Also, it is possible to see that it appears a menu entitles "Start". Under this menu, there are two options "New Project" and "Open Project from Code". The user must choose the first one. After that, it will appear all the project just under the "Recent" title.



Visual Studio

Iniciar
Nuevo proyecto...
Abrir proyecto...
Abrir desde el control de código fuente...

Reciente

Descubra las novedades de Enterprise 2015
Más información acerca de las nuevas funcionalidades de Enterprise 2015
Conozca las novedades de .NET Framework
Descubra las novedades de Visual Studio Team Services

¿Está preparado para llevar su experiencia a la nube?
Conectarse a Azure

Noticias

Don't Miss Connect(); // 2016 Day 2, 11/17 8AM – 5PM PST!
Live Sessions: November 17th we're once again back live streaming from 8:00am until 5:00pm PST. Over the course of nine hours you'll have the opportunity to di...
NUEVO jueves, 17 de noviembre de 2016

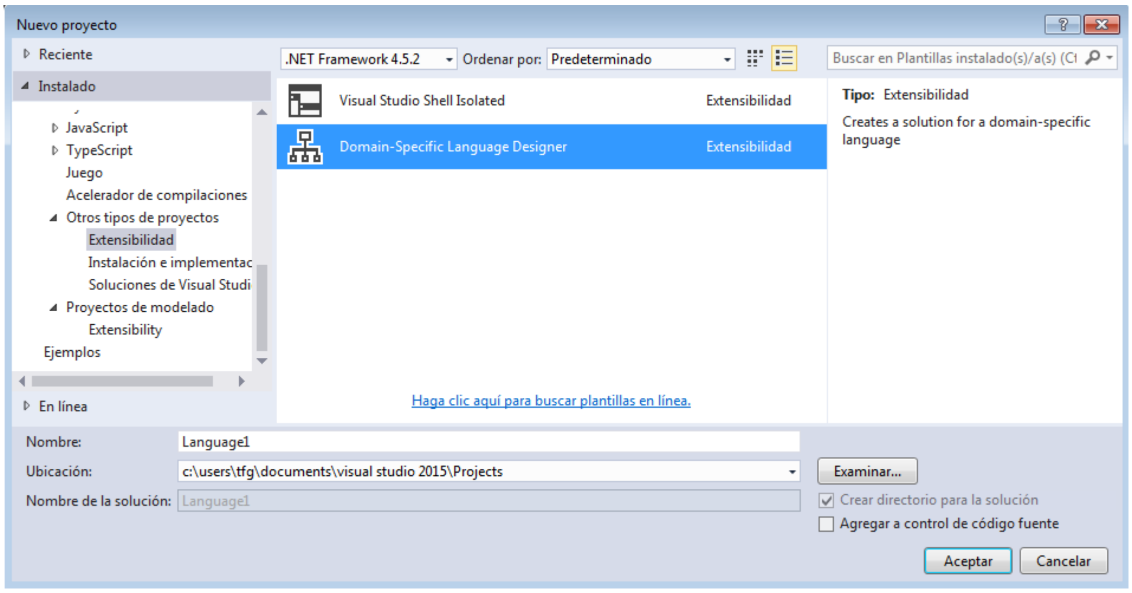
Announcing Entity Framework Core 1.1
Entity Framework Core (EF Core) is a lightweight, extensible, and cross-platform version of Entity Framework. Today we are making Entity Framework...

Novedades en las plataformas de Microsoft
Windows
Microsoft Azure
ASP .NET Core y Web
Microsoft Office
Desarrollo de SharePoint

Vídeos destacados
NUEVO IntelliTrace Experience
12:13
NUEVO Windows Presentation

Figure A-6. New Project

Once selected the "New Project" option, it will appear all the type of projects that can be created with Visual Studio 2015. Between them, there is in the "Other Type of Projects" option the "Extensibility" projects. The user must select this one and after that select the "Domain-Specific Language Designer" and click the button "Accept" (Figure A-7).


Nuevo proyecto

Reciente

Instalado

JavaScript

TypeScript

Juego

Acelerador de compilaciones

Otros tipos de proyectos

Extensibilidad

Instalación e implementac

Soluciones de Visual Studi

Proyectos de modelado

Extensibility

Ejemplos

En línea

.NET Framework 4.5.2 Ordenar por: Predeterminado

Buscar en Plantillas instalado(s)/a(s) (Ct

Visual Studio Shell Isolated	Extensibilidad
Domain-Specific Language Designer	Extensibilidad

Tipo: Extensibilidad

Creates a solution for a domain-specific language

Haga clic aquí para buscar plantillas en línea.

Nombre: Language1

Ubicación: c:\users\tfg\documents\visual studio 2015\Projects

Nombre de la solución: Language1

Examinar...

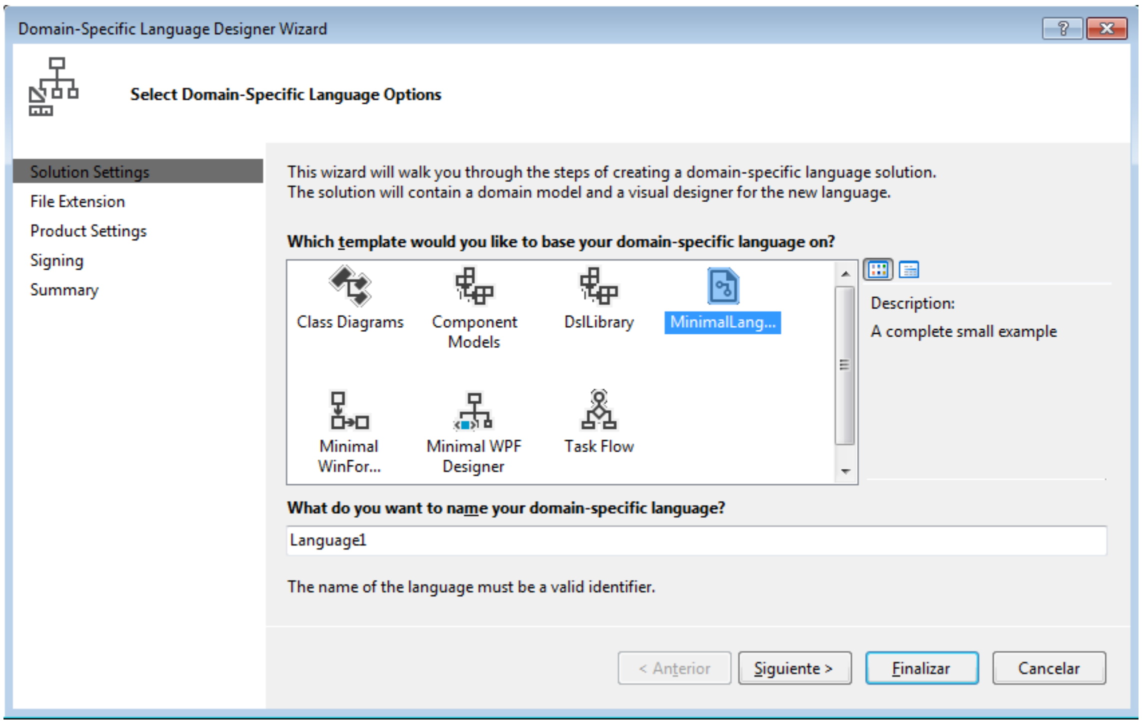
☑ Crear directorio para la solución

☐ Agregar a control de código fuente

Aceptar Cancelar

Figure A-7. Domain Specific Language Designer

Next, the assistance of new project creation will show a set of templates (Figure A-8). The user must select the “Minimal Language” template and click on the “Finish” button.


Domain-Specific Language Designer Wizard

Select Domain-Specific Language Options

Solution Settings

File Extension

Product Settings

Signing

Summary

This wizard will walk you through the steps of creating a domain-specific language solution.
The solution will contain a domain model and a visual designer for the new language.

Which template would you like to base your domain-specific language on?

Class Diagrams, Component Models, DslLibrary, MinimalLang..., Minimal WinFor..., Minimal WPF Designer, Task Flow

Description:
A complete small example

What do you want to name your domain-specific language?

Language1

The name of the language must be a valid identifier.

< Anterior Siguiente > Finalizar Cancelar

Figure A-8. Minimal Language

Once all these steps have been performed, the user will have created an example solution (Figure A-9) on which the user can see the capabilities of the language designer. For starting an empty project, the user only must delete the content of the “DSLDefinition.dsl” file.

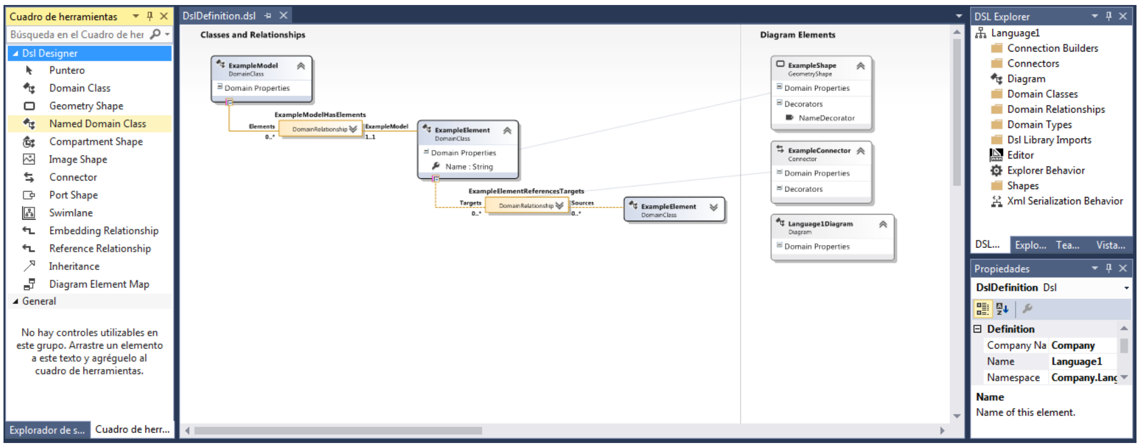


Figure A-9. Minimal Language Template

3. Components of the Visual Studio 2015

The user interface for DSL graphic design is divided into several parts (Figure A-10) that have a specific purpose.

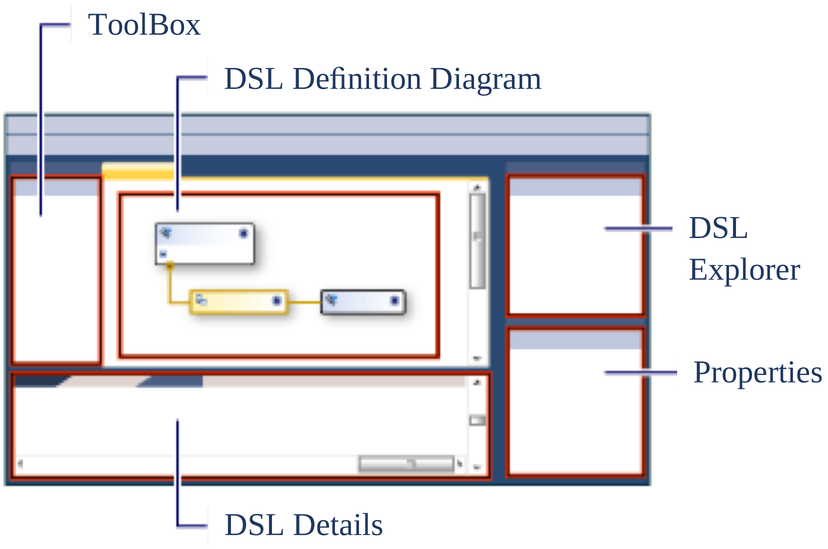


Figure A-10. Minimal Language Template

3.1. Tool Box

The right combination of the elements that compose the tool box (Figure A-11) will give the user the possibility of creating a DSL. Below each of the elements and their functionality are described. In the toolbox are mixed both the elements that will conform the abstract syntax as well as the concrete syntax.

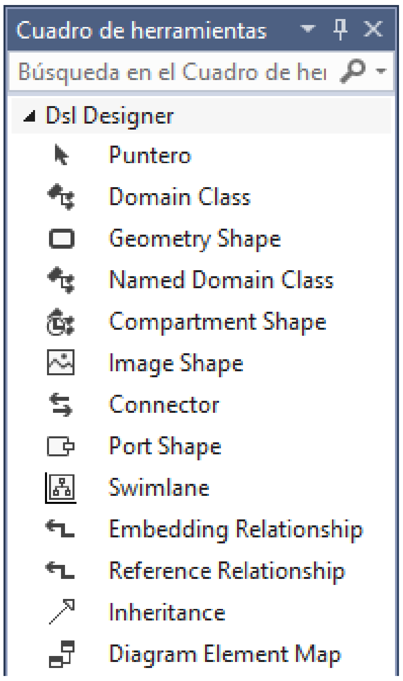


Figure A-11. ToolBox

- **Pointer**: this element let select one or more elements inside the DSL definition side. Also, it allows to drag and drop elements from the toolbox to the DSL Definition side.
- **Domain Class**: this element represents the domain classes. These classes may have properties.
- **Geometry Shape**: this element represents the form and appearance that will have the classes where this element is associated to. It may have different geometric forms and even contain icons placed in different places of the figure.
- **Named Domain Class**: this element represents the domain classes which will have a Name property by default, being this one the unique identifier of the element.
- **Compartment Shape**: this element represents the form and appearance that may contain one or more lists of elements. This element is very useful if the user wants to represent a class with a concrete behavior for the properties of the class and another for the methods that contains. It has expressive limitations because it does not allow to represent direct relationships with the elements of the boxes. The relationship type between the elements of the boxes must be embedded because one class contain the other one.
- **Shape**: this element is a form which when it is associated to a class, it shows the appearance of this class with the same aspect of the reference image. The advantage of this element is that it may be of any type of form.
- **Connector**: this element shows the appearance that will have the relationships of the figures. It may be tagged and its form and color can be changed because it is just another figure.
- **Port Shape**: this element represents a figure that allows to associate ports. This ports are represented as points where two figures may be connected through a connector element using the embedded relationship.

- **Swimlane**: this element represents a connector that allows to host several kinds of domain classes in it, may have one or more lanes in which to contain other figures. It uses the embedded relationship between the container class and the contained class.
- **Embedding Relationship**: this element represents the relationship between two components that will one to be part of the other one.
- **Reference Relationship**: this element represents the relationship between two components that show a reference from one to the other one.
- **Inheritance**: this element represents the inheritance relationship between father and son that will make the child have all the properties of the father plus his own ones. The inheritance can be applied to relationships, forms and connectors but should be kept within the same group, it means, father and son must be the same type of element.
- **Diagram Element Map**: this element represents the relationship between the components that are part of the classes and relationships and the elements of the diagram.

3.2.DSL Definition Diagram

This is the window that shows the content of any element that may be viewable from those found in the DSL browser. The main structure of this window can be observed in Figure A-12

A big variety of elements may be represented, but perhaps, the most important one is the DslDefinition.dsl. This element is where the user can add the domain classes, domain relationships and, the diagram elements. This window will open automatically when you click on DslDefinition.dsl located in the DSL browser.

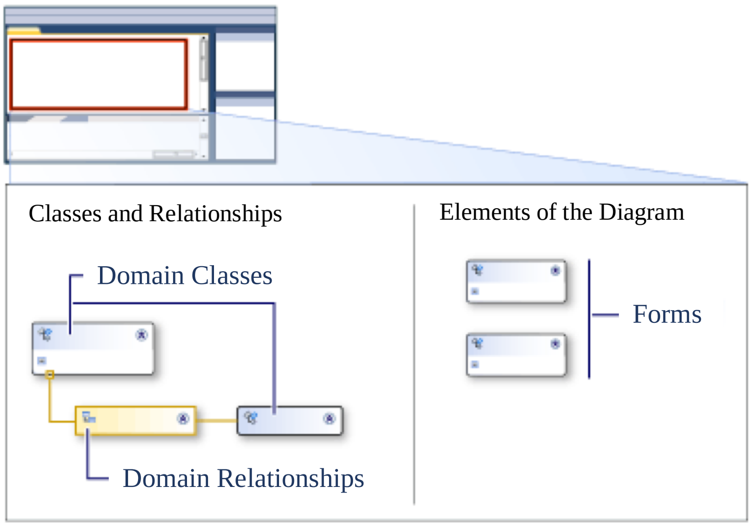


Figure A-12. DslDefinition.dsl Element

As Figure A-13 shows, the structure is formed hierarchically and it is not possible that the elements shown in the diagram occupy any place. The positions are determined by the relationships that each element has and the what that the user can do is moving up or down the elements to improve the readability of the diagram.

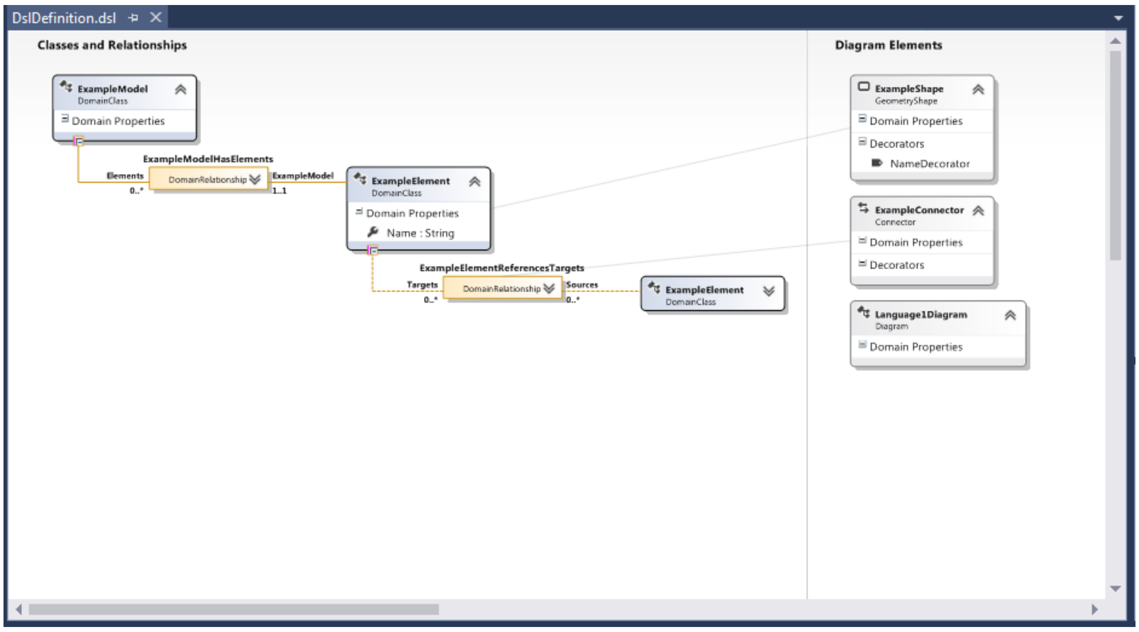
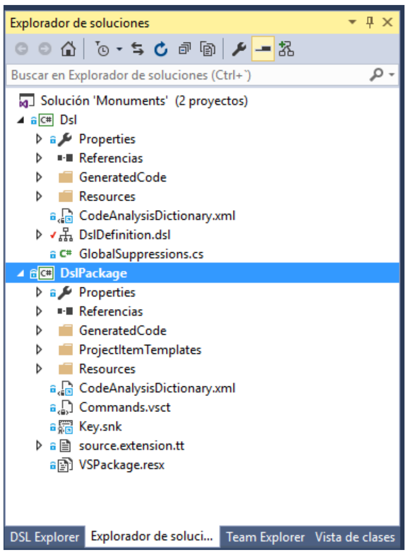


Figure A-13. DSL Definition Diagram

3.3.DSL SOLUTIONS EXPLORER

The more important elements of the DSL Solutions Explorer (Figure A-14) are:

- Dsl\Definition.dsl: this is the file on which the user drags elements of the toolbox and it is the one that will have all the information to be able to create the definition of the DSL.
- DSL Project: it contains the code that defines the DSL.
- DslPackage Project: it contains the code for open and edit the instances of the DSL by Visual Studio.


Explorador de soluciones

Buscar en Explorador de soluciones (Ctrl+`)

```
Solución 'Monuments' (2 proyectos)
  Dsl
    Properties
    Referencias
    GeneratedCode
    Resources
    CodeAnalysisDictionary.xml
    DslDefinition.dsl
    GlobalSuppressions.cs
  DslPackage
    Properties
    Referencias
    GeneratedCode
    ProjectItemTemplates
    Resources
    CodeAnalysisDictionary.xml
    Commands.vsct
    Key.snk
    source.extension.tt
    VSPackage.resx
```

DSL Explorer | Explorador de soluci... | Team Explorer | Vista de clases

Figure A-14. DSL Solutions Explorer

Inside this explorer, there are all the elements added to the DslDefinition.dsl grouped by different types of elements. If the DslDefinition is overloaded of elements and it is difficult to distinguish them clearly, the easily way to select them is selecting any element inside of the different folders, the properties of the element are displayed in the bottom side, inside of the properties tab (Figure A-15).

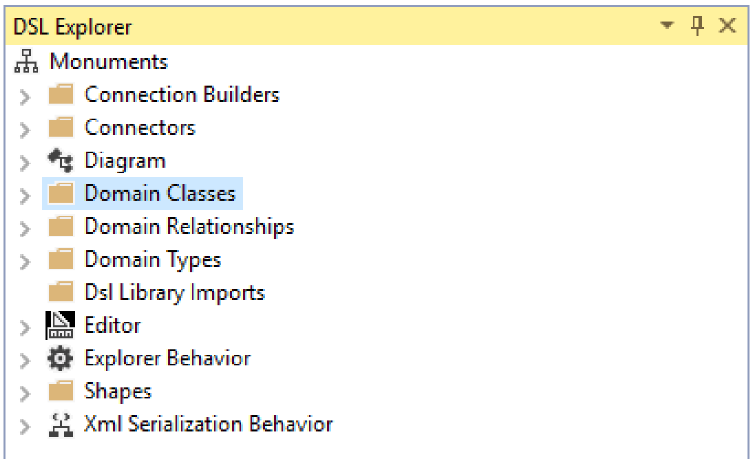

DSL Explorer
Monuments
- Connection Builders
- Connectors
- Diagram
- Domain Classes
- Domain Relationships
- Domain Types
- Dsl Library Imports
- Editor
- Explorer Behavior
- Shapes
- Xml Serialization Behavior

Figure A-15. DSL Elements Explorer

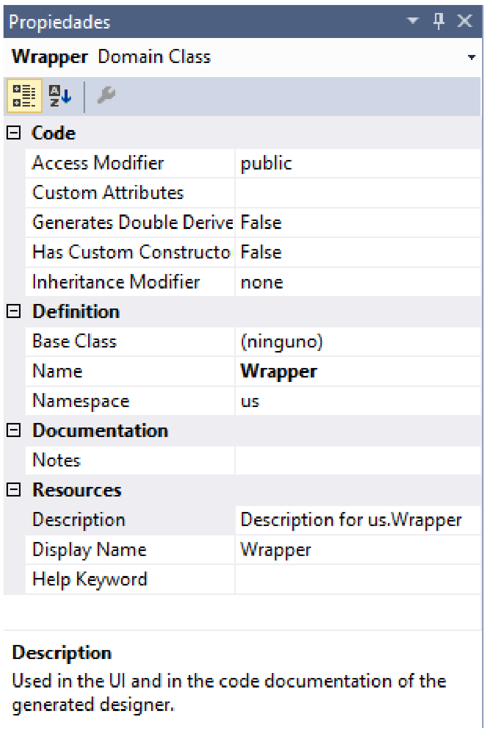
The more important types of elements for the creation of a DSL are:

- **Connectors**: it contains all connector type elements of the diagram, these elements define the appearance, definition and layout of the relationships among others.
- **Diagram**: it contains the mappings between the classes with their forms and the relationships with the connectors as well as their properties.
- **Domain Classes**: it contains all the domain classes.
- **Domain Relationships**: it contains all the domain relationships.
- **Domain Types**: it contains all the default data types and the ones expressly created for a specific DSL in cases where they have been defined.
- **Editor**: it contains the definition of the tools that will be the result of the execution of the DSL. This element will provide to the user a Tool Box in the executed instance.
- **Shapes**: it contains all the form type elements diagram, these elements define the appearance, definition and layout of the classes among others.

3.4.PROPERTIES

This window shows the properties of each element of the definition diagram of the DSL and the DSL Solution Explorer since one is the projection of another.

Figure A-16 how it looks like. It is possible to find a lot of options for the customization of each element of the DSL, allowing to provide a wide range of information types, representations and behaviors.



Code	
Access Modifier	public
Custom Attributes	
Generates Double Derive	False
Has Custom Constructo	False
Inheritance Modifier	none
Definition	
Base Class	(ninguno)
Name	**Wrapper**
Namespace	us
Documentation	
Notes	
Resources	
Description	Description for us.Wrapper
Display Name	Wrapper
Help Keyword	

Description
Used in the UI and in the code documentation of the generated designer.

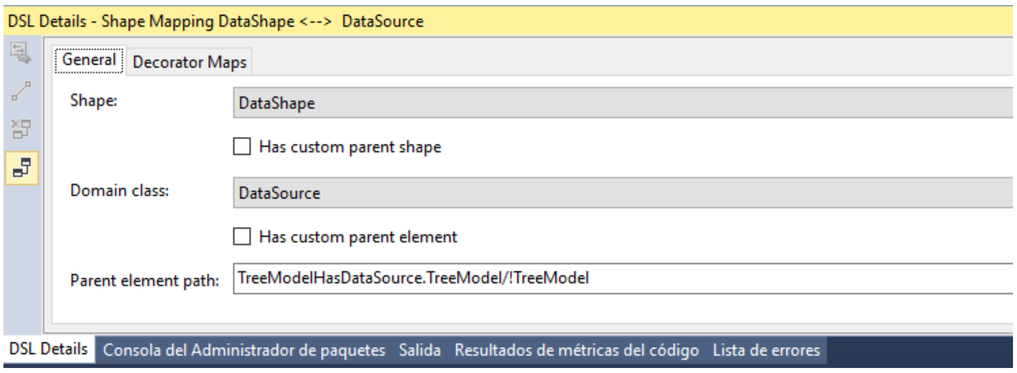
Figure A-16. Properties Window

3.5.DSL Details

This window is divided in two different tabs (DSL Details and List of Errors) that will help the user to set the configuration of the forms mappings and, they will show information about errors of the DSL if it exists.

DSL Details

The configuration that a user can perform in this window (Figure A-17) will let him manage all what concerns the mapping that exist between the elements of the diagram and the classes or relations, being able to customize the behavior of these.



DSL Details - Shape Mapping DataShape <--> DataSource

Shape:	DataShape
	☐ Has custom parent shape
Domain class:	DataSource
	☐ Has custom parent element
Parent element path:	TreeModelHasDataSource.TreeModel/!TreeModel

General
Decorator Maps
DSL Details
Consola del Administrador de paquetes
Salida
Resultados de métricas del código
Lista de errores

Figure A-17. DSL Details Tab

List of Errors

This is a list where the user can see different types of errors categorized in their importance level (Figure A-18). There are three types of errors: messages, warnings and errors. For each entry of the list of this panel, the user will have information that indicates its origin and details about why this information is displayed in this panel.

- **Errors**: show important fails of the DSL, whit this type of errors present in the DSL, it will not be executed.
- **Warnings**: are fails with a low level of importance. It warns the user that the use of an element is not being totally correct, but it allows to execute the DSL.
- **Messages**: are information that may be shown in the execution of the DSL, they are merely informative.

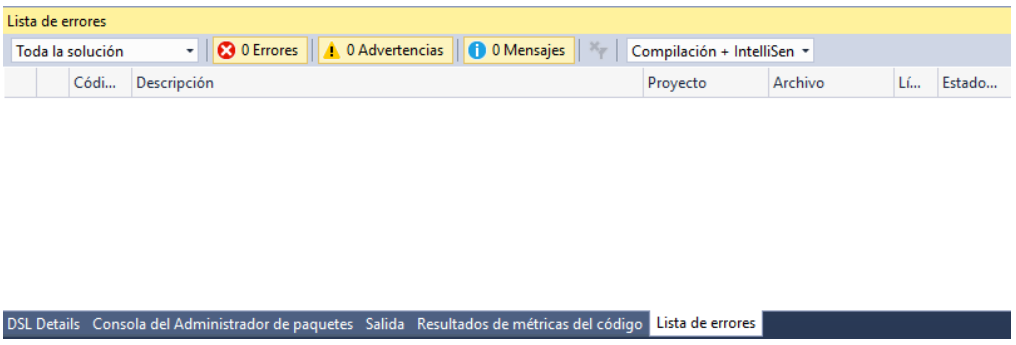


Figure A-18. DSL Details Tab

3.6. Compilation Menu

Once it has been explained the different windows that Visual Studio 2015 offers for the development of a DSL, now, it will be explained the options that the Compilation Menu offers. This menu is in the main horizontal bar of the IDE (Figure A-19).

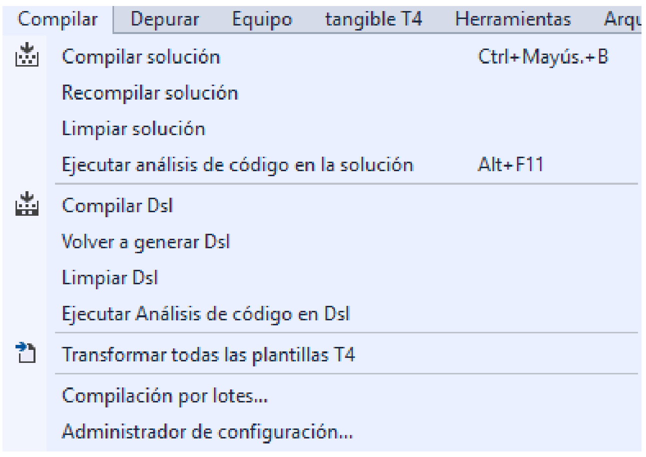

Compilar
Depurar
Equipo
tangible T4
Herramientas
Arqu
Compilar solución Ctrl+Mayús.+B
Recompilar solución
Limpiar solución
Ejecutar análisis de código en la solución Alt+F11
Compilar Dsl
Volver a generar Dsl
Limpiar Dsl
Ejecutar Análisis de código en Dsl
Transformar todas las plantillas T4
Compilación por lotes...
Administrador de configuración...

Figure A-19. Compilation Meny

Each time that the user makes a modification in the definition of the DSL, it will be necessary to execute the “Transform all the templates T4” option. Also, it is possible to compile the DSL o the complete solution. This is interesting if the user wants to know if any new change that provokes compilation mistakes have been performed. This step will be implicitly performed when the solution is executed and the instance of the defined DSL is generated.

3.7.DEBUG MENU

This menu (Figure A-20) contains the execution and debugging, along with all the tools that can be used to debug the project solution

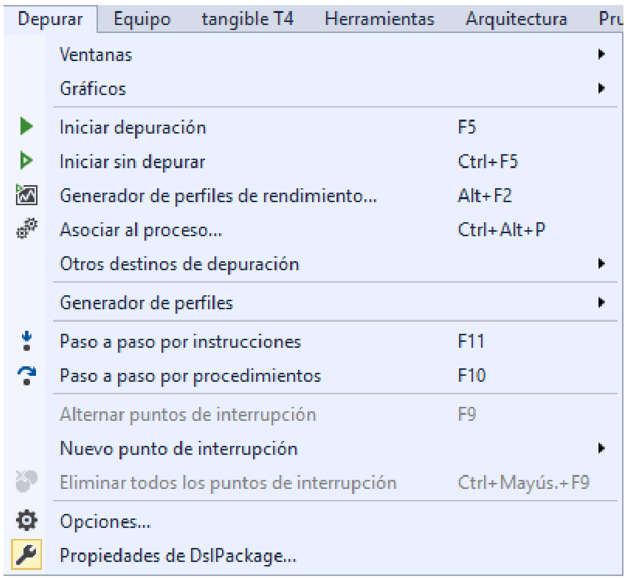

Depurar
Equipo
tangible T4
Herramientas
Arquitectura
Pru
Ventanas
Gráficos
Iniciar depuración F5
Iniciar sin depurar Ctrl+F5
Generador de perfiles de rendimiento... Alt+F2
Asociar al proceso... Ctrl+Alt+P
Otros destinos de depuración
Generador de perfiles
Paso a paso por instrucciones F11
Paso a paso por procedimientos F10
Alternar puntos de interrupción F9
Nuevo punto de interrupción
Eliminar todos los puntos de interrupción Ctrl+Mayús.+F9
Opciones...
Propiedades de DslPackage...

Figure A-20. Debug Menu

The option “Start debugging” will open an instance of the solution of the project along with a set of indicators that helps the debugging process such as the use of resources (memory, processor). Also, it allows to capture exceptions in the case of the execution of the solution develops an unexpected behavior.

The option “Start without debugging” will open an instance of the solution of the project as it would be executed in production environment.

ANNEX B. USER MANUAL

ANNEX B. USER MANUAL

This annex describes the use of the support tool presented in Chapter VII, where it was exposed the current implementation based on a defined concrete syntax for the definition of the models. This support tool was developed under the support of the Microsoft DSL Tools IDE.

1. USER MANUAL

The user of the support tool must have installed in his environment Visual Studio Enterprise Edition. Once the project has been opened, the next step that the user must do is run it. Performed this step, Visual Studio will generate an instance of the developed project obtaining a new environment with all the elements that will allow the user define this entity reconciliation problem.

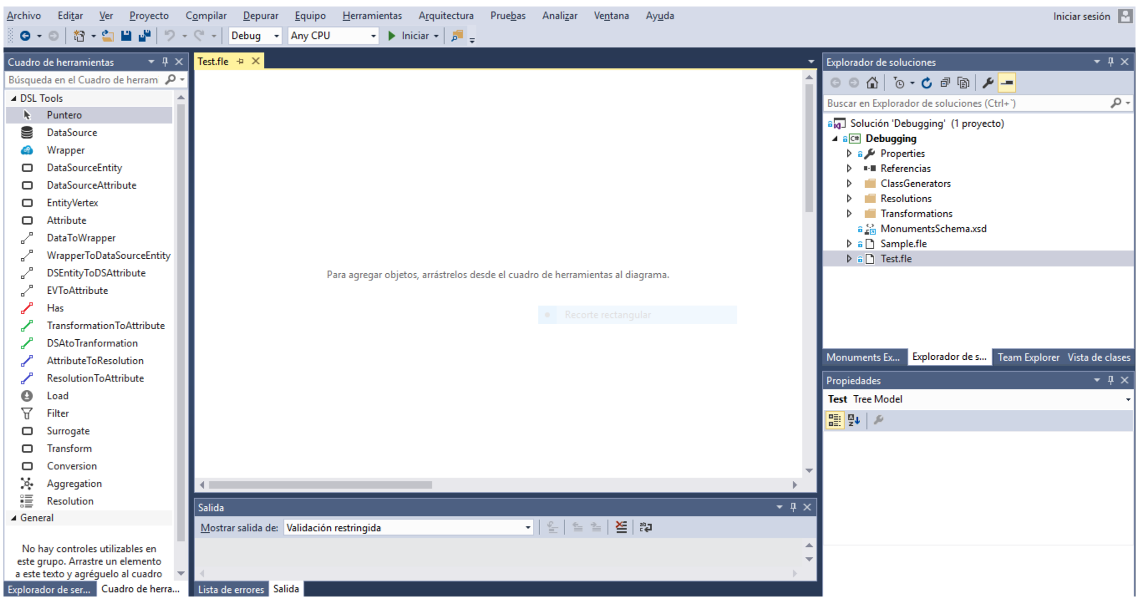


Figure B-1. Instance of Support Tool

As it is possible to see in Figure AnnexB-1, there are five clearly differentiated areas: toolbox, solutions explorer, properties, diagram window and list of errors and output.

- **Toolbox**: this is the window where all the elements that are possible to add in a diagram are displayed. For adding any into the diagram, the user will have to drag and drop the element into any file with ".fle" extension.
- **Solutions Explorer**: this window contains the structure of the project. In it, it is possible to find all the properties of the project, the referenced libraries, the T4 templates for code generation and the files that will contain the diagram.
- **Properties**: it is a window that shows the properties of any mouse selected element.

- **Diagram Window**: this is the central window where it is possible to drop the elements from the toolbox for designing the diagram.
- **List of Errors and Output**: this window is composed by two tabs. On one hand the list of errors that will show any error, warning or message information when the execution of the project is performed and on the other hand, the output that will show the processes information performed during the transformation tasks of the templates or diagram compilation.

The toolbox shows all the elements that may be used for designing a diagram. There are three types of elements clearly differentiated, the ones that refer to information that the domain must contains, the ones that show the operations on that domain and the ones that allow the relationships between previous elements.

Elements that refer to the information modeling are:

- **DataSource**: this element allows to define a data source specifying its name and its URL properties.
- **Wrapper**: this element does not have attributes and it represents an interface between the data origin and the entity to be modeled trough the DataSourceEntity element.
- **DataSourceEntity**: this element represents the entity on which the data source attributes will be referenced.
- **DataSourceAttribute**: this element represents the attributes composed by name and type properties.
- **EntityVertex**: this element represents a vertex of a graph structure.
- **Attribute**: this element represents the attributes that define the vertexes.

Elements that refer to the information operations are:

- **Load**: this element models the transformation pattern that is applied on an instance of the attribute of the data source entity element and it represents that its value will be loaded in an attribute of the entity vertex element.
- **Filter**: this element models the transformation pattern that is applied on some instances of the attribute of the same data source entity element and it relate them with an attribute of the entity vertex element.
- **Aggregation**: this element models the resolution pattern between some attributes of different entity vertex elements that will be concatenated in the attribute of another entity vertex element.
- **Resolution**: this element models the resolution pattern between attributes of different entity vertex elements among which one of them will be the one that is loaded in the attribute of another entity vertex element.

Elements that refer to the relationships between classes are:

- **DataToWrapper**: this is the element that allows to relate a data source element and a wrapper element.
- **WrapperToDataSourceEntity**: this is the element that allows to relate a wrapper element and a data source element.

- **DSEntityToAttribute**: this is the element that allows to relate a data source entity element and a data source attribute element.
- **EVToAtribute**: this is the element that allows to relate an entity vertex element and an attribute element.
- **Has**: this is the element that allows to relate an entity vertex element and another entity vertex element.

Elements that refer to the transformation relationships between attributes and classes are:

- **TransformationToAttribute**: this is the element that allows to relate transformations and attributes elements.
- **DSAtoTransformation**: this is the element that allows to relate data sources and transformations elements.
- **AttributeToResolution**: this is the element that allows to relate resolution transformations and attributes elements.
- **ResolutionToAttribute**: this is the element that allows to relate attributes and resolution transformations elements.

Once all the elements that compose the DSL are describe, it is time to model the diagram. For this purpose, it must be applied the procedure described in the analysis phase of MaRIA methodology described in Chapter IV. Whit the aim of show the possibilities of this support tool, it has been created an example based on the management of monuments information of the region of Andalusia (Figure AnnexB-II).

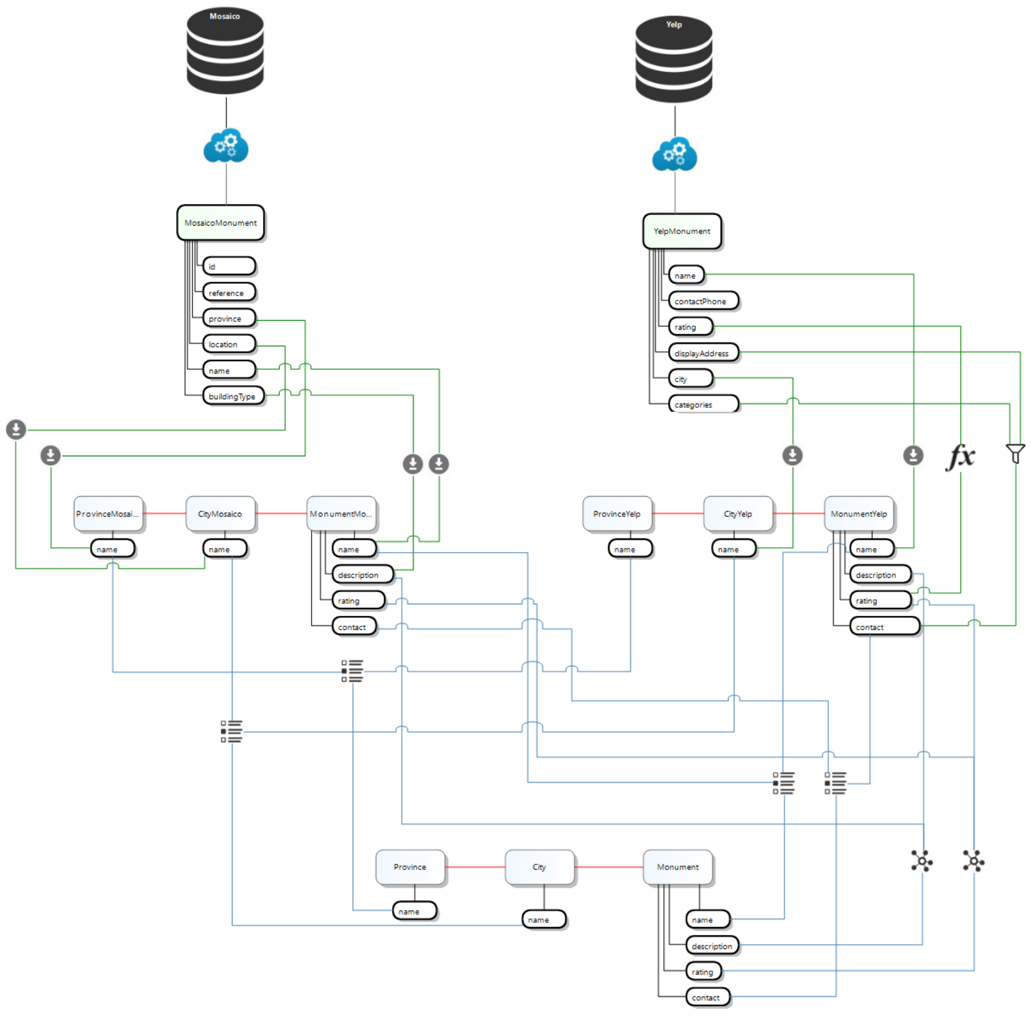


Figure B-2. Example of Model Design with the Support Tool

ANNEX C. GLOSSARY OF TERMS

ANNEX C. GLOSSARY OF TERMS

Annex C shows a glossary of terms, defining the terms and acronyms most commonly used in this Doctoral Thesis.

Abstract Syntax. Term used in this Doctoral Thesis to refer to the term metamodel, as the metamodels allow us to define in an abstract way the syntax used in entity reconciliation problem.

Algorithm Solutions. It represents solutions based on the application of an algorithm.

Automation. Application of automatic procedures in the execution of a process or in an industry.

Big Data Paradigm. Distribution of data systems across horizontally-coupled independent resources to achieve the scalability needed for the efficient processing of extensive datasets.

CASE. Acronym of "Computer Aided Software Engineering" and whose term encompasses those computer tools designed to increase productivity in the development of software reducing the cost of the same in terms of time and money.

Cloud Computing. paradigm for enabling network access to a scalable and elastic pool of shareable physical or virtual resources with self-service provisioning and administration on-demand

Concrete syntax. Term used in this Doctoral Thesis to refer to the term model, since the models allow us to define in concrete form the syntax used in in entity reconciliation problem.

Data Analyst. Actor defined in the ISO/IEC CD 20546 - Big data report in charge of performing logical data modeling » Identifies patterns in data.

Data Consumer. Actor defined in the ISO/IEC CD 20546 - Big data report Data which includes end users or other systems who utilize the results of an application.

Design Phase. Extrapolation of structural phase of UML. The solution is focused in the design of the problem and not in how the solution will be executed.

Early Testing. Testing that it is applied in the early stages of a software development.

Entity Reconciliation. The process of identifying and merging records judged to represent the same real-world entity.

Framework. Structure expressed in diagrams, text, and formal rules which relates the components of a conceptual entity to each other.

Heterogeneous Data Sources. Set of data sources that do not share the same structure or the same data even if they store information related to the same subject.

ISO/IEC CD 20546. Big data report created by ISO/IEC JTC 1 - Information technology.

IWT2. Acronym of «Ingeniería Web y Testing Temprano». Research group of the University of Seville, referenced in the "Plan Andaluz de Investigación" as PAIDI TIC021.

Mapping. Term used in this Doctoral Thesis to refer to the language Operational Mapping of QVT (also called QVTo), which implements transformations or parts of them.

MaRIA Tool. It represents the CASE tool that supports the framework defined in this Doctoral Thesis. This tool materializes and automates the metamodels, constraints and transformations that are part of the MaRIA methodology.

MaRIA. Acronym of «Model-driven entity ReconcilIAtion». It represents the methodology defined in this Doctoral Thesis, which allows the definition and analysis of entity reconciliation problems.

MDE. Acronym of «Model Driven Engineering». Paradigm that focuses on the creation and exploitation of domain models, allowing Software Engineers to become independent of the representation and to focus on the concepts.

Metamodel. Language that describes the concepts used in a concrete domain model.

Model. Term used in this Doctoral Thesis to refer to the representation through a concrete language of a concept used as part of an information system.

MOF. Acronym of «Meta Object Facility». OMG Standard which provides the basis for defining and manipulating metamodels, including the capabilities needed to manage their corresponding models.

MOFM2T. Acronym of «MOF Model to Text Transformation Language» which identifies a standard proposed by OMG to define derivation rules to generate a textual version of models.

Multi-Relational. Problem that entities present when they need to be related with some others entities.

Multi-Domain. Problem that data presents when it is necessary to be used in different domains.

Multi-Application. Problem that data presents when different applications with different requirements need to be served by this data in different formats.

Non-Heterogeneous Data Sources. Set of data sources that share the same structure or the same data.

Non-Validated – Theoretical Approach. It refers to a solution that proposes a theoretical approach but it has not been validated in any dataset.

OMG. Acronym of «Object Management Group» which identifies a non-profit consortium made up of various companies and organizations. It is dedicated to the care and establishment of various standards of object oriented technologies, as well as to encourage the use of object oriented technology through guides and specifications for them.

Operation Phase. Extrapolation of behavior phase of UML. The solution is focused in the operation performed to carry out the entity reconciliation process.

Real Dataset. Dataset that provides from a real environment, company or organization and it is used to test an entity reconciliation solution.

SLR. Acronym of «Systematic Literature Review». Literature review focused on a research question whose objective is to identify, evaluate, select and synthesize all relevant high quality evidence related to that research question.

SMS. Acronym of «Systematic Mapping Study». Type of SLR.

Software Engineer. A software engineer is a person who applies the principles of software engineering to the design, development, maintenance, testing, and evaluation of the software and systems that make computers or anything containing software work.

Structural Solutions. It represents solutions based on the data structures.

Synthetic Dataset. Dataset created by a user to test a concrete solution to a specific entity reconciliation problem.

Transformation. Term used in this Doctoral Thesis to refer to the specification of how to construct business rules from a model.

UML. Acronym of « Unified Modeling Language». A widely used graphical system modeling language, with which a software engineer can visualize, specify, construct and document a system.

Validated – Theoretical Approach. It refers to a solution that proposes a theoretical approach and it has been validated in any dataset.

Validated – Approach in Industry. It refers to a solution that proposes a theoretical approach and it has been validated in a real context solving an industry problem.

Virtual Graphs. Technology developed by the Computer Languages and Systems department of the University of Seville. It is a graph-based technology where the structures of the graphs are created in execution time.

ANNEX D. RECOGNITIONS AND RESEARCH ACTIVITY

Annex D. Recognitions and Research Activity

In this annex, the research activity of the doctoral student is collected, cataloging it in different sections: participation in scientific outreach events, publications, research projects, transfer projects and research networks in which the doctoral student has participated.

In addition, information about different acknowledges obtained during the elaboration of this work are collected.

Publications have been cataloged according the following typology: book chapters, journal articles and articles published in international congresses and conferences.

1. Acknowledgments

During the development of this doctoral thesis, the doctoral student was selected as beneficiary of a pre-doctoral scholarship granted by the Fujitsu Laboratories of Europe (FLE), being in force from the year of the beginning of his studies until its completion. It was conceded for a total amount of 18.000,00 pounds.

Also, it is important to highlight that in 2015, the doctoral student and their supervisors, won the “Fujitsu awards for innovation 2015”. This award was granted for his research activity in the field of research and development of the DIPHDA (Dynamic Integration for Patrimonial Heritage Data in Andalucía) solution.

2. Research Stays

During the development of this doctoral thesis, the doctoral student has performed some national and international pre-doctoral research stays of different periods. Chronologically descending, those are:

Period	**Research Center**	**Location**
30/10/2016 - 20/12/2015 (7 weeks)	University of California - Berkeley	Berkeley (United States)
01/06/2016 - 01/08/2016 (2 months)	Solent University of Southampton	Southampton (United Kingdom)
11/01/2016 - 14/01/2016 (1 week)	Estancia en Wyzsza Szkola Zarzadzania i Bankowosci w Poznaniu	Cracow (Poland)
06/12/2015 - 16/23/2015 (10 days)	Estancia en Instituto Superior Politécnico José Antonio Echeverría (CUJAE)	La Habana (Cuba)
29/10/2015 - 30/11/2015 (1 month)	Escuela Politécnica Superior de Ingeniería de Gijón (Universidad de Oviedo)	Gijón (Spain)
12/01/2015 - 15/01/2015 (1 week)	Fujitsu Laboratories of Europe	London (United Kingdom)

3. Scientific Outreach Events

During the development of this doctoral thesis, the doctoral student has participated in the following (chronologically descending) scientific outreach events:

Period	Event
2016	The doctoral student was part of the organizing committee, the program committee and scientific committee of the PAAMS conference (International Conference on Practical Applications of Agents and Multi-Agent Systems) held in Seville (Spain) during the month of June 2016. Among other tasks, this participation has resulted in scientific dissemination, review of articles as well as management and organization of the event.
	The doctoral student became part of the scientific committee of the congress JISBD2016 (Jornadas de Ingeniería del Software y Bases de Datos) making the review process of different articles.
	The doctoral student became part of the scientific committee of the congress RCIS2016 (IEEE 11th International Conference on Research Challenges in Information Science) making the review process of different articles.
	The doctoral student became part of the scientific committee of the workshop APMDWE2016 (International Workshop on Avanced practices in Model-Driven Web Engineering) under the WEBIST2016 (International Conference on Web Information Systems and Technologies) Conference making the review process of different articles.
	The doctoral student became part of the scientific committee of the journal "British Journal of Mathematics and Computer Science" making the review process of different articles.
2015	The doctoral student was part of the organizing committee of the THOT Project Dissemination Journeys.
	The doctoral student became part of the scientific committee of the journal "British Journal of Applied Science & Technology" making the review process of different articles.
2014	Since this year, the doctoral student is part of the scientific committee as an advisory researcher of the Ibero-American magazine GTI (Gerencia Tecnológica Informática). In addition to dissemination tasks, this participation has involved the review of different research articles submitted to the aforementioned journal.
2013	The doctoral student was part of the organizing committee, the program committee and scientific committee of the ISD2013 conference (International Conference on Information Systems Development) held in Seville (Spain) during the month of September 2013. Among other tasks, this participation has resulted in scientific dissemination, review of articles as well as management and organization of the event.

4. Publications

The following sections catalogue the research production of the doctoral student according to the typology of itself. In addition, this production is listed chronologically descending in each section.

4.1. Book Chapters

- Enríquez, J. G., Mayo, F. J. D., García, J. A. G., Cuaresma, M. J. E., & Risoto, M. M. (2017). A Framework to Manage Quality of Enterprise Content Management Systems. In Quality Control and Assurance-An Ancient Greek Term Re-Mastered. InTech.
- Enríquez, J. G., García-García, J. A., Domínguez-Mayo, F. J., & Cuaresma, M. J. E. (2017). ALAMEDA Ecosystem: Centering Efforts in Software Testing Development. In Quality Control and Assurance-An Ancient Greek Term Re-Mastered. InTech.

4.2. Journals

- González Enríquez, J., Domínguez-Mayo, F.J., García García, J. A., M.J. Escalona, M.J., An extension of NDT to model Entity Reconciliation Problems. J. Web Eng. Under review.
- Blanco, R., Enríquez, J.G., Domínguez-Mayo, F.J., Tuya, J., Escalona, M.J., Early Testing Integration for Entity Reconciliation in the context of Big Data. Journal of Systems and Software. Under review.
- García García, J. A., Urbieta, M., Escalona, M. J., Rossi, G. & González Enríquez, J. Identifying Functional Requirements Inconsistencies in Multi-Terms Projects Framed into a Model-Based Methodology. J. Web Eng. 16, 228–252 (2017).
- Domínguez-Mayo, F. J., Escalona, M. J., Mejías, M., Aragón, G., García-García, J. A., Torres, J., & Enríquez, J. G. (2015). A strategic study about quality characteristics in e-health systems based on a systematic literature review. The Scientific World Journal, 2015.
- Enríquez, J. G., Domínguez-Mayo, F. J., Escalona, M. J., Ross, M., & Staples, G. (2017). Entity reconciliation in big data sources: A systematic mapping study. Expert Systems with Applications, 80, 14-27.
- Enríquez, J. G., Mayo, F. J. D., & Cuaresma, M. J. E. Using MDE for the Reconciliation of Entities in Large Data Sources. http://ceur-ws.org/Vol-1812/JARCA16-paper-8.pdf
- Enríquez, J. G., Cid, V., Muntaner, N., Aroba, J., Navarro, J., Domínguez-Mayo, F. J., ... & Ramos, I. (2017). Behavior patterns in hormonal treatments using fuzzy logic models. Soft Computing, 1-12.

4.3. International Conferences

- Enríquez, J. G., Blanco, R., Domínguez-Mayo, F. J., Tuya, J., & Escalona, M. J. (2016, November). Towards an MDE-based approach to test entity reconciliation applications. In Proceedings of the 7th International Workshop on Automating Test Case Design, Selection, and Evaluation (pp. 74-77). ACM.

- García-García J., Urbieta M., Enríquez J. and Escalona M. (2016). Detecting Functional Requirements Inconsistencies within Multi-teams Projects Framed into a Model-based Web Methodology.In Proceedings of the 12th International Conference on Web Information Systems and Technologies - Volume 1: APMDWE, (WEBIST 2016) ISBN 978-989-758-186-1, pages 327-335. DOI: 10.5220/0005931803270335
- Enríquez, J. G., Domínguez-Mayo, F. J., Escalona, M. J., García García, J. A., Lee, V., & Goto, M. (2015). Entity Identity Reconciliation based Big Data Federation-A MDE approach.
- G Enríquez, J., Lee, V., Goto, M., J Domínguez-Mayo, F., & J Escalona, M. (2015, April). Entity Identification Problem in Big and Open Data. In Proceedings of the 17th International Conference on Enterprise Information Systems-Volume 1 (pp. 404-408). SCITEPRESS-Science and Technology Publications, Lda.
- Domínguez Muñz, M., Aroba, J., González Enríquez, J., Ramos Román, I., Lucena Soto, J. M., & Escalona Cuaresma, M. J. (2014). Advances in the Decision Making for Treatments of Chronic Patients Using Fuzzy Logic and Data Mining Techniques. In ICEIS 2014: 16th International Conference on Enterprise Information Systems (2014), p 325-330 (pp. 325-330). ScitePress Digital Library.
- Enríquez J., Domínguez-Mayo, F. J., García-García J. and Escalona M. (2016). An extension of NDT to model Entity Reconciliation Problems.In Proceedings of the 13th International Conference on Web Information Systems and Technologies - Volume 1: APMDWE, (WEBIST 2017)
- Garcia-Garcia, J.A., Enriquez, J.G., Garcia-Borgoñon, L., Arevalo, C. & Morillo, E. (2017, May). A MDE-based framework to improve the process management: The EMPOWER project. In IEEE 15th International Conference on Industrial Informatics (INDIN)

4.4.National Conferences

- Enríquez, J. G., Blanco, R., Domínguez-Mayo, F. J., Tuya, J., & Escalona, M. J. (2017, July). Towards an MDE-based approach to test entity reconciliation applications. In Jornadas de Ciencia e Ingeniería de Servicios (JCIS) 2017.

5. Projects

This section is divided in: research and transfer projects.

5.1. Research Projects

Name	Explorando Soluciones Guiadas para Sistematizar el Aseguramiento Temprano de la Calidad del Software. POLOLAS
Code	TIN2016-76956-C3-2-R
Scope	National
Main Researcher	Escalona-Cuaresma, María José
Initial Date	30/12/2016
Final Date	29/12/2019
Quantity (Euros)	181.200,00 €

Name	Mecanismos Guiados en Etapas Tempranas para la Mejora del Software. MeGUS
Code	TIN2013-46928-C3-3-R
Scope	National
Main Researcher	Escalona-Cuaresma, María José
Initial Date	01/01/2014
Final Date	31/12/2016
Quantity (Euros)	148.830,00 €

5.2. Transfer Projects

Name	Proyecto ADAGIO (ANALYTICS DATA AGGREGATED GEOLOCATION OPEN)
Code	P106-16/E09
Scope	National
Main Researcher	Domínguez-Mayo, Francisco José; Escalona-Cuaresma, María José
Initial Date	01/09/2016
Final Date	31/08/2018
Quantity (Euros)	75.000,00 €

Name	Proyecto CIPHER
Code	P047-14/E09
Scope	National
Main Researcher	Mejías-Risoto, Manuel; Sánchez-Gómez, Nicolás
Initial Date	01/03/2015
Final Date	31/10/2016
Quantity (Euros)	87.600,00 €

Name	Proyecto Empower
Code	P047-14/E09
Scope	National
Main Researcher	García-García, Julián Alberto; Domínguez-Mayo, Francisco José
Initial Date	01/03/2015
Final Date	31/10/2016
Quantity (Euros)	60.000,00 €

Name	Proyecto Alameda
Code	P046-14/E09
Scope	National
Main Researcher	Escalona-Cuaresma, María José
Initial Date	01/03/2015
Final Date	31/03/2016
Quantity (Euros)	45.000,00 €

Name	Proyecto Auriga (Algoritmo de cálculo y planificación de líneas de autocares y generación de turnos de perfiles de conductores)
Code	P005-15/E09
Scope	National
Main Researcher	Domínguez-Mayo, Francisco José; Sanchez-Gomez, Nicolas
Initial Date	09/02/2015
Final Date	08/07/2015
Quantity (Euros)	9.000,00 €

Name	Plataforma Interactiva en Cloud Computing VALORTIA (plataforma cloud de VALORación y certificación de competencias TIc Adquiridas)
Code	P012-14/E09
Scope	National
Main Researcher	Domínguez-Mayo, Francisco José
Initial Date	08/05/2014
Final Date	19/12/2014
Quantity (Euros)	32.701,06 €

Name	Soporte metodológico al proyecto SICATA
Code	P079-13 E09
Scope	Autonomic
Main Researcher	Escalona-Cuaresma, María José
Initial Date	01/10/2013
Final Date	31/10/2014
Quantity (Euros)	50.114,00 €

Name	Asesoramiento y Desarrollo de aplicaciones informáticas en el ámbito clínico de Trasplantes.
Code	P005-13/E09
Scope	National
Main Researcher	Domínguez-Mayo, Francisco José
Initial Date	07/02/2013
Final Date	13/02/2015
Quantity (Euros)	31.854,05 €

6. SOFTWARE INTELLECTUAL PROPERTY RECORDS (PATENTS)

During the research career of the doctoral student, he also has made some software intellectual property records (patents) in the University of Seville. Chronologically descending, those are:

- Gomez Barea, David, Morillo Baro, Esteban, Gonzalez Jimenez, Javier, Sanchez Gomez, Nicolas, García García, Julián Alberto, et. al.: Alameda. Patente de invención, Propiedad industrial. Solicitud: 2016-04-19
- Gonzalez Jimenez, Javier, J.G. Enríquez, Cabello Ruiz, Rafael, Morillo Baro, Esteban, Sanchez Gomez, Nicolas, et. al.: Evaluador. Patente de invención, Propiedad industrial. Solicitud: 2015-05-04
- J.G. Enríquez, F.J. Domínguez-Mayo, M.J. Escalona et. al.: MaRIA Tool. Patente de invención, Propiedad industrial. Solicitud: 2017-05-04

7. RESEARCH NETWORKS

During the research career of the doctoral student, he also has been part of some research networks. Chronologically descending, those are:

- Thematic Network for the quality solutions software development in PLM environments (SoftPLM Network). TIN2015-71938-REDT.
- Mexican Thematic Network in Software Engineering (244467) of the National Council of Science and Technology of Mexico in the call Conacit Thematic Networks 2014.