A Systematic Requirements Engineering Approach for Decision Support Systems

Master in Information Technology for Business Intelligence (IT4BI)

By: **Stephany Garcia**

5 September 2014
Barcelona, Spain

Advisors:
Dr. Oscar Romero
Dr. Ruth Raventós

IT4BI

ULB UNIVERSITÉ LIBRE DE BRUXELLES

UNIVERSITÉ FRANÇOIS - RABELAIS TOURS

UPC UNIVERSITAT POLITÈCNICA DE CATALUNYA BARCELONATECH

A thesis presented by ***Stephany Garcia***
in fulfillment of the requirements for the degree of
Master in Information Technology for Business Intelligence - Erasmus Mundus

Abstract

Decision Support Systems have emerged as a dominant technology capable to integrate heterogeneous sources into an analytical fashion to facilitate and provide a better decision-making process. Successful projects of Decision Support Systems implementation have confirmed a high-level of user satisfaction and return on investment. Despite the potential of these systems, several surveys have indicated that the failure rate of Decision Support System projects in case studies and literature is considerably high. The issue starts from setting the wrong requirements by approaching Decision Support Systems as Operational Systems, without considering that their development is rather different. Particularly, Decision Support Systems are expected to: deal with diverse domain terminology causing poor communication between business users and IT people, integrate heterogeneous sources demanding complex procedures to control the integration and transformation phases, produce high-quality documentation, create analytical results expected to be explored across multiple levels (e.g. dimensions), provide a validated and verified output, provide traceability of sources, and a comprehensible MD (Multidimensional) Design is mandatory. In addition to the aforementioned characteristics, at the early stages of Decision Support System projects, typically, business needs are manually collected (e.g. interviews, etc.) to be then translated into a conceptual MD and ETL Designs; this manual process tends to be error-prone, that demands several rounds of redesign to satisfy all the business requirements stated by the stakeholders.

On the basis of the evidence currently available, it seems fair to suggest that leading Information Systems professionals lack of an adequate approach to determine information needs specifically for Decision Support Systems. Inasmuch as, it was needed to define a novel, Requirements Engineering approach, tailored to Decision Support Systems: We have named it ***"RE4DSS" – (Requirements Engineering for Decision Support Systems).*** *Our systematic and block-oriented approach RE4DSS, is coupled with a set of models and techniques, guiding the developer throughout the activities of elicitation, negotiation and documentation to produce goals, scenarios and solution-oriented requirements, followed with the management and validation activities. As RE4DSS aims to improve the manual practices by systematizing the requirements process, while it is accompanied by a system called GEM (Generating ETL and MD Designs) to provide a semi-automatic means translating each of the stakeholders' requirements to finally produce the appropriate and unified ETL and MD Designs. To pursue our main goal, we present a profound literature classification to address the complexities of Requirements Engineering for Decision Support Systems. In closing, a real case study conducted by the Technical University of Catalonia (UPC) in collaboration with the World Health Organization (WHO) demonstrates how RE4DSS approach has been successfully applied during the requirements process to create the ETL and MD Designs of an information surveillance system to control/eliminate the neglected Chagas disease.*

Acknowledgements

I would like to express my gratitude to:

I wish to sincerely thank both of my supervisors, Oscar Romero and Ruth Raventós for their continuous support, guidance and constructive criticism throughout the course of the study. Their profound knowledge and interest in the field enabled them to provide me with the right guidance to achieve a successful work.

My deepest gratitude goes to my immediate family, to whom this dissertation is dedicated. My Father, Juan Antonio, who showed me the joy of the intellectual pursuit ever since I was a child. My Mother, Luz Elena, whose unconditional love has been my greatest strength. My brother Ivan and Raquel, for their continuously invaluable advice. Last but not least, my niece Maria Lúcia, for brightening each one of my days.

And most importantly, to the one who made everything possible. I thank God for giving me the unique opportunity to learn and live with Him every day during my work and providing with strength and wisdom to pursue this degree.

Stephany García

Contents

Abstract ii

Acknowledgements iii

Contents v

List of Figures vii

List of Tables xi

1 Introduction 1

1.1 Context . . . 1

1.2 Motivation . . . 2

1.3 Objectives . . . 4

1.4 Structure of the Document . . . 5

2 Setting the Context 7

2.1 PART #1: Requirements Engineering Context . . . 7

2.2 PART #2: Business Intelligence & Decision Support Systems Context . . . 27

3 Current Approaches of Requirements Engineering in Decision Support Systems 35

3.1 Reference Framework . . . 35

3.2 Scope of Classification . . . 37

3.3 Literature Classification . . . 39

3.4 Complete Graphical Representation . . . 61

3.5 Chapter Conclusions . . . 64

4 RE4DSS: A Systematic Approach of Requirements Engineering for Decision Support Systems 67

4.1 RE4DSS Approach Overview . . . 68

4.2 Core Activities: . . . 71

4.3 Requirements Artifacts: . . . 73

4.4 Final Output - Unified Schema and ETL Design . . . 89

5 Case Study:

Data Warehouse Requirements for the global WHO surveillance system to eliminate Chagas disease 91

5.1 Background - Chagas Disease . . . 91

5.2 Methodology . . . 93

5.3 Core Activities . . . 95

5.4 Requirements Artifacts . . . 101

5.5 Automatic Verification & Management Activities . . . 118

5.6 Final MD & ETL Design . . . 119

5.7 Validation & Exploration Requirements . . . 120

6 Conclusions and Further Work 123

Bibliography 127

List of Figures

1.1 Master Thesis Overriding Structure 5

2.1 The Requirements Engineering Framework. Reprinted from: (Poh10) 11
2.2 Four Context Facets. Reprinted from: (Poh10) 12
2.3 Requirements Artifacts Reprinted from: (Poh10) 13
2.4 Example of goal modelling using AND/OR trees. Reprinted from: (Poh10) 14
2.5 Example of a strategic rationale model in i*. Reprinted from: (Poh10) 15
2.6 Example of a goal model in KAOS. Reprinted from: (Poh10) 15
2.7 An example of a template for the use case specification. Reprinted from: (Tag03) 17
2.8 Documentation of a scenario in a sequence diagram (Left). Documentation of the control flow of scenarios (Right). Reprinted From: (Poh10) 17
2.9 Coarse-grained characterisation of artifacts and their relationship. Adapted from (Poh10) 18
2.10 Model-based documentation of the three perspectives. Reprinted from: (Poh10) 19
2.11 Requirements Engineering Core Activities. Reprinted from: (Poh10) 20
2.12 Identification of Relevant Requirement Sources (Two-Step Procedure). Reprinted from: (Poh10) 21
2.13 Cross-sectional activity: validation (left-hand side). Reprinted from: (Poh10) 22
2.14 Goal and scenario-based selection of requirements for prototype development). Reprinted from: (Poh10) 24
2.15 Requirements Engineering Management. Reprinted from: (Poh10) 25
2.16 Requirements Engineering Management. Adapted from: (Poh10) 25
2.17 Left: Traceability matrix for several relationship types. Right: Representation of traceability information by means of traceability graphs. Reprinted from:(Poh10) 26
2.18 A very clearcut dividing line exists between operational processing and Decision Support Systems informational processing. Reprinted from: (IIS02) 28
2.19 Data Warehousing Architecture. Reprinted from (CD97) 31

2.20 Three-dimensional cube modeling inspected_dwellings: Entomologists from WHO want to visualize in maps and compare the inspected dwellings in Argentina where there is at least one nymph insect of the specie Triatoma_pseudomaculata on 14/12/2013. Adapted from (GR09) 32

3.1 Column Definition for Literature Approaches Table based on (Poh10) 36
3.2 Literature Selection and Classification Process 37
3.3 Specifications for Literature Classification Table 39
3.4 Elicitation Activity - Proposed Approaches in Literature 41
3.5 Elicitation - Pohl's Framework Instantiation for Decision Support Systems 44
3.6 Negotiation Activity - Proposed Approaches in Literature 44
3.7 Negotiation - Pohl's Framework Instantiation for Decision Support Systems 45
3.8 Goals Artifact - Proposed Approaches in Literature 47
3.9 Tropos (Left) and Soft-goal Interdependency Graph (Right) reprinted from (Pro14) and (Pro12) 48
3.10 Goals - Pohl's Framework Instantiation for Decision Support Systems 49
3.11 Scenarios Artifact - Proposed Approaches in Literature 50
3.12 Scenarios - Pohl's Framework Instantiation for Decision Support Systems 51
3.13 Solution-Oriented Requirements Artifact - Proposed Approaches in Literature 52
3.14 Bus Matrix (Left) and Crud Matrix - Function Decomposition Table (Right) based on (Kim98) and (IP01) 54
3.15 Solution Oriented Requirements - Pohl's Framework Instantiation for Decision Support Systems 55
3.16 Management Cross-Sectional Activity - Proposed Approaches in Literature 56
3.17 Requisite Pro - Traceability Matrix (Left) and 2by2 Priority Matrix (Right) based on (Kim98) and (Woo97) . . . 57
3.18 Management - Pohl's Framework Instantiation for DSS 58
3.19 Validation Cross-Sectional Activity - Proposed Approaches in Literature 59
3.20 Requirements Checklist for Data Warehouse, based on (PdC03) 60
3.21 Validation - Pohl's Framework Instantiation for DSS 61
3.22 Pohl's Framework Instantiation for Decision Support Systems 62
3.23 Pohl's Framework Instantiation for Decision Support Systems 63

4.1 Requirements Engineering 4 Decision Support Systems (RE4DSS) Approach 69
4.2 Elicitation Process (RE4DSS) Approach 71
4.3 Goal Classification Tree Levels - Extracting Information Goals and Creating Use Cases 74
4.4 Goal's Classification Process and Scenario Use Cases - (RE4DSS) Approach 75
4.5 Components of the Solution Oriented Requirements - (RE4DSS) Approach 76
4.6 Domain Requirments and Quality Properties illustrated as Structured Requirements in XML 77
4.7 (RE4DSS) Approach - Vocabulary Specification 79
4.8 Ontology Source and Source Mapping - (RE4DSS) Approach 79
4.9 Relational Data Source represented by means of OWL Ontology. Example extracted from (Jov11) 80

4.10 Mapping of an ontology datatype property. Example extracted from (Jov11) . . . 81
4.11 Solution-Oriented Requirements and GEM System - (RE4DSS) Approach . . . 82
4.12 Inputs for GEM system - (RE4DSS) Approach . . . 83
4.13 GEM System Architecture - Reprinted From (RSA11) . . . 85
4.14 Requirements Verification Method. Reprinted from (RSA11) . . . 85
4.15 Annotated Ontology Subset & Verification - (RE4DSS) Approach . . . 86
4.16 Algorithm - Defining Paths Between Concepts. Reprinted from (RSA11) . . . 87
4.17 Verification, Management and Validation - (RE4DSS) Approach . . . 88
4.18 Unified MD design and ETL design - (RE4DSS) Approach . . . 90

5.1 Blood-sucking insect responsible for the Chagas disease. Reprinted from: (WHO13) . . . 92
5.2 Global distribution of cases of Chagas disease, based on official estimates, 2006 - 2010. Reprinted from: (WHO13) 93
5.3 Chagas Case Study Methodology . . . 94
5.4 Elicitation of Business Users and source data stores. Adapted from: (RSA11) . . . 96
5.5 (RE4DSS) Approach - Chagas Case Study Vocabulary Creation. Reprinted from (VN14) . . . 98
5.6 (RE4DSS) Approach - Standard structure for software requirements specification according to [IEEE Std 830-1998]. Reprinted from (Poh10) . . . 100
5.7 Goal Classification Tree of Chagas Case Study - RE4DSS . . . 102
5.8 Goal Classification Tree of Chagas Case Study Identifying Requirements - RE4DSS . . . 103
5.9 Complementing Vocabulary with Use Cases - RE4DSS . . . 106
5.10 Solution-Oriented Requirements & GEM - RE4DSS . . . 106
5.11 XML Defining Structured Requirements: Levels Measures and Slicers - RE4DSS . . . 108
5.12 XML Defining Structured Requirements: Quality Properties - RE4DSS . . . 109
5.13 Representing Relationships Between Dwelling and Inspection Tables - RE4DSS . . . 110
5.14 Representation of Source Tables in OWL Classes - RE4DSS . . . 110
5.15 Attributes Representation of Dwelling Table using OWL - RE4DSS . . . 111
5.16 Attributes Representation of Inspection Table using OWL - RE4DSS . . . 111
5.17 Source Associations “BelongsTo” Between Tables Dwelling and Inspection - RE4DSS . . . 111
5.18 Cardinalities Between Dwelling and Inspection Tables 1..1 and 0..* - RE4DSS . . . 112
5.19 Vocabulary Representation - Vectorial Intervention Hierarchy - RE4DSS Reprinted from (VN14) . . . 113
5.20 Vocabulary Representation - Geographic and Temporal Hierarchy - RE4DSS Reprinted from (VN14) . . . 114
5.21 Vocabulary Representation - Intervention Hierarchy - RE4DSS Reprinted from (VN14) . . . 114
5.22 Prototype Validation of Vocabulary Shared Ontology - Individual Dwelling Inspection.” . . . 115
5.23 Source Mapping - Mapping of an ontology class - RE4DSS . . . 116
5.24 Source Mapping - Mapping of an ontology datatype property - RE4DSS . . . 117
5.25 Source Mapping - Mapping of an ontology object property - RE4DSS . . . 117
5.26 Source Mapping - Mapping of an ontology datatype property - RE4DSS . . . 117

5.27 Consolidation of Final MD & ETL Design - RE4DSS 120
5.28 Prototype Validation of MD Design - “Data about infested dwellings per insecticide used and area.” 121
5.29 Prototype Validation of MD Design - “Graphics showing the temporal variation of evaluated inspected houses by department every two years for La Rioja.” 122

6.1 Literature Analysis of DSS Requirements Engineering 133
6.2 Literature Analysis of DSS Requirements Engineering 134
6.3 Literature Analysis of DSS Requirements Engineering 135
6.4 Literature Analysis of DSS Requirements Engineering 136
6.5 Literature Analysis of DSS Requirements Engineering 137
6.6 Vocabulary Representation - Domain Model/Data Analysis. Reprinted from (VN14) 138
6.7 Vocabulary Representation - Domain Model - Intervention Hierarchy/Treatment. Reprinted from (VN14) 139
6.8 Vocabulary Representation - Domain Model - Intervention Hierarchy/Diagnosis. Reprinted from (VN14) 140
6.9 Vocabulary Representation - Domain Model - Intervention Hierarchy/Drug Availability. Reprinted from (VN14) . 141
6.10 Vocabulary Representation - Domain Model - Intervention Hierarchy/Normative. Reprinted from (VN14) 142
6.11 Vocabulary Representation - Domain Model - Intervention Hierarchy/Systematic. Reprinted from (VN14) 143

List of Tables

2.1 A comparison between Operational and Decisional Systems. Reprinted from: (Rom11) 29

Part I

Preface

CHAPTER

"You can have data without information, but you cannot have information without data."

– Daniel Keys Moran

1 Introduction

1.1 Context

The World Health Organization (WHO), a United Nations (UN) agency founded in 1948, is the directing authority for health matters, responsible to provide leadership on global health matters by setting standards and providing technical support to monitor the health trends (Org14). In October 2010, WHO launched the First Report on Neglected Tropical Diseases (NTD) (SDC10), which included the "Chagas disease" among others. Chagas disease (also known as Human American Trypanosomiasis) is classified as a life-threatening disease caused by a parasite named 'protozoan parasite.' According to the report, Chagas is nowadays usually found in 21 Latin American countries, where the disease is transmitted to humans by the faeces of infected triatome bugs. The WHO report has estimated seven to eight million people infected worldwide, although the concern is that the number of infected people keeps increasing year by year.

By being categorized as NTD, not much attention is placed to the Chagas disease. In fact, the worldwide population is not aware that statistics indicates that NTDs have already caused substantial illness for more than one billion people of our world's poorest countries and causing death to 534,000 people every year (fDCP11). Due to the worrying situation, in 2010 the World Health Assembly (WHA) decided to approve the "Chagas disease: control and elimination" resolution (SDC10). But it was not until 2013 that the Tricycle Strategy of the Programme on Control of Chagas disease was presented (VN14). The presented strategy had the objective to advance in the disease control by creating a Chagas Information Database (CID) system for surveillance to raise awareness on the Chagas disease. In particular to facilitate the access to information from different sources related to Chagas disease, to then exploit the data by the means of dashboards, and analytical tools (e.g. visualize disease statistics, maps and diagrams for transmission routes, infested patients, etc).

The CID project involves specialized stakeholders from very different domains such as health specialists, entomologists, national authorities and software analysts, among others, who have complementary yet non-trivially integrable perspectives of the Chagas disease. For instance, entomologists set their focus on studying the triatome bugs (the main disease vector) and how to interrupt this transmission by such vectors. Doctors, in turn, focus on patient diagnosis and treatment and the WHO monitors all relevant aspects of the disease, including provided infrastructures and drug delivery in each country, and provides

the multi-perspective expertise needed to glue all of them together. In consequence to this, software analysts are expected to deal with the highly-complex domain of the Chagas disease to begin collecting the requirements before creating the CID system.

For the aforementioned characteristics of involving extremely domain specialized stakeholders, the Requirements Engineering process resulted to be incredibly challenging within the context of the CID project and promoted the research conducted in this master thesis: Requirements Engineering for complex Decision Support Systems (DSS).

1.2 Motivation

According to the statistics, data management and analytics will grow at a compound annual growth rate of 10%, being twice as fast as the software business (The10). As data grows massively, more and more decision-makers and stakeholders just like the WHO are demanding computerized support of their work by requesting an intelligent solution able to analyze/visualize their data to achieve their goals and make the best decisions. As stated by (PdC03), data growth has demanded the creation of Decision Support Systems that can be expanded quickly and on an efficient manner while integrating heterogeneous sources into an analytical fashion in order to enable decision-makers and stakeholders to make the best choices. In addition, the author confirmed that successful projects of Decision Support Systems implementation have demonstrated high-level of user satisfaction and return on investment.

Despite the potential of these systems, several surveys have indicated that the failure rate of Decision Support System projects in case studies and literature is considerably high (GRG08). The issue starts from setting the wrong requirements by approaching Decision Support Systems as operational systems, without considering that their development is rather different (PdC03). In contrast to operational systems, Decision Support Systems are expected to deal with the following:

- **Complexity of Decision Support Systems:** Unlike operational systems, Decision Support Systems such as Data Warehousing are a complex and expensive component of the corporate's IT infrastructure involving a conglomerate of extraction, cleansing and transforming systems, Data Warehouse database(s), Data Marts, metadata management system, usually presented in a summarized format for visualization purposes (WS03). For instance, the CID project consisted of gathering information from heterogeneous sources such as: hospitals using databases to record the infected patients, researchers using excel forms to record data about the raids and dwellings inspections, and health minister officers managing information about regulations in PDF format; making this a complex process to build the CID system.

- **Integration with diverse data sources:** Decision Support Systems collect data from heterogeneous sources including the operational systems or other external sources. Therefore, procedures and requirements must be specified for the data collection and integration (PdC03). Plus interfaces and connections to already existing systems (inside or outside the organization) must be considered (WS03). During the CID project, heterogeneous sources mentioned in the previous example had to be integrated to present a single unified interface; thus, problems of heterogeneous source integration were present such as: *technical heterogeneity* (integrating different file formats and databases such as Microsoft Excel,

PDFs, and SQL Server databases) and *semantic heterogeneity* (identifying and integrating concepts from the sources which semantically correspond to one another, such as "kiss bug" in the U.S.A is semantically equivalent to "vinchuca" in Argentina).

- **Expressing Decision Support Systems requirements:** At early stages of the Decision Support Systems project, the future users of the system have difficulties to express their requirements (CMM97). In contrast to operational systems which are based on consistent specifications (WS03). For instance, in the CID project, the stakeholders aimed to collect as much information as possible with the maximum level of detail (e.g. number of insects per house). However, it is too ambitious to obtain this level of detailed data, plus the system could become too complex. Hence, an agreement among the stakeholders had to be reached by negotiating the right balance of having information with enough level of detail (i.e. aggregated data per certain area) to impact the main objective of elimination of Chagas. As believed by Paul Valéry: *"What is simple is wrong, what is complicated is useless"* (Val37).

- **Identifying relevant subjects and perspectives of analysis:** The multidimensional model has become the de facto standard for Decision Support Systems that categorize the data as facts (representing a business activity) or dimensions (perspective of the data) being usually textual (Ped09). While analyzing user requirements, in Decision Support Systems it is crucial to identify the subject of analysis by perceiving the metrics behind the user demands. Also to determine the extent to which a metric can be operated (summarized, counted, etc.) along with the different perspectives from where to analyze the relevance subjects of analysis (PdC03). As discussed later, this specific analysis-oriented conception requires considering specific data models.

- **Data exploration and summarization:** It is needed to firstly discuss the relevance of exploring data from different points of view, meaning changing the data granularity (data detail level) on demand. To do so, summarizability (assuring the correctness of summary operations) is a must; specifically to validate the correctness of aggregated results for whichever combination between facts and dimensions (PdC03). Summarizability is done in order to avoid erroneous conclusions and decisions (jLS97). Thus, it is important to express the requirements constraint regarding the aggregated data. For instance, in the CID project, visual representations of hierarchical structures in space and time were required, for instance, showing "Infected Dwellings" for Argentina regions over a period 2008-2014 to detect potential infected geographical areas.

- **Fast track of user requirements changes:** Decision Support Systems are in constant evolution and must face changes during its development that may impact on the models built to meet the end-user requirements (PdC03). The issue is that the relationship between the requirements, elements in the model and data sources are lost in the process since no traceability is specified neither documented (MT12). Accordingly, track of all the requirements and changes must be kept in Decision Support Systems during the complete project's life-cycle. For instance, showing requirements dependencies with cross-references analysis to demonstrate the impact of a certain change in a fact or dimension (e.g. traceability matrices) (TCM02).

- **High-quality documentation and Standard Vocabulary:** Decision Support Systems projects use preexisting operational documentation in order to define the data extraction and integration procedures; however, legacy documentation usually lacks quality, which makes the task of extracting the requirements harder (PdC03). In addition, a vocabulary, is necessary to identify definitions, acronyms and abbreviations for establishing lexis to be shared among all parties related to the project (BDGM08). For instance, during the CID project, health specialists (e.g. entomologists) certainly use medical and domain terminology for their every day communication such as "trypanosomiasis and Triatominae", which software analysts and neither doctors were able to understand, and neither to match these with the heterogeneous sources. For this reason, there is a huge need to bridge the gap between stakeholders and IT people by creating a Vocabulary. However, not much advantage can be taken from the vocabulary in a plain document, but certainly there is a need to translate this vocabulary into a machine-readable format with a cross-reference structure to be used when creating the future schema for the system.

Although several efforts have been made to utilize the traditional Requirements Engineering process for Decision Support Systems, we have found during our study that leading Information Systems professionals lack of a reference approach tailored specifically for Decision Support Systems to determine information needs.

1.3 Objectives

Given our motivation, we aim at defining a novel, Requirements Engineering approach, tailored to Decision Support Systems. Therefore, this thesis main goal is to propose **"RE4DSS"** *(Requirements Engineering for Decision Support Systems)*: A systematic, block-oriented approach, coupled with a set of models and techniques, guiding the developer throughout the activities of elicitation, negotiation and documentation to produce goals, scenarios and solution-oriented requirements, followed with their management and validation activities. To achieve the conception of RE4DSS, we have decomposed our main objective into three sub-objectives:

1. Analyze the current approaches of Requirements Engineering for Decision Support Systems and create a profound literature classification outlining the proposed different models, techniques and guidance.
2. Propose a systematic approach entitled "Requirements Engineering For Decision Support Systems" (RE4DSS) based on the thorough view of the domain gained during the first sub-objective.
3. Validate our approach (RE4DSS) with a real case study conducted by the Technical University of Catalonia (UPC) in collaboration with the WHO by applying RE4DSS during the requirements process to create the respective ETL & MD Designs of an information surveillance system to control/eliminate the neglected Chagas disease (CID project introduced in the Context section).

1.4 Structure of the Document

Our study is organized in the following chapters (see figure 1.1):

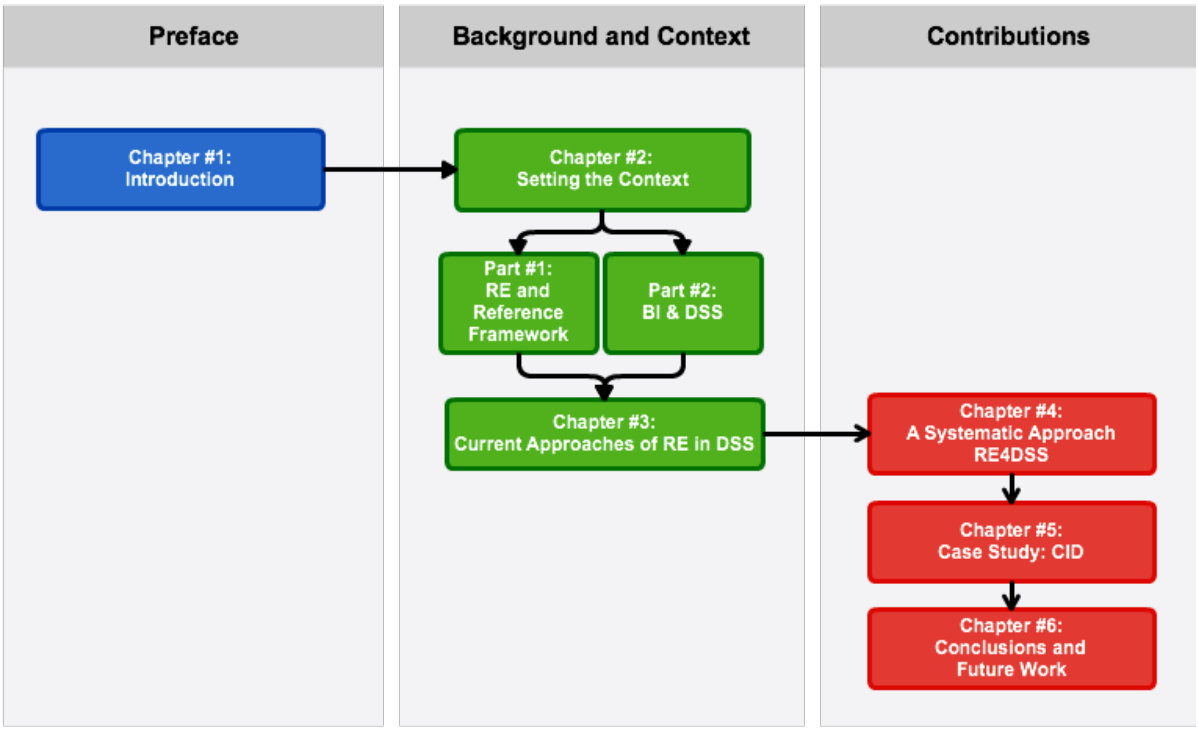


Figure 1.1: Master Thesis Overriding Structure

Chapter 2 - Setting the Context: This context chapter is divided in two main parts (sections): Requirements Engineering, and Business Intelligence & Decision Support Systems. In Part #1, we briefly introduce the importance of the Requirements Engineering field, the main concepts and the different types of requirements. Followed by the description of the "Requirements Engineering Framework" proposed by Klaus Pohl in 2010. This includes the system context, requirements artifacts, solution-oriented requirements, validation and management elements. This framework will be later used as a reference in following chapters to fit the Decision Support System activities.

Part #2 of this chapter incorporates the basic concepts of "Business Intelligence" narrowing to "Decision Support Systems" also while identifying the differences between these systems and operational systems. All these concepts represent the background needed to understand the following chapters.

Chapter 3 - Current Approaches of Requirements Engineering in Decision Support Systems: This chapter reviews the literature of Requirements Engineering for Decision Support Systems to obtain a good grasp of the main published work. In order to compare and better comprehend these approaches, we will map the found approaches to Pohl's framework to enable a classification. Hence, the first subsection aims to outline the main reasons why we have used a reference framework (by Pohl). Followed by the scope, which details the selection of the literature (literature search). Next, we present the classification tables of the literature mapped to the pre-defined blocks of our Requirements Engineering reference framework. And finally, at the

end of this chapter, we provide a summary with the complete overview of the practices mapped into the reference framework and all the elements integrated for easier understanding. This will be later used in chapter 4 as the base in order to create our own approach RE4DSS and then implement it on the Chagas Case Study in chapter 5.

Chapter 4 - A Systematic Approach Requirements for RE in Decision Support Systems: In this chapter, we present the main contribution of this master thesis. We formally describe our proposed approach entitled ‘‘*Requirements Engineering for Decision Support Systems"* along with all its components and its relationship to the reference framework. The sections in this chapter describe RE4DSS starting with a complete overview. Subsequently each of the elements in our approach are mapped to our reference framework. Section 4.2 focuses in the Core Activities, Section 4.3 in Requirements Artifacts, Section 4.4 includes the description of the GEM *(Generating ETL and Multidimensional Designs)* system and its role for the Solution-Oriented Requirements. And last but not least, we describe the Verification, Validation and Management activities in our approach before describing the final output.

Chapter 5 - "Chagas Case Study": This final chapter presents the contributions of our RE4DSS approach introduced in chapter 4, employed successfully to a real case study provided by the WHO to advance in disease control by creating a Chagas Information Database system for surveillance and elimination of the disease. This chapter initializes with the description of the Chagas Disease, along with its background. Followed by the methodology used to implement our RE4DSS approach into the Chagas case study with the assistance of the GEM system. Finally, we implement each of the RE4DSS practices into the case study.

Chapter 6 - Conclusions and Further Work: This final study provides the conclusions of the work at hand. It presents a summary of the contributions and points out possible future work specifically in the field of traceability of metadata.

Part II

Background & Context

CHAPTER

"Data is a precious thing and will last longer than the systems themselves."
– Tim Berners-Lee

2 Setting the Context

The objective of this chapter is to present the background concepts needed to understand the following chapters. Therefore, this second chapter is divided into two main parts: Part#1 Requirements Engineering and Part#2 Business Intelligence & Decision Support Systems. This first section includes the importance of the Requirements Engineering field along with its concept and its existing types. Later, we provide a description of the "Requirements Engineering Framework" proposed by Klaus Pohl and an explanation of its different elements such as techniques, models, etc. During Part #2 of this chapter, we will dig into the concepts of Business Intelligence from the business and IT perspectives. Followed by a comparison between Decision Support Systems and Operational Systems. Finally, we describe the Data Warehousing architecture and On-Line Analytical Processing (OLAP).

2.1 PART #1: Requirements Engineering Context

The Importance of Requirements Engineering

Nowadays, innovative software systems are being developed with higher quality standards than the past, and at lower costs (Poh10). Consequently this has become a great opportunity and at the same time a big challenge for the developers to create successful projects. According to the Standish Group Research, a study done in 2013 regarding why projects succeed or fail (representing more than 90,000 completed IT projects), the results provide confirmatory evidence that inappropriate analysis of Requirements Engineering is one of the top reasons that leads to project failures (Int13). In summary, during this study from the Standish Group, the figures revealed the following: 43% of the IT projects were challenged (late, over budget, and/or with less than the required features and functions) and 18% failed (cancelled prior to completion or delivered and never used). The biggest problem is that successful projects according to the study (completed on-time and on-budget) were only 39%, which is less than half.

On a deeper study done by the Standish Group, they identified the following reasons why the finished projects overspent the estimated resources and/or restricted the implemented functionality (Cla95):

1. Lack of User Involvement 12.8%
2. Incomplete Requirements 12.3%
3. Changing Requirements 11.8%
4. Lack of Executive Support 7.5%
5. Technology Incompetence 7.0%
6. Lack of Resources 6.4%
7. Unrealistic Expectations 5.9%
8. Unclear Objectives 5.3%
9. Unrealistic Time Frames 4.3%
10. New Technology 3.7%
11. other 23.0%

As seen by the figures above, the underlined bullets 1, 2, 3, 7 and 8 are directly related to insufficient and poor Requirements Engineering that summed together are equal to of 48%. Meaning that almost half of the reasons are related to Requirements Engineering issues. In consequence, according to the work (AW05), researchers agree that the Requirements Engineering process has moved from being the first phase in software development cycle to a key activity that spans across the complete software development life-cycle in many organizations. Further, the same work suggests that understanding and modeling of Requirements Engineering practices has become a crucial step towards improving the projects to achieve successful software aligned to the company's objectives.

The Concept of Requirements Engineering

Every single project must begin with the statement of a requirement. In computer science, a requirements refers to the description of how a software product should perform (AW05). The IEEE 610.12-1990 defines "requirement" as follows:

Definition: **Requirement**

1. "A condition or capability needed by a user to solve a problem or achieve an objective"
2. "A condition or capability that must be met or possessed by the system or system component to satisfy a contract, standard, specification, or other formally imposed document"
3. "A documented representation of a condition or capability as in (1) or (2)"

Reference: (Iee90)

Therefore requirement is composed of a collection of needs that arise from the users and various other stakeholders (general organization, government bodies, community and industry standards) (AW05). The main distinction of defining the requirements and developing the design of a system is that the requirements focus on "what" the system should do, rather than "how" it should be done (AW05). Meaning that the requirements are centered on defining the problem "what should be developed," while the design specifies "how the system should be developed" (Poh10). Requirements may be classified in different types; these are explained in the following subsection.

Types of Requirements

In the literature, requirements are found to be classified in many ways. Klaus Pohl makes the distinction of three main types of requirements (Poh10):

- **Functional Requirements:** What the system will do. Describe the systems functionalities in detail: input and output, exceptions, and other.
- **Quality Requirements:** Define the quality properties of the system to be developed (e.g. system's performance, reliability or stability). These are also known by the literature as non-functional requirements.
- **Constraints:** Organizational or technological requirements that restrict the way in which the system shall be developed (e.g. cultural, legal, physical and project's constraints).

After understanding what is a requirement and the types of existing requirements, we may now define the concept of the topic of this thesis - Requirements Engineering:

> Definition: **Requirements Engineering** is a cooperative, iterative and incremental process which aims to ensure the following:
>
> 1. All relevant requirements are explicitly known and understood at the required level of detail.
> 2. Sufficient agreement about the system requirements is achieved between the stakeholders and the involved.
> 3. All requirements are documented and specified in compliance with the relevant documentation/specification formats and rules.
>
> Reference: (Poh10)

In the present work from (Poh10), Requirements Engineering is performed throughout the entire development process of a system/product and starts with the goal to change the current reality called a "vision". Although, the author argues that the information in the vision is not sufficient to define all the requirements in the system at the required level of detail (as the definition states). For instance, diverse sources must be elicited (e.g. customers, system users, domain experts, etc), existing documentation (e.g. laws, guidelines, business rules), and existing systems (e.g. legacy systems). For this reason, the

author decided to create a Requirements Engineering Framework being possible to define the major structural elements of the Requirements Engineering Process (read following section for more details).

Requirements Engineering Framework by Klaus Pohl

Today, the use of software-intensive systems has become part of our everyday life while at the same time is gaining importance in an exponential manner due to its high demand. Therefore, as explained at the beginning of this chapter, Requirements Engineering is considered as a key for the development of software systems in order to decrease the risk of failure when collecting, analyzing and documenting the needs and intentions from the stakeholders. For the sake of our study, we have decided to use a framework entitled "Requirements Engineering framework" proposed by Klaus Pohl in order to explain the main building-blocks from the Requirements Engineering field. This framework is described in the following subsection. **Notice that this following entire subsection regarding the framework is intended to give a background and summary of the Requirements Engineering framework. Hence, it is entirely based on the book entitled "Requirements Engineering - Fundamentals, Principles, and Techniques" by Klaus Pohl** (Poh10).

Framework Overview

Klaus Pohl proposed the "Requirements Engineering Framework" which main purpose is to define the main structural blocks of a Requirements Engineering process. This framework is intended to convert the stakeholder's needs to goals, scenarios and solution-oriented requirements by iterating the following activities: elicitation, documentation, and negotiation.

The reasons why this Requirements Engineering framework was chosen are: First, it is a well recognized framework in the Requirements Engineering Field. Second, it provides a comprehensible and well-structured base for the fundamentals, principles and techniques of requirements engineering; while it is not adhered to a specific methodology, neither to any type of software project. Consequently this makes it easier to adapt and utilize depending on the software project. Moreover, third, this framework has been successfully validated and transferred to the industry using it as a reference for training their staff (e.g. managers, engineers, developers).

The "Requirements Engineering Framework" (see figure 2.1) consists of three main building blocks and two cross-sectional activities which are explained in the following sections of this document:

- **System Context:** The system is divided into four context facets, namely: Subject, Usage, IT system, Development. (See "System Context" section, P. 12)
- **Requirements Artifacts:** Goals, scenarios, and finally solution-oriented requirements. (See "Requirements Artifacts" section, P. 13)
- **Core Activities:** Elicitation, Documentation, and Negotiation which are performed iteratively with the aim to establish the vision within the existing context. (See "Core Activities" section, P. 20)

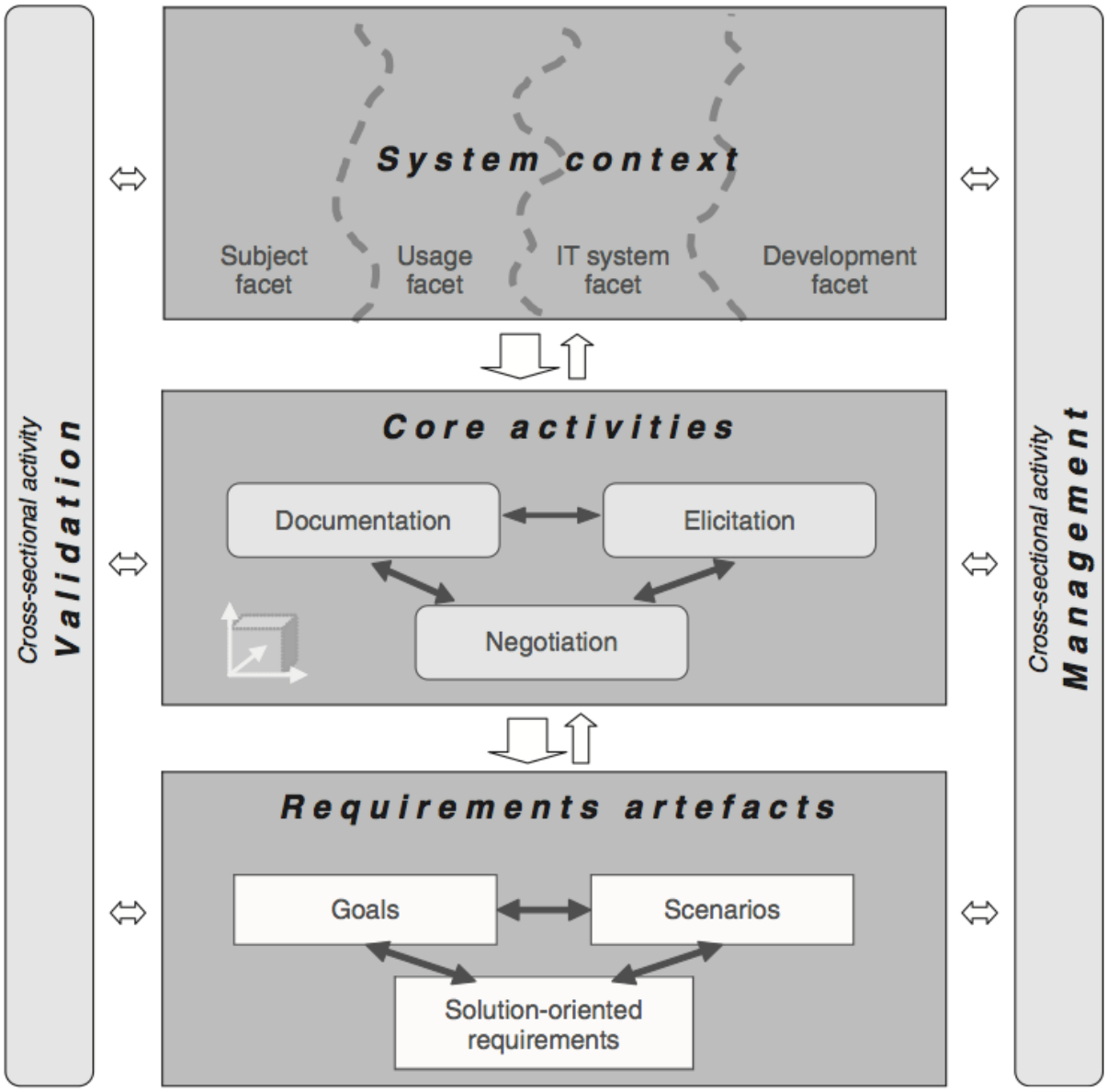


Figure 2.1: The Requirements Engineering Framework. Reprinted from: (Poh10)

- **Two Cross-sectional Activities:** (1)Validation and (2)Management support the core activities. (See section "Management" and "Validation" sections, starting in P. 24)

Is important to notice from the arrows in the Framework (figure 2.1) that this is an iterative process, just like in practice. For instance, elicitation techniques (e.g. interviews) may be used in the beginning and in the end for validation of the requirements. In the following pages of our study, we will dig into the details of these main building blocks listed above. Starting from the System Context, followed by Requirements Artifacts, then Core Activities, and finalizing with the Two Cross-Sectional Activities (Management and Validation). We have followed the sequence of the explanation of the framework given by the author Klaus Pohl in his book.

System Context

The requirements in every software system are heavily influenced by the system context (see figure 2.2). Consequently the requirements for these systems may not be defined without considering the environment where the system is embedded. These include material or immaterial objects such as the following: technical or non-technical systems, people, technologies, business processes and work-flows, laws, already existing hardware and software components, other systems that will interact with the new system, physical laws, safety standards, system users, etc. Therefore, the understanding of the context is a crucial preparation in order to develop a successful requirements specification. The framework is structured in the following four different context facets which must be considered as the prerequisite phase on the Requirements Engineering process:

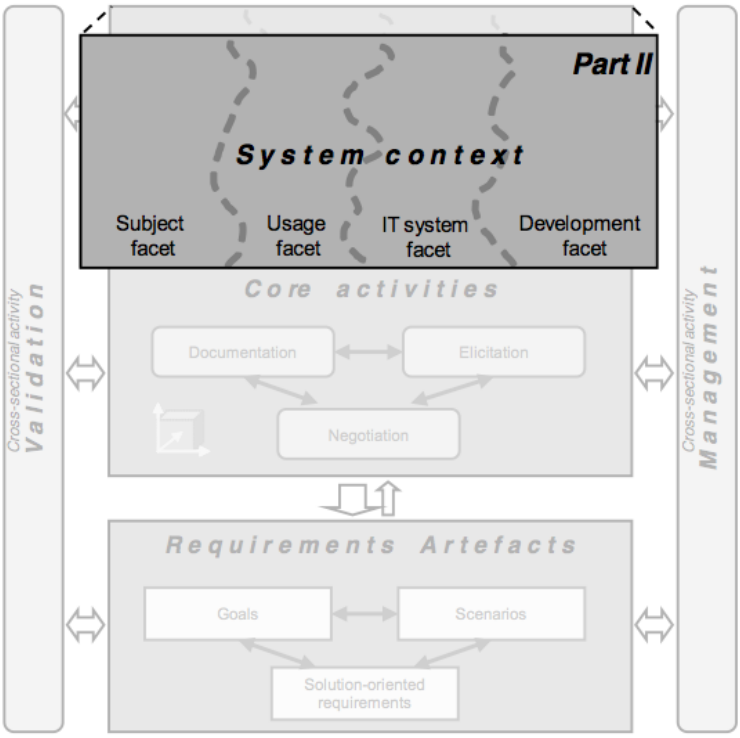


Figure 2.2: Four Context Facets. Reprinted from: (Poh10)

1. **Subject Facet**

   This facet is composed mainly of the objects and the events that must be represented in the system; given that the system must process or store information about these. Additionally, this facet also includes aspects that constraint the representation of the system such as data privacy laws disallowing the storage of a certain type of data or accuracy constraints.

2. **Usage Facet**

   This facet which refers to all aspects concerning the usage of the system by people or other systems such as: user groups with specific characteristics, usage goals, desired usage work-flows, or modes to interact with the system through the system's interface.

3. **IT Facet**

   This facet encompasses all aspects related to the operational or technical environment in which the system is deployed. IT facet includes hardware and software components (e.g. software platforms, networks, existing components), IT strategies and policies.

4. **Development Facet**

   Last but not least, the development facet encompasses all the aspects which influence the system's development process, for instance: process guidelines, development tools, quality assurance methods, and other techniques to assure quality in the system. This may always be restricted by law, for instance if there are only certain tools the client requested to be used during the systems' development.

In order to facilitate a more systematic consideration of the system and achieve requirements specification with higher quality, the author Klaus Pohl suggests creating a simple checklist for each context facet and its relationship with the other facets.

Requirement Artifacts

The term "requirements artifacts" stands for a documented requirement using a specific format. In this section we differentiate between three different requirements artifacts according to Klaus Pohl, namely goals, scenarios and solution-oriented requirements (Poh10):

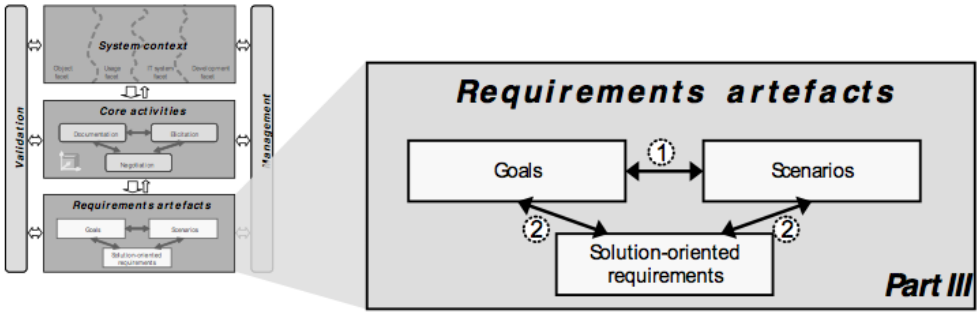


Figure 2.3: Requirements Artifacts Reprinted from: (Poh10)

1. **Goals:**

   In Requirements Engineering domain, the stakeholder's intentions about the objectives are documented as goals stating what exactly is expected or required from the system.

   Goal Decomposition is used to refine the system vision into objectives to be fulfilled by the system. There are two types: AND decomposition is utilized when all sub-goals must be satisfied the super-goal, while OR decomposition only one of the sub-goals is needed to satisfy the super-goal. There are different goal models which main aim is to understand the goals and to complement the template-based documentation. Some of the common models for documenting the need of the different stakeholders are: AND/OR Trees and AND/OR Graphs, goal-oriented modelling framework i* (i-Star) and KAOS framework shown below:

   - **AND/OR Trees and AND/OR Graphs:** This type of model presents the goal decomposition using nodes (see figure 2.4). Each node consists of one super goal while the graphical notation indicates the type of decomposition that it may either be an AND or OR.

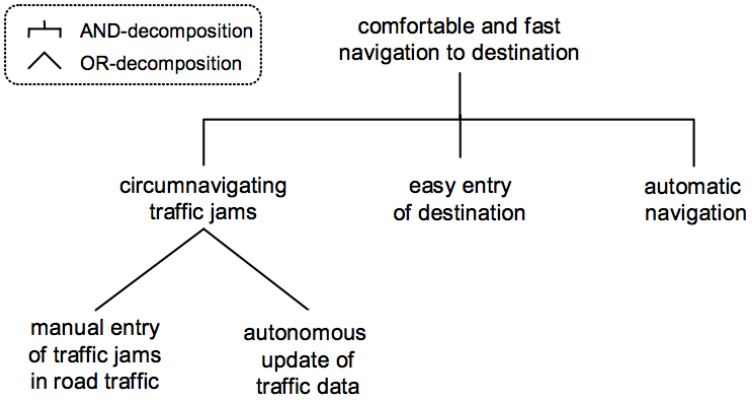


Figure 2.4: Example of goal modelling using AND/OR trees. Reprinted from: (Poh10)

   - **i* (i-Star):** The i* modeling is based in the Goal-Oriented Requirements language and applies AND/OR trees in order to document the decomposition of the goals. The notation of the i* modelling is divided into objects, dependencies and links (see notation and example in figure 2.5).

   - **KAOS - Modelling Language**

     The KAOS (Knowledge Acquisition in automated Systems) modelling language is utilized nowadays for eliciting and also analyzing goals, requirements, and scenarios. KAOS modelling is composed of objects and relationships (see notation and example on the figure 2.6).

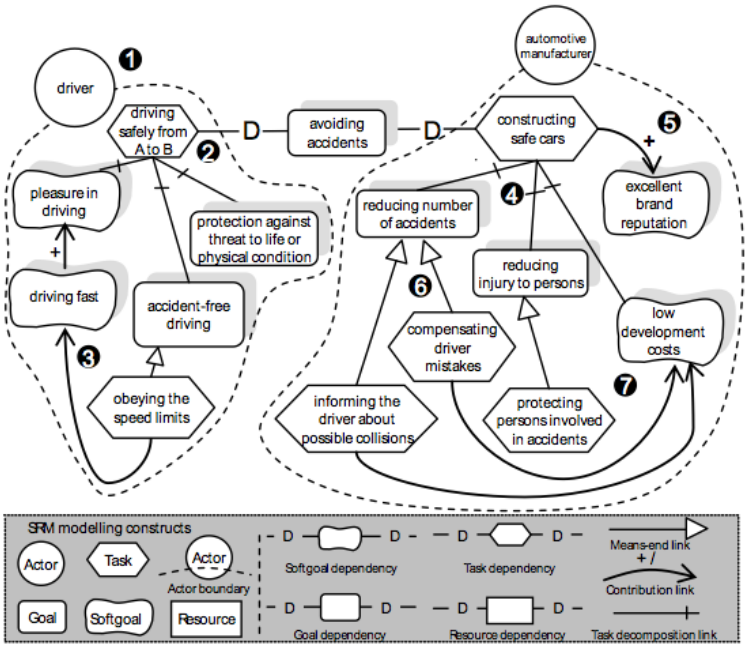


Figure 2.5: Example of a strategic rationale model in i*. Reprinted from: (Poh10)

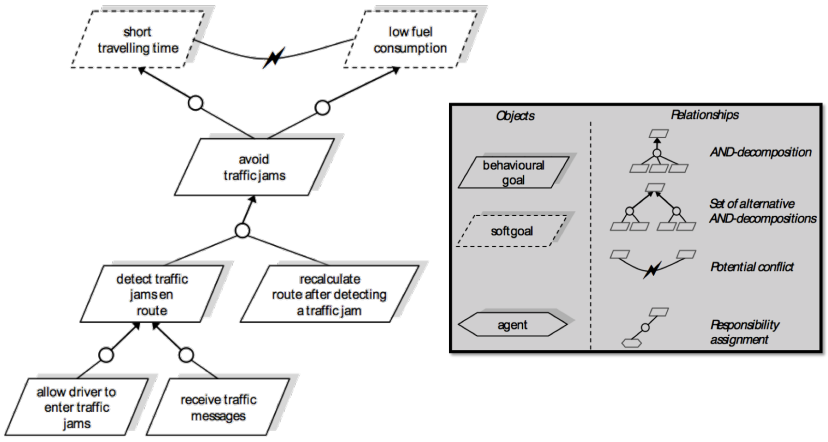


Figure 2.6: Example of a goal model in KAOS. Reprinted from: (Poh10)

2. **Scenarios:**

   As seen in the previous section, goals are used to document the stakeholder's intentions. Therefore, scenarios are further used to document intentions by illustrating if the scenario satisfies or fails to satisfy the goal by representing the interactions between the system and its actors.

   - Scenario Types:

     Different scenarios may be classified according to its criteria, purpose, contribution to goal's satisfaction or its abstraction level. Some of the different scenario types include the following:

     a) Use Cases - Grouping Scenarios: Use cases are a sequence of actions that include variant and error sequences of the system. A use case contains: Context information (stakeholder's goals), Main scenario (sequence to satisfying the goal), Alternative scenarios and Exception scenarios.

     b) Current State and Desired State Scenarios

     c) Positive and Negative Scenarios: Positive referring to document the desired sequence of interactions leading to the satisfaction of a goal while negative leads to failure.

     d) Misuse Scenarios: Scenario where the hostile actor uses the system against the stakeholder's intention.

     e) Descriptive, Exploratory and Explanatory Scenarios

     f) Main Scenario, Alternative Scenarios, and Exception Scenarios

   - Scenario Documentation:

     There are different techniques utilized for documenting the scenarios and use cases using natural language and models. Some of them are described below:

     a) Narrative Scenarios: Documented in natural language

     b) A Reference Template for Use Cases: reference template for documentation using attributes in columns (ID, name, author, version, change history, goals, actors, precondition, post-condition, etc.). See an example of a template for the use case specification on the figure 2.7:

     c) Sequence Diagrams: Using models like sequence diagrams has several advantages over natural language. A sequence diagram is an interaction (message exchange) between the system and a set of actors over time. Each role consists of a lifeline indicating when this role is active. (See Left Sequence Diagram: 2.8).

     d) Activity Diagrams: UML activity diagrams allow emphasis of the control flow between multiple scenarios. The main focus on activity diagrams is the control flow (actor's activities and the possible order of the activities) (see Right Activity Diagram: 2.8).

USE CASE <Use case code>	<Use case name>	Date:	<date of the last change >
		Version:	<use case version>
Description:	<use case brief description>		
User priority:	<development priority requested set by users>		
Performance:	<requested performance>		
Primary actor:	<primary actor name> <primary actor interest in this service>		
Secondary actor:	<secondary actor name> < secondary actor interest in this service>		
Preconditions:	<description of the conditions which must be fulfilled before the use case can be executed>		
Post-condition	on success:	<description of the conditions that describe the state of a system after that this use case is successfully completed>	
	on failure:	<description of the conditions that describe the state of a system after that this use case is unsuccessfully completed>	
Trigger:	<description of the event that fires the use case>		
MAIN SCENARIO			
1.	<actor/ system>	<action description>	
2.	<actor/ system>	<action description>	
…	<actor/ system>	<action description>	
n.	System:	Terminates use case successfully.	
ALTERNATIVE SCENARIO: <condition which leads to this alternative scenario>			
h.1.	<actor/ system>	<action description>	
		<action description>	
h.n.	System:	Resumes use case at point x.y.	
EXCEPTION SCENARIO: <condition which leads to this exception scenario>			
i.1.	<actor/ system>	<action description>	
		<action description>	
i.m.	System:	Terminates use case unsuccessfully.	
ANNOTATION			
	<extra information>		

Figure 2.7: An example of a template for the use case specification. Reprinted from: (Tag03)

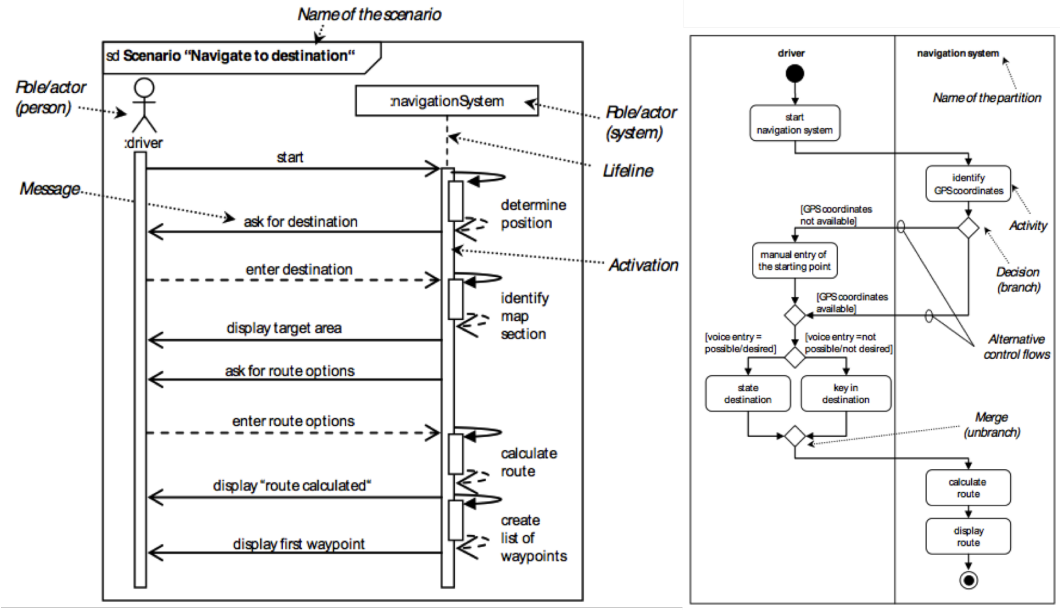


Figure 2.8: Documentation of a scenario in a sequence diagram (Left). Documentation of the control flow of scenarios (Right). Reprinted From: (Poh10)

3. **Solution-Oriented Requirements:**

Differences between Solution-Oriented Requirements, Goals and Scenarios

As seen in the previous two sections, goals and scenarios are the basic foundations for developing solution-oriented requirements. In contrast to goals and scenarios, solution-oriented requirements are created to specify a deeper required level of details (see figure 2.9). By joining these three together requirements, we may define the reference software engineers to implement the system.

The key differences from solution-oriented requirements, goals, and scenarios are summarized in the following bullets:

- *Agreement:* Solution-oriented requirements must be agreed by all stakeholders.
- *Completeness:* Solution-oriented requirements must be as complete as possible.
- *Conflicts:* Goals have conflicts. Solution-oriented should be free of conflicts.
- *Level of Detail:* Solution-oriented requirements should be at a level of detail far more details than goals and scenarios. (See figure 2.9)

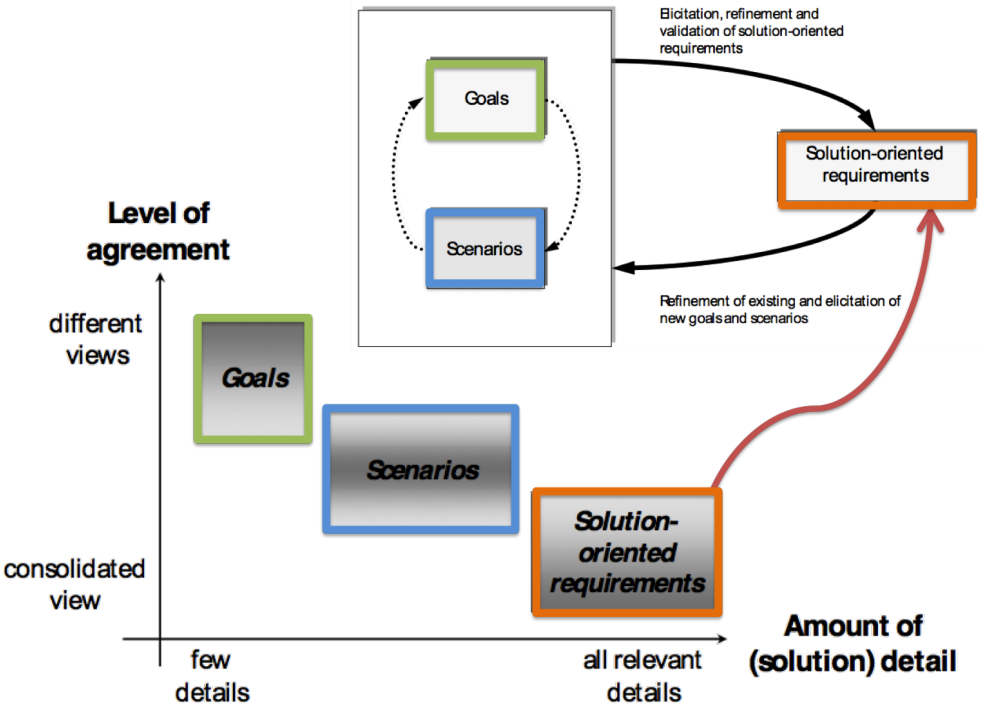


Figure 2.9: Coarse-grained characterisation of artifacts and their relationship. Adapted from (Poh10)

Three Perspectives on a Solution

According to the author, conceptual modeling languages have been developed to support the documentation of solution-oriented requirements. There are three main traditional requirements perspectives for defining the solution-oriented requirements (see the figure 2.10):

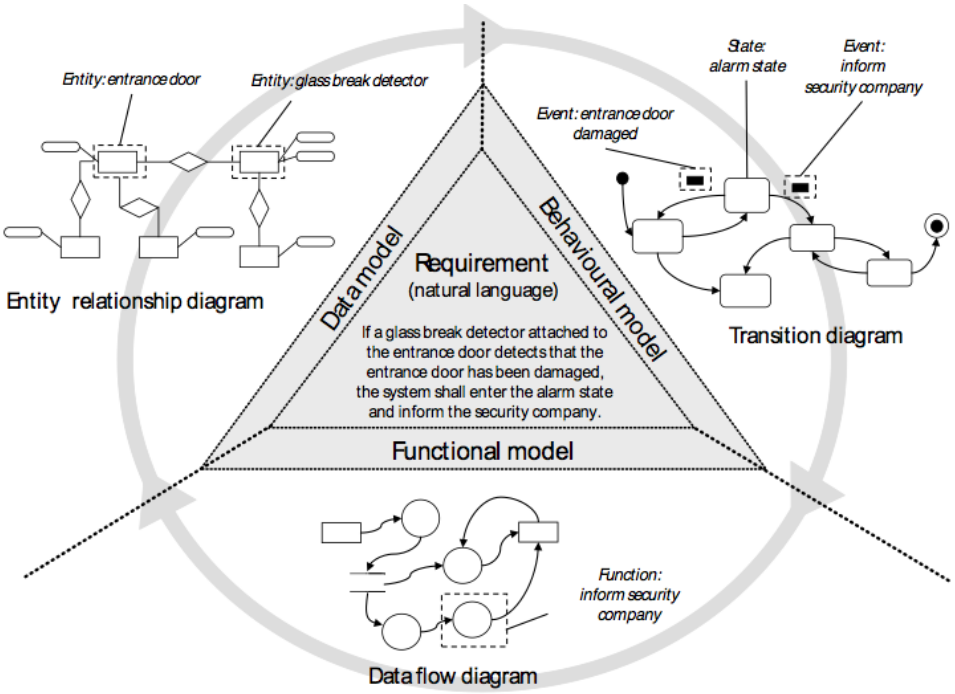


Figure 2.10: Model-based documentation of the three perspectives. Reprinted from: (Poh10)

- **Data Perspective:** Defines data to be managed by the software-intensive system. Entity-relationship models and class diagrams are used to document requirements in data perspective. In addition, UML class diagrams are mainly used to document the static view of the system.
- **Functional Perspective:** These are document requirements for the system in terms of functions. Data flow diagrams are used to documents the functional perspective of the system.
- **Behavioural Perspective:** Defines the overall behavior of the system. Commonly automata, state-charts, and state-machine are used for defining behavior.

Core Activities

This block is composed of the three main core activities Documentation, Elicitation and Negotiation. These are described below:

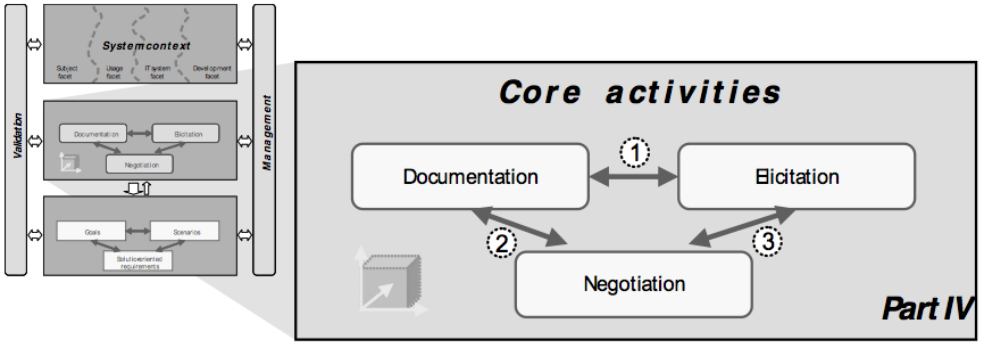


Figure 2.11: Requirements Engineering Core Activities. Reprinted from: (Poh10)

1. **Documentation**

   The main aim of this activity is to document the elicited requirements according to the rules. According to Klaus Pohl, documentation is important in order to establish a common reference, promote communication between stakeholders, support training for new employees and to preserve expert knowledge. Rules for documentation may be either: 1-General Rules (e.g. define layout, headers, version history), 2-Documentation Rules (e.g. rules to ensure quality of template for the use of elicitation and negotiation), or 3-Specification Rules (more restrictive than documentation rules e.g. rules for subsequent development activities like the use of requirements specification language). In order to avoid lexical ambiguity, the author suggests glossaries may be used along with synonyms and related terms.

2. **Elicitation**

   The goal of this activity is to elicit the requirements at all the levels of the system. These requirements may exist in different forms, for instance: in ideas, intentions, requirements models, in existing systems, etc. In addition, there are many sources (stakeholders, documents, existing systems) that have to be elicited in order to gather all the possible requirements to support the acceptance of the system. The three activities to identify relevant sources are the following:

   a) Identify relevant requirement sources using two-way procedure (see figure 2.12)

   b) Elicit existing requirements from the identified sources

   c) Develop new and innovative requirements

   Requirements may be elicited from the sources by using different techniques:

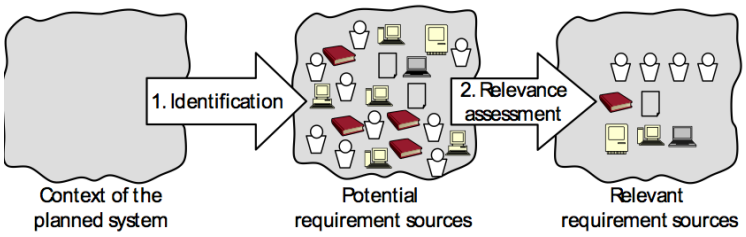


Figure 2.12: Identification of Relevant Requirement Sources (Two-Step Procedure). Reprinted from: (Poh10)

- Elicitation Technique from stakeholders: Interviews, conversations, questionnaires and observation.
- Elicitation Technique from documents: Perspective-based reading
- Elicitation Technique from existing systems: Using/observing the system, interview with system's stakeholders and analyzing the system's documentation.
- Assistance Techniques: Prototyping, brainstorming, KJ model and Mind mapping.

3. **Negotiation**

   Each stakeholder has different wishes and needs than may be in conflict with each other; reason why negotiation aims to achieve an agreement among them. Conflicts must be resolved involving the most relevant stakeholders while also these may be seen as an opportunity to create also new ideas. Negotiation Techniques: Win-Win and Interaction Matrices (requirement overlapping). Conflict management involves four steps: identify, analyze, resolve and document conflicts. According to Klaus Pohl, there are four conflict types:

   - Data conflict: Stakeholders are wrongly or incompletely informed or misunderstood.
   - Interest conflict: different interests of stakeholders contradict
   - Value conflict: Different stakeholders evaluate the requirement differently

Inter-dependencies Between the Three Core Activities (see figure 2.11):

1. Documentation + Elicitation: Elicitation activity must document its output according to the rules defined on the project. Missing information in the documentation may be covered with elicitation activity.
2. Documentation + Negotiation: During the documentation activity, conflicts may be detected and then these can be resolved using negotiation techniques, further this must be documented.
3. Elicitation + Negotiation: During the elicitation, conflicts may arise. Therefore, this must be resolved using negotiation techniques.

Considering that all the core activities are interrelated, the implementation or development of one of these may directly influence the others; for instance, by eliciting a new requirement, requires to create more documentation.

Validation

Validation aims to state if the input and output (requirements artifacts) of the core activities satisfy the criteria established. This phase of validation is done by involving different resources, such as: stakeholders, requirement sources (laws and standards) and external reviewers. Validation answers the question: Am I building the right system? (see figure 2.13). The three main goals of validation are:

Validation Goal #1: check whether the outputs of activities are in accordance with the defined quality criteria. This is done by checking the requirements artifacts for incompleteness, ambiguity and contradictions between them.

Validation Goal #2: check whether the inputs of activities are in accordance with the defined quality criteria. This is done by checking the context aspects, if these are overlooked, or documents contain errors, then this will lead to having errors in the artifacts.

Validation Goal #3: checking whether the execution of activities is in accordance with the process definitions. For instance, Requirements Engineering may violate the guidelines by not using the checklist previously defined for techniques.

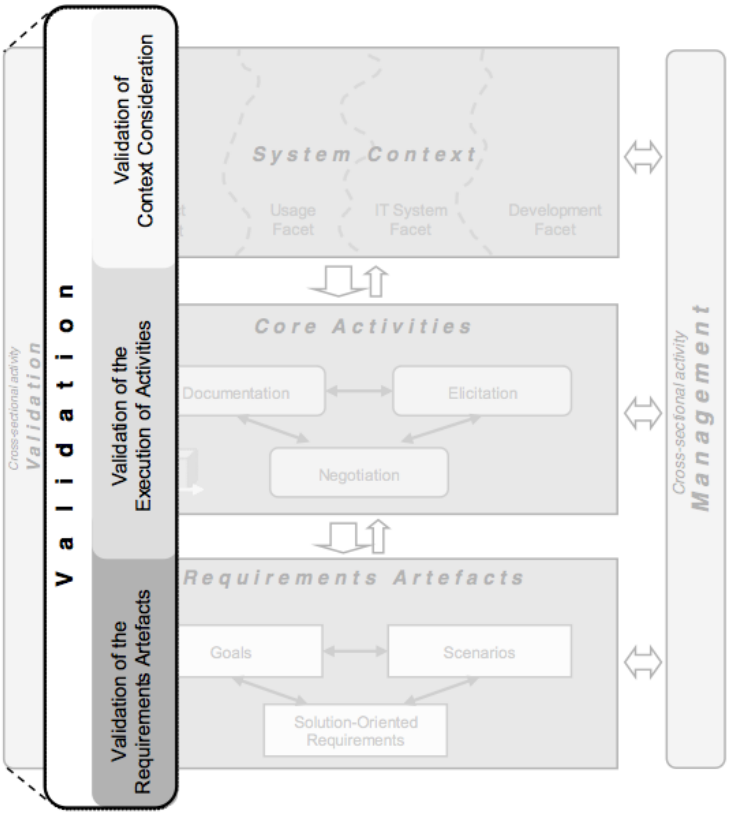


Figure 2.13: Cross-sectional activity: validation (left-hand side). Reprinted from: (Poh10)

Some of the risks caused by insufficient validation are:

- Error Propagation: If an error in a requirements artifacts is not detected until the release of the system, then all artifacts have to be again revised.
- Legal Issues: Requirements artifacts are part of the contract between contractor and client, therefore if these are insufficiently validated this may lead to legal disputes.
- Reduction of costs and risks: Even if the requirement's validation activity takes much time, it can reduce future costs and risks.

Requirements validation is elaborated in the following sub-activities:

- **Validating the Created Requirements Artifacts:** Each requirement artifact must be validated about the three validation aspects in order to define its acceptance or rejection:
  - Validation about the content dimension: Verify if all relevant requirements are known and understood to the required level of detail while also checking the requirements for deficiencies.
  - Validation about the documentation dimension: Verify if the requirements are documented according to the specification rules. Rules for documentation may be either: 1-General Rules, 2-Documentation Rules or 3-Specification Rules (more restrictive).
  - Validation about the agreement dimension: Verify if the stakeholders reached an agreement on the requirements that are documented.
- **Validation in Goals and Scenarios:**
  - **Incomplete documentation of goals:** Incomplete documentation of goals may be caused due to not sufficient communication among stakeholders.
  - **Supporting goal validation with scenarios:** Stakeholders cannot decide to approve or reject a documented goal until they have provided an example of its realization and see if it fails.
  - **Scenarios combined with prototypes:** Scenarios combined with the use of prototypes in practice. This will enable the stakeholders to explore the scenarios interactively and detect possible changes. (See figure 2.14)
- **Validation Techniques:** There are four validation techniques used to validate the requirements artifacts.
  - Inspections
  - Desk-Checks
  - Walkthroughs
  - Prototypes (see figure 2.14)

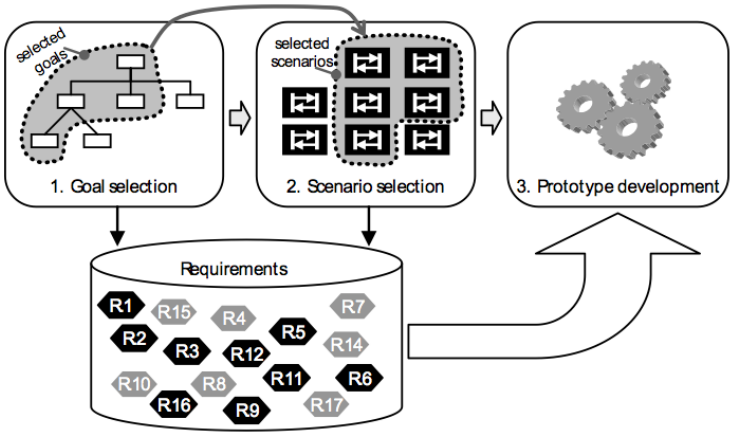


Figure 2.14: Goal and scenario-based selection of requirements for prototype development). Reprinted from: (Poh10)

- **Assistance Techniques:** There are several assistance techniques used to support the validation of the requirements.
  - Validation Checklist
  - Perspective-Based Reading
  - Creation of Artifacts

Management

Apart from Validation, Management is another one of the cross-sectional activities in the Requirements Engineering framework. The following three bullets explain the three sub-activities in the management phase related to the middle framework blocks (see figure 2.15):

1. **Observing the System Context:** This sub-activity is in charge to identify the changes in the context (sources or objects) facets and estimate the impact that these changes may have. For instance: new technology, new competing product emerging, law changes, new stakeholders, etc.
2. **Managing Requirements Engineering Activities:** This sub-activity ensures the fulfilment of the goals of Requirements Engineering. There are two basic approaches for managing the Requirements Engineering activities: phase-oriented approach or situative approach.
3. **Managing Requirements Artifacts:** The aim of the management activity is to keep track of the requirements artifacts and its relevant attributes, relationships and evolution. Traceability and Prioritization being the most important examples.

In the following description, we will focus on the examples provided in bullet #3 (Managing Requirements Artifacts).

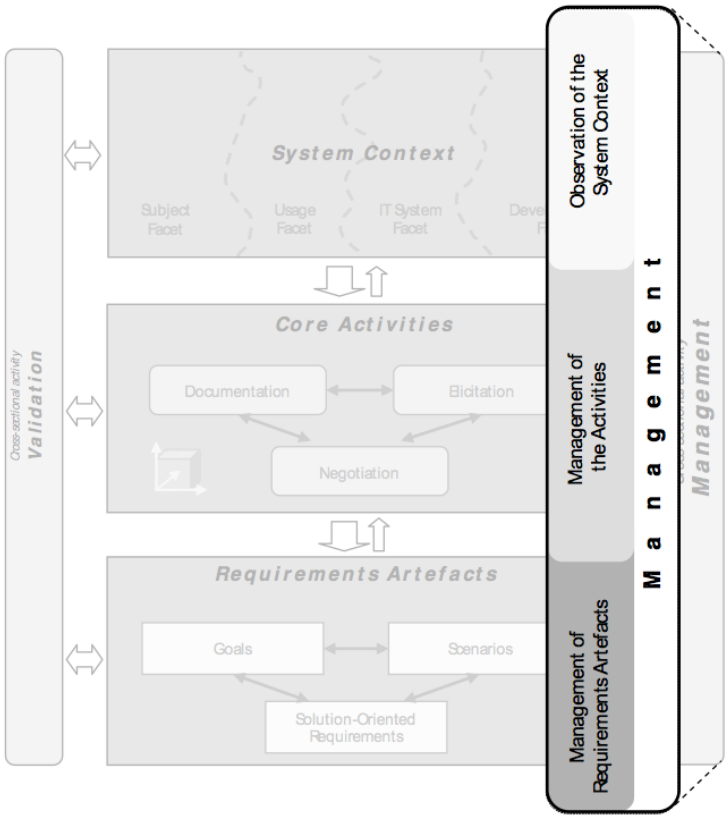


Figure 2.15: Requirements Engineering Management. Reprinted from: (Poh10)

Requirements Traceability

First, Requirements Traceability are meant to describe and follow the life of the requirement forward and backward though-out all periods (see figure 2.16) .

- Pre-traceability ensures the traceability to its source or its origin.
- Post-traceability ensures the traceability from the requirement to implementation of artifact.

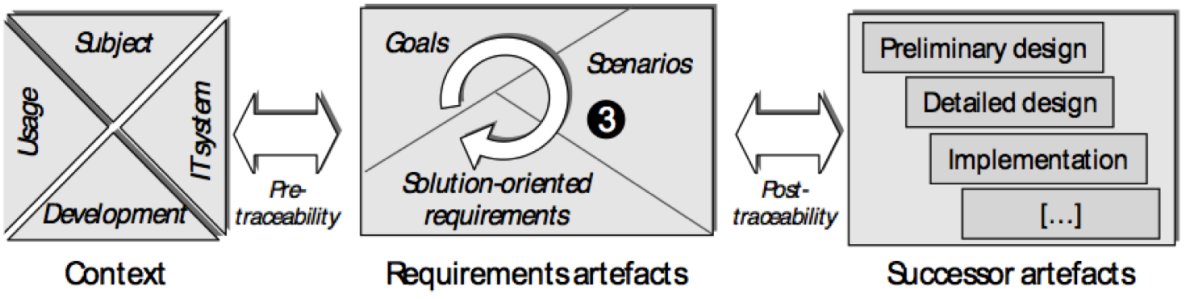


Figure 2.16: Requirements Engineering Management. Adapted from: (Poh10)

Presentation of Traceability Information

There are different ways to present or visualize the documents that demonstrate the traceability of the requirements:

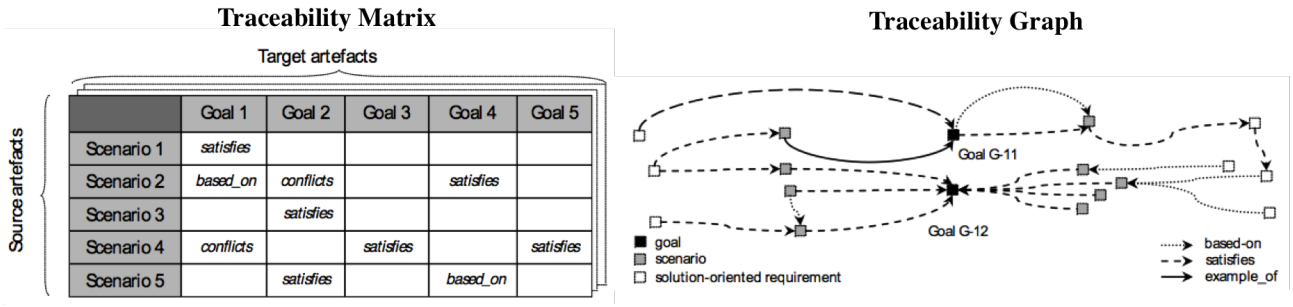


Figure 2.17: Left: Traceability matrix for several relationship types. Right: Representation of traceability information by means of traceability graphs. Reprinted from:(Poh10)

- Traceability Matrices: these are used to present the visualization form of the information between the artifacts and predecessor or successor of the artifacts. Rows represent the source artifacts and columns represent the target artifacts. (See figure left: 2.17)
- Traceability Graphs: graphs utilize different types of nodes in order to represent the artifacts (goal, scenario, or solution-oriented requirement) and different types of edges represent the traceability (basedOn, satisfies, exampleOf). (See figure right: 2.17)

Prioritizing Requirements

In practice due to resource limitation, not all requirements may be considered during the development of the system, neither in the same degree. Therefore, these limited resources must be used the best way possible by using a priority criteria within the requirements by selecting the requirements that will first be implemented. The simplest way to create requirements prioritisation is according to a single criterion (importance, cost, damage, duration, risk or volatility). Prioritization techniques may be:

- Ranking
- Top 10
- One-Criterion Classification
- Kano Classification
- Winger's Prioritisation Matrix
- Cost-Value Approach

2.2 PART #2: Business Intelligence & Decision Support Systems Context

In Part #1 (Requirements Engineering Context) of this Chapter, we had explained the importance of the field in nowadays projects and described the Requirements Engineering framework. Given that Pohl's framework is not tied to a specific kind of software project, instead is more generic and adaptable, in this Part #2 we provide the context of the systems we will use for our study, namely Decision Support Systems. We will in later chapters map the techniques and practices of Decision Support Systems to the Requirements Engineering framework. Correspondingly, we explain Decision Support Systems starting from a higher-level concept, Business Intelligence, continuing with a comparison among Decision Support Systems and traditional Operational Systems and finalizing with one of its most typical architectures, Data Warehousing.

Business Intelligence

Business intelligence aims to understand what makes the motor of an enterprise help to predict future trends on current decisions; reason Business Intelligence plays a key role in the decision-making process of nowadays enterprises (IIS02). Even if we hear about the term "Business Intelligence" very often nowadays, the literature points out that this term origins back in 1958. One of the information systems pioneers, Hans Peter Luhn, defined the term as *"The ability to apprehend the interrelationships of presented facts in such a way as to guide action towards a desired goal"* (Luh58). Back then in the 1950's, people in the companies had the not much choice than to analyze and forecast future trends based on their experience, however, as the time passed this scenario completely changed when the world became what we now call a computerized world.' According to the work from (DBL08), unlike the current context of Business Intelligence (e.g. Data Warehousing and database systems) which its main focus is on enterprise transactional data (e.g. sales), Luhn's definition in 1958 focused mostly on the information contained in business-related documents. Yet, this work also claims that despite the definition was proposed more than 50 years ago, nowadays it remains with the same essence: *improving the business decision-making process*; the difference is that the information instead of being a pile of documents, nowadays this has turned to be computerized' enterprise data including different types of information and formats being integrated to be further analyzed for the decision-making process.

More recent definitions of Business Intelligence include different perspectives. For instance, one of the business oriented definitions provided by Lonnqvist defines the term as: *"An organized and systematic process by which organizations acquire, analyze, and disseminate information from both internal and external information sources significant for their business activities and for decision-making"* (SS14). On the other hand, Business Intelligence also has the technical definition: *"aimed at gathering, transforming and summarizing available data from existing sources to generate analytical information suitable for decision-making tasks"* (RA13).

As we read these two perspectives above, we can notice that Business Intelligence may be defined from various points of views, which do not compulsorily have to be derived from the technology perspective referring to information systems, but may also be defined from a business perspective. In consequence, the two perspectives merge to create Decision Support Systems: *"A set of expansible, interactive IT techniques and tools designed for processing and analyzing data and for supporting managers"* (GR09). From an IT perspective, Decision Support Systems retrieve useful information in huge amounts of data stored in heterogeneous sources; whereas from the business perspective, decision-makers can formulate their information

needs and execute complex analyses on the information without slowing down operational systems (GR09). The following subsection defines the main characteristics of Decision Support Systems and how these differ from traditional Operational Systems.

Operational Processing and Decisions Support Processing

In general there are two categories of tasks in all organizations. First ones dealing with daily operational tasks (Operational Systems), while the second ones are utilized for making business decisions. The latter refers to a computer-based system that bring together information from heterogeneous sources for analysis purposes, whose most usual center of existence is the Data Warehouse (IIS02). According to one of the most well-known definitions by (Inm96), Data Warehouse is defined as read-only (non-volatile) repository of data which represents a single and integrated view of the organization aimed to be exploited by its users using a set of exploration tools (more information regarding Data Warehouse and its architecture may be found on the following subsection).

According to the literature, there are several reasons why the information system architects need to separate Operational processing from Decision Support Systems/Data Warehouse data and processing (e.g. performance). (IIS02) clearly shows the delimitation between these two in figure 2.18. According to the authors, the application environment is settled in the world of the operational processing, while the Data Marts and the Data Warehouse are in the world of Decision Support Systems processing; having the ODS (Operational Data Store) standing in the middle of both.

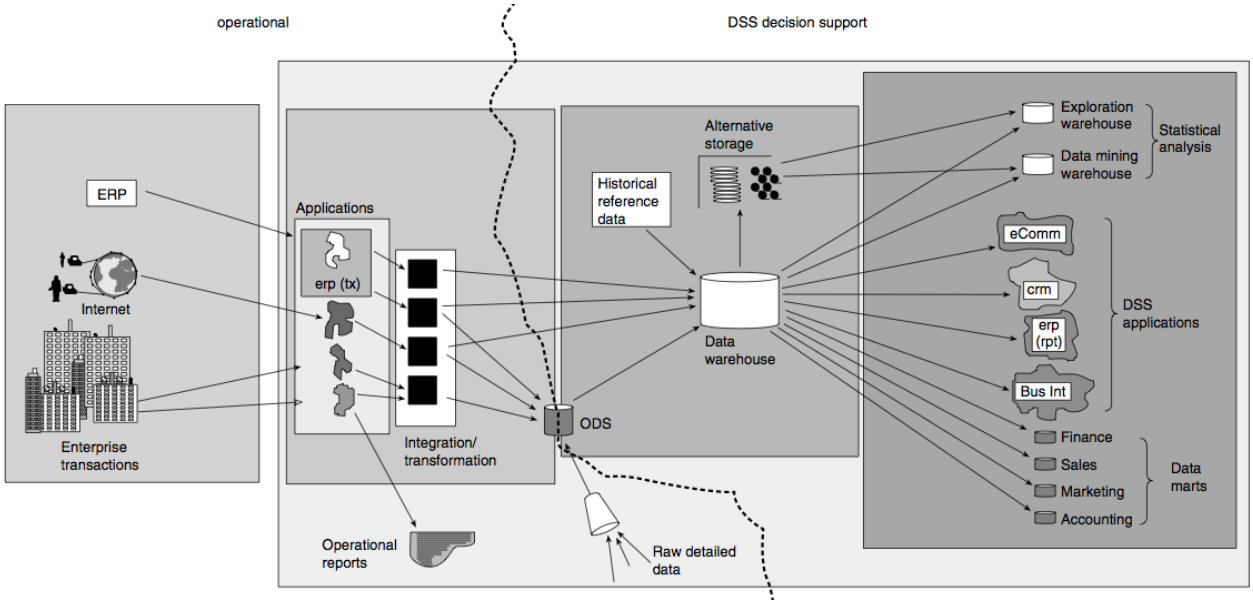


Figure 2.18: A very clearcut dividing line exists between operational processing and Decision Support Systems informational processing. Reprinted from: (IIS02)

If we look at the figure 2.18 in more detail, the flow of the data starts by entering the enterprise in various forms, such as raw data collected by the applications. The raw detailed data is then refined by the applications to be passed into a layer of programs that integrate and transform into the ODS (Operational Data Store) and then is passed into the Data Warehouse

(IIS02). The Data Warehouse may be fed by the ODS or by the integration/transformation layer. Finally, the authors clearly illustrate in the figure that after the data has passed through the Data Warehouse to be accessed, analyzed visualized.

Accordingly, Operational Systems and Decision Support Systems tasks have different functions in nowadays organizations. Let us provide with an example related to the Chagas project to illustrate the difference:

For example:

- An example is a day-to-day task decision, such as a WHO Researcher uses the system to consult whether La Rioja has been inspected (Yes/No) to either send or not the Inspection team.
- Another example is the entomologists of the WHO uses a system that shows how Chagas activities have changed from this current year compared to the last 4 years. This is done by accessing plenty different sources (e.g. excel with inspected dwellings, PDF's with regulations, databases from hospitals) and correlating how Chagas has spread over the time using visualizations on maps. All this to decide whether or not to open a new hospital in La Rioja. This is known as a Decision Support System.

As we can see from the summarizing Table 2.1, there are various differences between Operational and Decision Support Systems. These differences are well-presented and described by (Cip97) in the following bullets:

Comparison		
	Operational	Decisional
Objective	Business operation	Business analysis
Main functions	Daily oper. (OLTP)	DSS (OLAP)
Usage	Repetitive (predefined)	Innovative (unexpected)
Design orientation	Functionality	Subject
Kind of users	Clerks	Executives
Number of users	Thousands	Hundreds
Accessed tuples	Hundreds	Thousands
Data sources	Isolated	Integrated
Granularity	Atomic	Summarized
Time coverage	Current	Historical
Access	Read/Write	Read-only
Work units	Simple transactions	Complex queries
Requirements	Performance & consistency	Performance & precision
Size	Mega/Gigabytes	Giga/Tera/Petabytes

Table 2.1: A comparison between Operational and Decisional Systems. Reprinted from: (Rom11)

- Decision Support Systems are usually based upon the Data Warehouse while it represent a multi-dimensional architecture of a data model instead of a relational model. Whereas operational is focused on clerical data processing tasks being short, structured and isolated transactions (e.g. banking transactions).
- Decision support usually requires historical and external data that go further from the operational databases in order to make analysis and predictions as seen in table 2.1, in contrast to operational databases that store only current data.
- In addition to the historical and other loaded data integrated from diverse operational databases, data warehouses include pre-computerized data aggregations derived from these, tending to have a large magnitude (Giga/Tera/Peta) than the one used in the operational databases (Mega/Giga).
- The workload utilized in Data Warehouses using OLAP (On-line Analytical Processing) are mostly complex queries implemented to access millions of records with scans, joins and aggregates, see table 2.1. Some of the OLAP operations include: roll-up and drill-down, slice and dice, and pivot (more information regarding OLAP may be found in the following subsections).

Overall, Operational systems focus on daily needs while Decision systems focus on providing a single detailed view for decision making based on information. For this reason, we have centered our study into the systems capable of providing help for business managers to make better decisions; these are Decision Support Systems. As explained before, Decision Support Systems most of the time load data from different sources into the system's core (Data Warehouse) and then different tools are used to exploit the data to obtain relevant knowledge for the business; the formal definition of this concept is Data Warehousing. Even though Data Warehousing is not the only Decision Support System, it is the most typical one (GR09). More details regarding the Data Warehousing (architecture) are explained in the following subsection.

Data Warehousing Architecture

As stated in the book of *"Data warehouse design: Modern principles and methodologies"* by (GR09), above all, the Decision Support Systems, Data Warehousing is apparently the well-known architecture that the academic community and the industry have recognized. According to the book, Data Warehousing is formally described as: *"a collection of methods, techniques, and tools used to support knowledge workers to conduct data analyses that help with performing decision-making processes and improving information resources."* The book describes the three main components of the architecture, namely:

- **ETL (Extraction, Transformation and Loading)**: As seen in figure 2.19, Data Warehousing uses heterogeneous sources of data including external sources that may come from Operational databases and sources from outside the enterprise infrastructure. During the ETL, sources are extracted, cleansed, filtered, and ready to merge with a common schema to finally be loaded into the core: the Data Warehouse.
- **Data Warehouse**: The Data Warehouses are the heart of this process its role is to act as data repositories. The Data Warehouse is a read-only database that may be accessed directly, but as seen in figure 2.19, it may also act as a source for the Data Marts (subsets or aggregation of Data Warehouse).

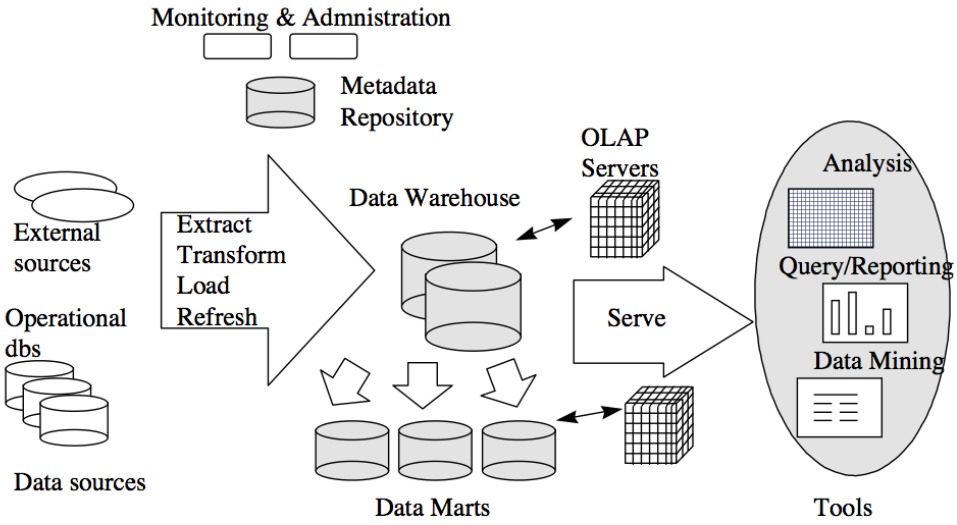


Figure 2.19: Data Warehousing Architecture. Reprinted from (CD97)

- **Exploitation Tools:** After the data is integrated, the data is accessed to provide the business users with reports and simulate different business scenarios (e.g. adding data navigators, OLAP (read the following subsection), visualizations for analysis, reports embedded in dashboards, etc.).

OLAP and Multidimensional Model

Given the description of the Data Warehousing in the previous section and its exploitation tools, its been said that OLAP is the most popular way to exploit the information in the Data Warehouse by giving the users the opportunity to analyze and explore the data in an interactive manner based on a multidimensional model (GR09). OLAP was introduced by E.Codd during the 90's (CCS93) and the term was called to contrast with OLTP (On-line Transaction Processing). E.Codd was famous for defining twelve rules for analytic systems, however, in 1995 Nigel Pendse stated in *"The OLAP Report: Succeeding with On-line Analytical Processing"* that these rules were an unsuitable way for detecting 'OLAP compliance' (PCF95). Before long, Pendse decided to propose in this report the FASMI test (standing for: Fast Analysis of Shared Multidimensional Information). Decisional Systems such as well-designed Data Warehouses aim for fast answers, easy-to-use, and efficient analysis. Therefore, the popular FASMI test described in the report by Pendse aims to determine whether a Data Warehouse has achieved this goal with reference to the characteristics described below (PCF95):

1. **Fast Analysis:** Fast query answering (less than 5 seconds) supporting flexibility of calculation so this may be quickly defined.
2. **Shared:** Supports the sharing and the accessibility on a secure manner.
3. **Multidimensional:** The data model must be Multidimensional.
4. **Information:** Being the most important feature, they must support huge amount of data for analysis tasks based on multidimensionality.

After defining OLAP, we may now present the term of multidimensionality. In order to explain what is a Multidimensional model, let us provide some real examples elicited to the WHO specialists:

- **Entomologist:** *I want to display data about infested dwellings per insecticide used and area*
- **WHO:** *I need to visualize maps showing the distribution of triatomines found in intradomicilary AND peridomiciliary structures in 2012 per province in Latin America*
- **Entomologist:** *I want to generate graphics showing the temporal variation of evaluated (inspected) houses by department every two years for La Rioja (Argentina)*

By these examples, is not hard to notice that using traditional languages (e.g. SQL) it will be difficult for a non-technical user to express these queries. This without taking into account the response time may lead to having unsatisfied users. Therefore, multidimensional models start with the *facts* which represent the observation of the factors (e.g. inspected_dwellings). Each of these facts is described by quantitative descriptions called *measures*, (e.g. nbrOfNymphinsects - Number of baby insects). The *dimensions* represent the perspectives of an event, for instance, inspected_dwellings may have the following dimensions: Specie (e.g. Triatoma_pseudomaculata), Country (Argentina) and dates (12/14/2013), see example in figure 2.20. This last concept of *dimension* gives the basis to represent the multidimensional data by means of what is called a 'cube.' According to (GR09), a multidimensional cube (see figure 2.20) stands on a fact which is relevant for the decision-making by showing a set of events with measures: each axis in the cube shows a possible dimension, and each of these dimensions can be analyzed at different levels specified by the attributes defined in the hierarchy (e.g. date, month, year). In summary, 'multidimensionality' is utilized to model an event that consists of facts of interest and descriptive attributes related to metrics.

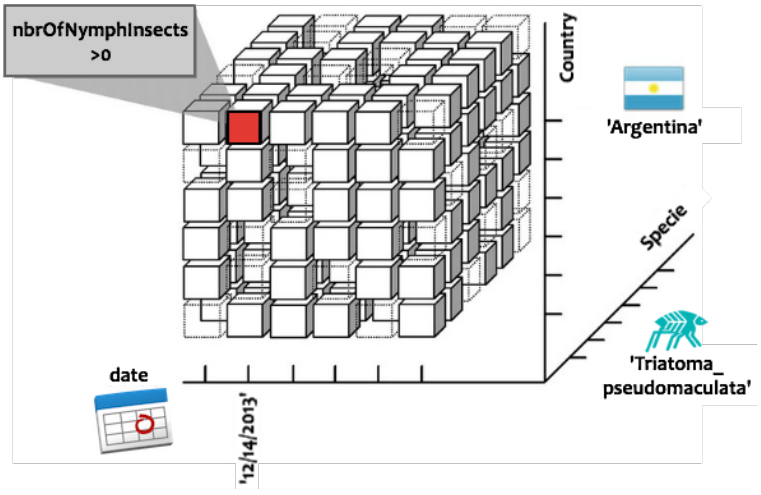


Figure 2.20: Three-dimensional cube modeling inspected_dwellings: Entomologists from WHO want to visualize in maps and compare the inspected dwellings in Argentina where there is at least one nymph insect of the specie Triatoma_pseudomaculata on 14/12/2013. Adapted from (GR09)

In addition, OLAP users must be able to navigate the data in real-time, for instance, making queries and analyzing. According to (GR09), OLAP consist of a *navigation path* provided by the user, where each of the nodes is navigated (derived) from the previous one. Each of the nodes is characterized by an *OLAP operator*; the most common operators stated by the authors are:

- *roll-up:* increase in data aggregation level
- *drill-down:* decrease in data aggregation level
- *slice-and-dice:* reducing the number of dimensions after setting a specific value
- *pivot:* changing the layouts by reorienting the multidimensional view

In conclusion, 'multidimensionality' enables business users (such as managers in the need to take critical decisions) to discover knowledge from their data through fast queries and analytical tasks with the objective to take the best decision.

CHAPTER

"It is a capital mistake to theorize before one has data."
-Sherlock Holmes

3 Current Approaches of Requirements Engineering in Decision Support Systems

According to the previous chapter and our motivation presented in this thesis, the design of Decision Support Systems is rather different from the design of traditional Operational Systems that supply the data to the warehouse. The concern is that leading Information Systems professionals have a lack of an adequate approach to determine information needs for these systems. For this reason, we believe that Requirements Engineering should be specialized for Decision Support Systems systems in order to achieve better results.

To achieve the main goal of this thesis, in this chapter we will begin by engaging with the literature in order to obtain a good picture of the published work concerning Requirements Engineering in Decision Support Systems. Since we aim to gain insight about the area in an organized manner, we have decided to map the found literature to the building-blocks in Pohl's framework presented in the previous chapter to finally enable a complete classification of the literature.

Hence, the first subsection aims to outline the main reasons why we have used a reference framework (by Pohl). This is followed by the scope, which details the selection of the literature (literature search). We will then explain the notation we have used to organize the literature for a better classification. Next, we present the classification tables of the literature mapped to the pre-defined blocks of our Requirements Engineering reference framework. Finally, at the end of this chapter, we provide a summary with the complete overview of the practices mapped into the reference framework and all the elements integrated for easier understanding. This will be later used in the chapter 4 as the base in order to create our own approach of Requirements Engineering for Decision Support Systems and finally implement it on the Chagas Case Study in chapter 5.

3.1 Reference Framework

Previously in chapter 2, we have presented a framework for Requirements Engineering proposed by Klaus Pohl (see figure 2.1 P. 11). This framework defines the major building-blocks and elements of a Requirements Engineering process needed to establish a vision within an existing context. After carefully reading the literature related to the Requirements Engineering framework, we have noticed that we would achieve a more understandable classification by organizing the found literature

according to this building-blocks proposed in the framework. We believe that the use of this framework is a good fit for our study due to the following reasons:

1. As mentioned previously, the first and main reason is that the framework defines the major structural building-blocks and elements of the Requirements Engineering process (e.g. Elicitation, Negotiation, Documentation, Goals, etc.).
2. It provides a well-structured base for the fundamentals, principles and techniques of Requirements Engineering.
3. It is not adhered to a specific methodology or a type of software project.
4. In addition, this framework consolidates various research results and has been proven to be successful in a number of organizations for structuring their Requirements Engineering process.

For the aforementioned reasons, the use of this framework provides a better organization and structure of the found literature (Requirements Engineering for Decision Support Systems). Note that it is not our aim to do a state-of-the-art discussion about Requirements Engineering since it is already a mature field, but we will focus on the literature pointing specifically to Requirements Engineering for Decision Support Systems.

Before proceeding with the literature classification, we will recall the main concepts and clarify the differences between the blocks from our reference framework. The Requirements Engineering framework is composed of three **Core Activities** (Elicitation, Negotiation and Documentation), and just as the name implies, these are the activities which main purpose is to contribute to the creation of the **Requirements Artifacts** (Goals, Scenarios, Solution-Oriented Requirements). Requirements Artifacts refer to the documented requirements (e.g. Multidimensional Model). Mainly the idea focuses that by iterating the activities, we convert the stakeholder's needs to: goals, scenarios and solution-oriented requirements for a new system. In addition, two cross-sectional activities influence the entire process, namely: **Validation** and **Management**. Each of these blocks are described in figure 3.1 for the reader's better understanding:

Requirements Engineering Framework Definitions by K.Pohl							
Requirements Artifacts			Core Activities			Management	Validation
Goals	Scenarios	Solution-Oriented Requirements	Documentation	Elicitation	Negotiation		
Intention with regard to the objectives, properties or use of the system	Describes a concrete example of satisfying or failing to satisfy a goal (or a set of goals)	Define the data, functions, behavior, quality and constraints Often imply a conceptual solution	Documentation and specification of the elicited requirements according to the defined documentation and specification rules.	Identify the relevant resources. Elicit existing requirements from the identified sources. Develop new and innovate requirements.	Identify conflicts. Analyze the cause of each conflict. Resolve the conflict by means of appropriate strategies. Document the conflict resolution and the rationales.	Observe the system context to detect context changes. Manage the execution of requirements engineering activities. Manage the requirements artifacts.	Denotes checking whether the inputs, performed activities, and created outputs (requirements artifacts) of the requirements engineering core activities fulfill defined quality criteria.

Figure 3.1: Column Definition for Literature Approaches Table based on (Poh10)

From figure 3.1 we may note that the System Context (see chapter 2 for more details regarding the system context) which is also part from the framework, was not included during the literature classification. The System Context mainly focuses on the system's environment that must be considered during the development of any software system such as people, systems, processes, etc. However, we did not find relevant differences when considering the environment for traditional software systems in comparison to Decision Support Systems.

In the following section, we provide the description of the filtering process done in order to obtain the current approaches of Requirements Engineering in Decision Support Systems from the literature.

3.2 Scope of Classification

After detailing the reasons why we have utilized a reference framework, this section describes the process carried out to select the literature that iteratively matched our goals and our reference framework.

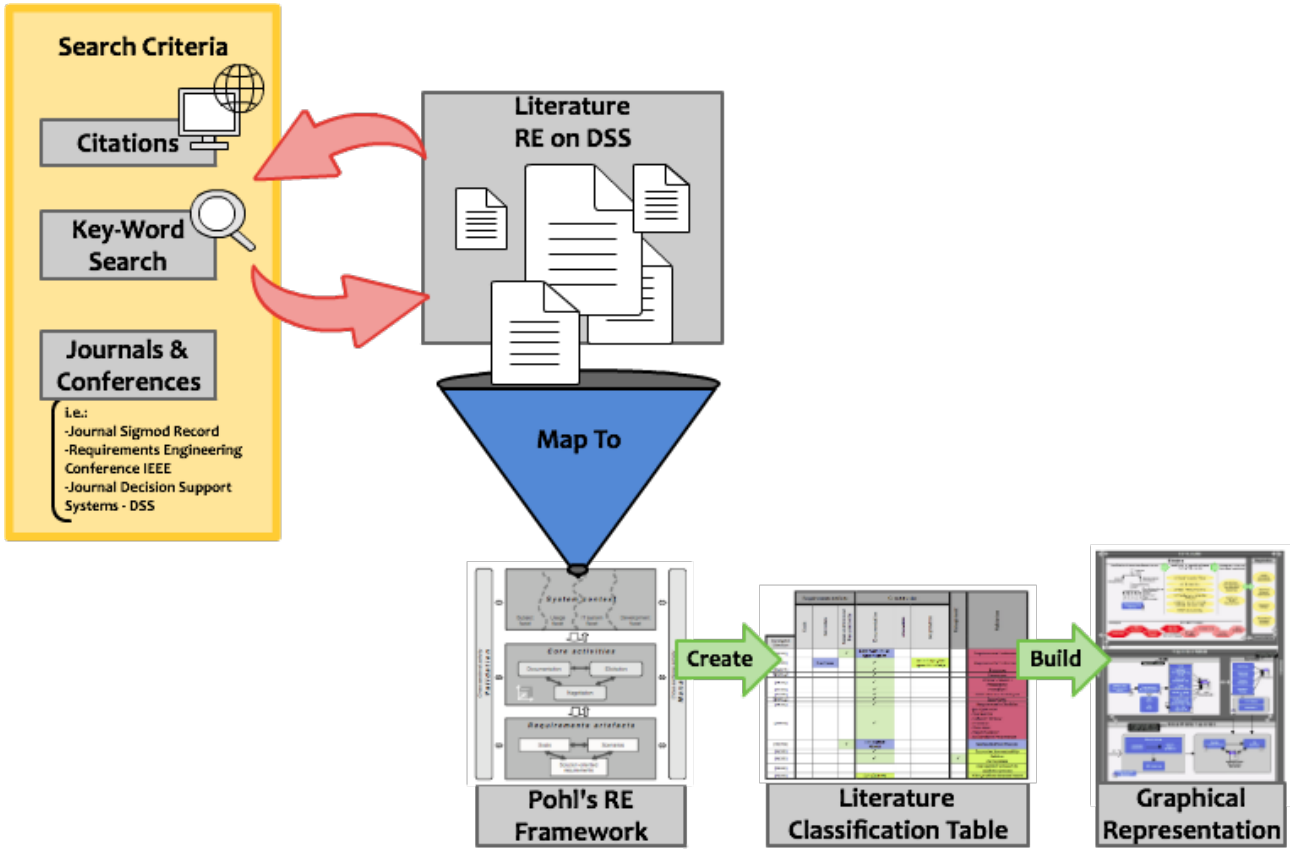


Figure 3.2: Literature Selection and Classification Process

In order to identify the relevant scientific literature, we carried out the following iterative steps (see figure 3.2):

1. **Basic Search:** Initially we began with an exploration of the literature of Requirements Engineering for Decision Support Systems by:

   a) **Journals and Conferences:** Exploring the published documentation through DBLP (Bib14) and Google Scholar (Sch14) for instance.

   b) **Basic Keyword Search:** Making Key-word search in titles and abstracts using the following basic words: Requirements Engineering, and Decision Support System.

   c) **Relevant Books and papers:** We have revised the literature that we were already familiar with, such as (GRT11), (GRG08) and (SWW11).

2. **Extending Literature:** In order to increase the amount of literature, we used the collected literature from the previous step and iteratively continued extending the list of key-words, journals & conferences, and in addition we also extended the literature by revising the citations using "cited-by" in (Sch14).

   Some of the additional keywords were the following: Business Intelligence, Data Warehouse Systems, Requirements Analysis, Information Requirements, Information needs, OLAP, Data Mining, Conceptual Model, Elicitation, etc.

   In addition, some of the most relevant Journals and Conferences from where we extracted the literature were: IEEE International Conference on Requirements Engineering (RE), Data Warehousing and Knowledge Discovery (DaWaK), Business Information Systems Engineering (BISE), Journal Decision Support Systems (DSS), Hawaii International Conference on System Sciences (HICSS), Journal Business Information Systems Engineering (BISE), International Journal of Emerging Technology and Advanced Engineering (IJETAE). All being between 1997 and 2014.

3. **Completing Sample:** We have stopped the iteration process until we completed a representative sample considering the criteria that the literature started repeating.

4. **Filtering & Mapping:** The resulting literature was mapped to the corresponding building-block (activity or artifact) from Pohl's framework. However, we created a filter where we have focused on the Requirements Engineering process (for Decision Support Systems) discarding any work that dealt with already structured requirements without describing the activities matching the framework (e.g. goals elicitation, negotiation, etc.).

5. **Creating Literature Classification:** Subsequently after mapping the literature to the building-blocks from the framework, we have represented these using a literature classification table (see "Classification" subsection further on this chapter).

6. **Building Graphical Representation:** Finally, we have created a graphical representation with the complete view of the models, techniques and guidance specified in the literature. This may be found at the end of this chapter.

3.3 Literature Classification

Notation

After describing the reference framework used for our work and having described the scope of the literature classification, we will in this subsection introduce the notation used to map the selected literature to the building-blocks from the framework (see figure 3.2). The final literature classification is represented in a table where the columns represent the Building Blocks (Requirements Artifacts, Core Activities, Validation and Management), and the rows represent the reference of the investigated literature. Inside, we label the different practices found with the literature (see the complete table in annexe).

Findings were then arranged according to the degree of formalization provided in the literature (see figure 3.3). At the highest degree of formalism, **Models** are identified as a documented representation of information for specific purposes, such as Use Cases. Followed by **Techniques**, which aim to provide the procedures by which a task can be accomplished, for instance, eliciting goals through interviews. In a lower degree of formalism, **Guidance** provides advice or additional information in an informal manner giving direction to achieve the process of Requirements Engineering. Finally, in the lowest degree of formalism, we defined the **check-marks**; these represent a relationship between the a model, technique or guidance to the column of the check-mark. For instance, Use Cases (Scenario Model) must be documented, therefore, it require a check-mark in documentation activity. All these are represented in different color for readers' easier visualization.

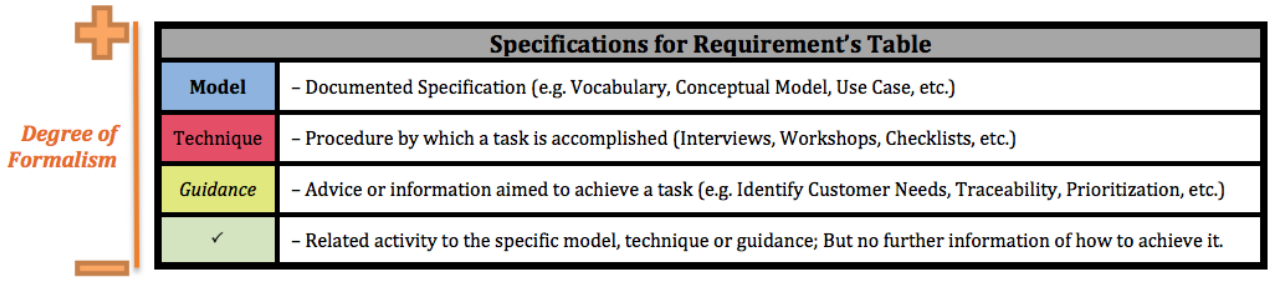


Figure 3.3: Specifications for Literature Classification Table

At the end of this chapter 3, we provide a graphical representation (as seen in figure 3.2) with a complete view of the practices from the literature mapped to each of the building-blocks from the framework.

Classification

After describing the notation, this section provides the classification of practices found in the literature of Requirements Engineering for Decision Support Systems. We will start with the Core Activities. This is followed by the Requirements Artifacts, and finally Management & Validation.

Core Activities:

The Core Activities of Requirements Engineering are the specific activities which aim to contribute to the creation of the Requirements Artifacts. According to the literature, most of the differences between the Requirements Engineering process for specifically Decision Support Systems are found during the elicitation activity. Negotiation mainly consists of the same activities like any software although the concepts of the conflicts may vary for Decision Support Systems (e.g. agreement of stakeholders in the level of granularity in data, agreed dimensions for visualization, etc). And finally, for the readers' best comprehension, documentation will be later explained in the Requirements Artifacts since documentation activity is mostly related to the documentation of the artifacts (e.g. documentation of scenarios using Use Cases).

1. **Elicitation:**

   During the elicitation activity, requirements are elicited from the stakeholders and it's goal is to gather and improve the understanding of the requirements. Figure 3.4 shows the approaches extracted from the literature for the Elicitation of Requirements Engineering in Decision Support Systems.

   The main ideas point that three main 'sources' are used in order to obtain the requirements: business users, data processing staff/information administrators, and from the operational sources:

   - **Business Users:** First, Business users (end-users), are the ones who decide the future target requirements (GR09) (SWW11). Decision Support Systems usually start with the business requirements and fully engage the stakeholders (IBM09). In addition to this, the elicitation process should not just address the employees who have the operational tasks; however, it is also important to include middle-management responsibilities and decision-making competence, such as department or team leaders (SWW11). It is crucial to know how to deal with the type of user from whom the requirements are elicited during the interview, for instance overbooked user, comatose, overzealous, know-it-all, clueless and nonexistent user (Kim98). As seen in the check-marks from the figure 3.4, during the elicitation activity we start by extracting mainly the goals from the business users.
   - **Data Processing Staff:** In addition, a second human source is also needed which includes the information administrators and the data processing staff, considered as the second source are the point of reference for designers (GR09). This source must be asked regarding the technical constraints to build the information needs in order to build the right system (SWW11).
   - **Data Sources:** In addition to the two human sources, in the Data Warehouse environment, the data is also collected from other data sources that may be gathered from heterogeneous sources within and without the enterprise environment (PdC03).

   After identifying the main sources, it is now possible to start identifying the relevant requirements from the identified sources:

	Requirements Artifacts			Core Activities				
Investigated Literature	Goals	Scenarios	Solution-Oriented Requirements	Documentation	Elicitation	Negotiation	Management	Validation
[GR09]			**Conceptual Model**	✓	*Identify: Measures, Dimensions, Facts*			
[PdC03]			**Business Rules Specification**	✓	*Elicit Business Rules*			
[GR09]	*Collaboration of designers and end users*		✓		*Elicit goals and limitations*		*Prioritization*	
[Kim98]				✓	Facilitated Sessions			
[GR09]				✓	Facilitated Sessions + Brainstorming			
[Kim98]				✓	Interviews			
[GR09]				✓	Interviews			
[Kim98]	*Identify Customer Needs*			✓	Specialized Inteviews			
[PdC03]				✓	Specialized Inteviews			
[Kim98]				✓	Specialized Inteviews			
[Kim98]				✓	Face-to-face Interviews			
[Kim98]				✓	Interviews by User			
[Kim98]				✓	Data Audit Interviews			✓
[GR09]				✓	Interviews			
[Kim98]				✓	Elicitation Checklist			
[PdC03]				✓	DW Requirments Workshop	*Promote fast agreement about the course of actions for system's delivery*		
[IBM09]				✓	-Brainstorm -Analysis Interviews -Group Interface -Observations -Prototyping Reverse -Engineering Surveys -Workshops	✓		
[RB12]	✓	✓	✓	✓	Prototypes			
[GR09]	✓				*Elicitation Source: Business Users*			
[IBM09]	✓	✓			*Start w/Business Requirements*			
[SWW11]	✓	✓			*Include middle management & decision-making competence*			
[GR09]		✓	✓		*Elicitation Source: Data Processing Staff*			
[SWW11]			✓		*Elicit Technical Constraints*			
[BDGM08]	✓		✓	✓	*Data, technical resources and security limits*			
[PdC03]			✓	*Use pre-existing operational documentation*	*Data Extraction & Integration Procedures*			
[PdC03]			✓	✓	*Data Collection Procedures*			
[SWW11]	✓				*Elicit Future Target Requirements*			
[PdC03]		*Determine the extent metrics can be operated*		✓	*Identify Facts, metric operations, and dimension hierarchies.*			

Figure 3.4: Elicitation Activity - Proposed Approaches in Literature

- **Elicit "Needs" Rather Than "Wants":** Most of the practices during the interview process are done the same as in operational systems, although one of the main differences found in the literature is the interviewer must ask "what do you do (and why)" and not ask "what do you want" in order to extract the customer needs (Kim98) (RB12).

- **Elicit Business Rules:** Business rules to regulate the Data Warehouse, and Data Mart functionalities must be also elicited (PdC03).

- **Elicit Data Collection Procedures:** (PdC03) mentions that the sources must be integrated according to the defined rules, considering the data exchange between the identified systems, the periodicity, the loading of the data and its priorities of what is first executed.

- **Elicit Data Extraction & Integration Procedures:** In Data Warehouse projects, specifying procedures (extraction and integration) is an important task to control how data is collected from heterogeneous sources inside and outside the enterprise environment (PdC03); to achieve this, the authors suggest the analysis of pre-existing documentation.

- **Identify Data Technical Constraints Resources & Security Limits:** While making the interviews to the Data Processing Staff, requirement limits must be elicited, such as the data (access to the sources of information, and data quality), human and technical resources (limitations of hardware and software, human resources) and finally to security issues (BDGM08). In addition, this phase also includes a collaboration between the business users and the designers, aiming to achieve a common agreement to specify the goals and the limitations of the Data Warehouse (GR09). According to the check-marks from the figure 3.4, all these mentioned constraints, limitations, and procedures must be taken into account during the elicitation given that these will directly influence the final delivery in the Solution-Oriented Requirements to produce the right model.

- **Define/Create Vocabulary:** The vocabulary and glossary is one of the most addressed practices by the literature due to its great importance during the creation of Decision Support Systems. The Vocabulary will be explained in detail on the Solution-Oriented Requirements section, however, the creation of the Vocabulary must start since the initial phase of a Decision Support System project in order to avoid linguistic inconsistencies during the elicitation process and the desires may be expressed using homogenized terms (i.e. dimensions) understood by business users (domain experts), developers, or anyone involved in the project (ACNM06).

After identifying the relevant sources and the requirements from these identified sources, the literature points out that from this point on, we may start identifying facts, metrics and dimension hierarchies that may be useful and needed for the system (PdC03). (GR09) agrees that the identification of the facts, dimensions and measures are a crucial step before creating the conceptual design of the data warehouse or the data mart. In addition, future target requirements should also be elicited (SWW11).

Regarding the techniques, four main techniques were found especially appropriate to elicit requirements for Decision Support Systems, namely:

- *Interviews:* Interviews usually have several preliminary activities, these are well described by (Kim98) and these include pre-interview research, interviewee selection, question development, interview scheduling, and preparation. These are usually done to individual users or small homogeneous groups in order to achieve a more detailed

list of the specifications (GR09). Different type of questions that may be done to the users are: Open-ended, closed and evidential (GR09). The specialized interviews may also include straight questions tailored to the Data Warehouse issues, such as granularity and multidimensionality matters, for this, (PdC03) suggests using kimball's approach.

On interviews, (Kim98) suggests starting by initiating a conversation with the business users with an easy topic, for instance their jobs, their vision, the future objectives and challenges; this is followed by the IT interviews. The authors recommend doing face-to-face or voice-to-voice interviews without relying on non-interactive surveys or questionnaires, nevertheless, not because you produce a 3" documentation of elicited requirements means that the requirement analysis is already finished (Kim98). Finally, the authors claim that data audit interviews are important in order to understand if the data is complete, reliable and is ready to align with what the business actually needs.

- *Facilitated Sessions:* Instead of being like interviews focusing on small and homogeneous groups, facilitated sessions are done in large, and heterogeneous groups and everyone can actively participate; the final result is a very detailed list of specifications (GR09) (Kim98).

- *Data Warehouse Requirements Workshop:* The requirements workshop is a specialized workshop specifically for the Data Warehouse that encourage consensus over the scope of the multidimensional solution (PdC03). According to the author, this is done in order to: achieve an agreement among all the involved parties about the course of actions of the Data Mart delivery, to efficiently solve the encountered issues and to help capture the major functionalities and operational constraints.

- *Observation:* This technique is done to observe the company's events and their personnel. This then can be used in order to see what are the user needs on a Data Warehouse basis. This technique is effective for defining how the individuals do their job and assessing their work environments (IBM09). Repeated observation is mostly needed to corroborate the facts gathered.

Note from check-marks in figure 3.4 that all techniques during the elicitation must provide a document as an output. Further from the techniques of elicitation described above, there are also assistance techniques supporting the requirements elicitation of Decision Support Systems which must produce documentation also. These include the following:

- Reverse Prototyping: Building prototypes helps users understand their requirements and the system's capabilities (RB12). Prototypes can be used to represent Goals, Scenarios or Solution-Oriented Requirements.

- Brainstorming: This is considered as a creativity technique to generate ideas from a group of stakeholders depending on their needs (IBM09).

- Elicitation Checklist: During the interview, it is recommended by the literature to bring out a list which includes the data elements in order to track what is needed (Kim98).

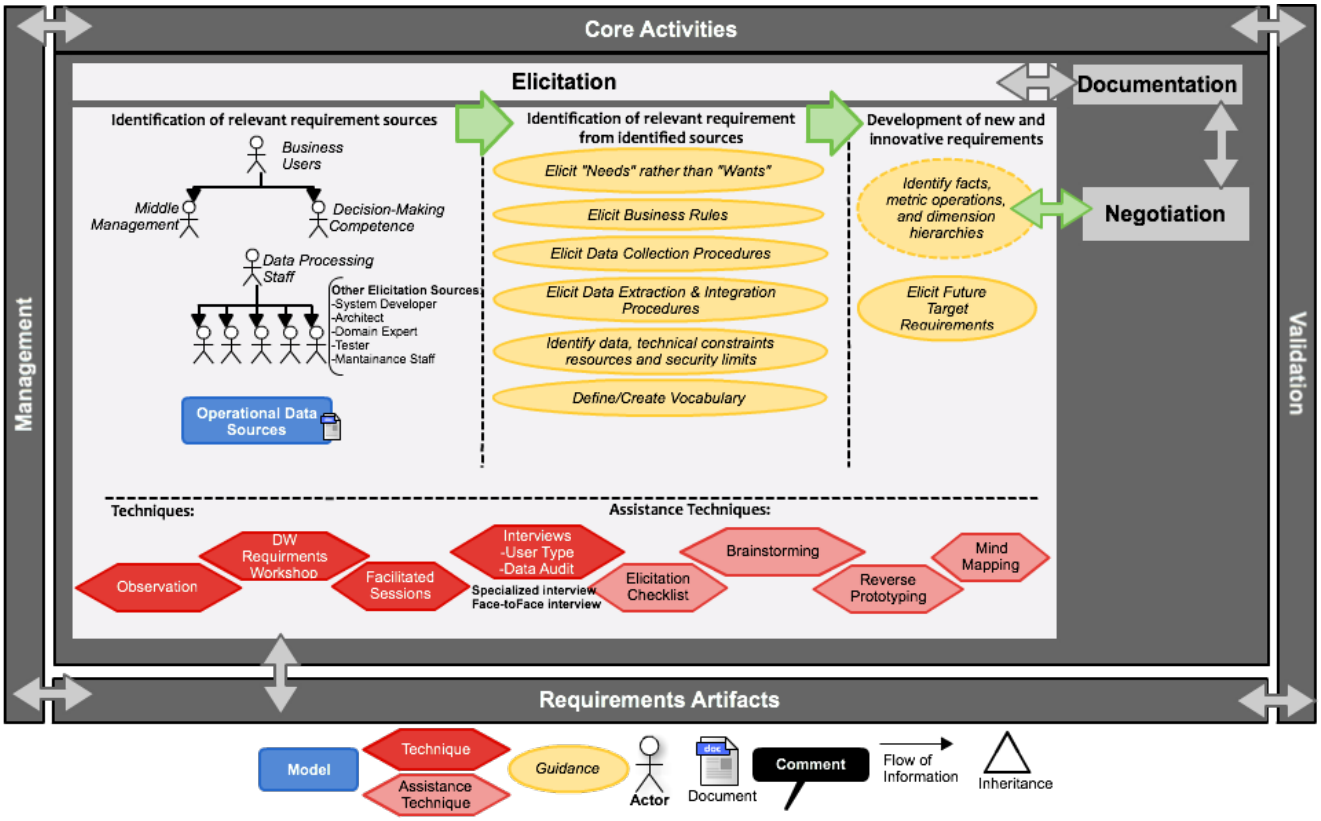


Figure 3.5: Elicitation - Pohl's Framework Instantiation for Decision Support Systems

A summary of Pohl's framework instantiation with regards to elicitation may be seen in figure 3.5.

2. **Negotiation:**

	Requirements Artifacts			Core Activities			Management	Validation
Investigated Literature	Goals	Scenarios	Solution-Oriented Requirements	Documentation	Elicitation	Negotiation		
[PdC03]			✓	**Data Warehouse Specification**	✓	✓	*Requirements Iteration*	
[PdC03]			✓	**Data Warehouse Specification**		*Overlapping Requirements*		
[PdC03]		**Use Cases**		✓		*Reusability of the agreed knowledge*		Requirements Conformance
[PdC03]				✓		✓		**Revision Report**
[PdC03]				✓	DW Requirments Workshop	*Promote fast agreement about the course of actions for system's delivery*		
[SWW11]						*Identify Overlapping Requirements*		
[PdC03]	*Project Objectives*					*Agreement among all stakeholders*		
[GR09]	*Investigate available sources*					*Negotiation Stategy*		

Figure 3.6: Negotiation Activity - Proposed Approaches in Literature

The final product must fulfil the needs and wishes from the stakeholders although usually different opinions given by the stakeholders may be contradicting one another. Figure 3.6 shows the practices extracted from the literature for the Negotiation activity in Requirements Engineering for Decision Support Systems. In Data Warehouse for instance, the objectives and project restrictions such as aggregated information must be agreed by all stakeholders involved in the project (PdC03); the check-mark indicates that this clearly relates to the project goals or objectives. In addition, mutually dependent and the requirements overlapping each other must be identified and solved with a common agreement during this negotiation activity (SWW11), for instance: conflicting points of view regarding the visualization, either bar or pie charts.

Accordingly, (PdC03) claims that redundancy between requirements that oversee the entire Data Warehouse must be avoided before producing the "Data Warehouse specification" document. The check-marks in documentation in figure 3.6 imply that after achieving an agreement between stakeholders, documentation is of great importance in order to promote re-usability of 'agreed knowledge' in the project in case that similar conflicts take place (PdC03).

As seen from figure 3.6, negotiation activity in Decision Support Systems requirements does not change much from other systems. What may be seen from the figure 3.7 and in accordance with the study done by (PdC03), negotiation is addressed by relatively few approaches, especially when describing how the needs must be prioritized. Although what does change is the concept of the conflict; for instance, stakeholders may or may not agree on the level of detail in the data, dimensions needed (e.g. country, time), or metrics (e.g. price) to be used, etc. A summary of Pohl's framework instantiation with regards to negotiation may be seen in figure 3.7.

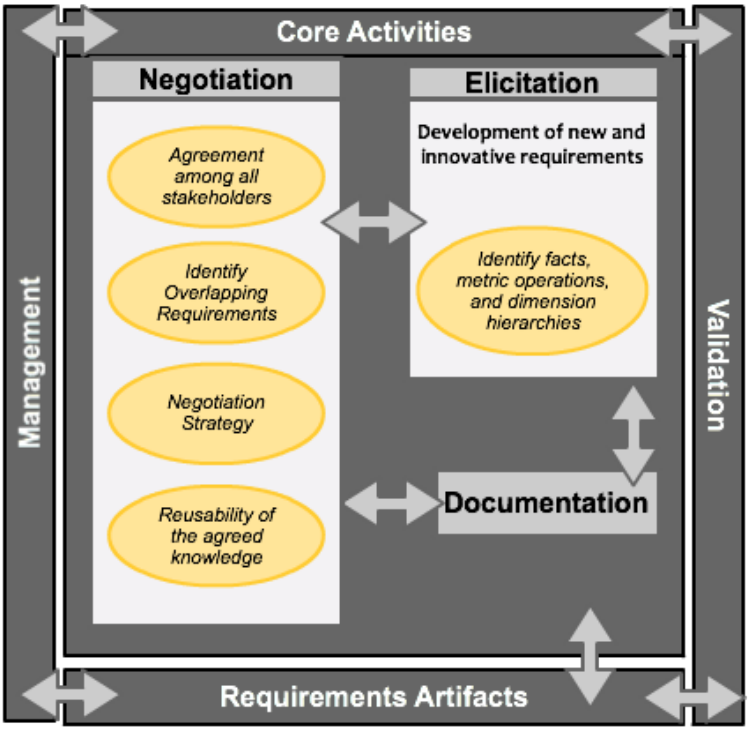


Figure 3.7: Negotiation - Pohl's Framework Instantiation for Decision Support Systems

3. **Documentation:**

   Documentation is an activity that is on-going throughout the complete cycle of Requirements Engineering. The current literature appears to suggest that documentation activity is at the core of the requirements analysis by suggesting abounding documentation for models, techniques and assistance techniques (e.g. Tropos, Use Cases, Process diagrams, Conceptual design, Glossary, Constraints, Business Rules, Identified Sources, Testing Methods, Traceability Matrices, etc). As seen from the example list in the previous sentence, the activity of documentation is mostly related to the document ion of the Requirements Artifacts (Goals, Scenarios, Solution-Oriented Requirements). For this reason, we have not provided a figure/neither table for the documentation instantiation given that we explain in more details each of the documentation of these artifacts in the Requirements Artifacts section (for instance, Use Cases defined in Scenarios use the documentation "Use Cases Specification").

   However, diverse templates or meta-documents are proposed by in order to document functional, non-functional and domain-specific requirements which suit precisely for the description of a Data Warehouse and its Data Marts. Some of these examples provided by (PdC03) include documenting the following:

   - Requirements Management Plan
   - Project Glossary
   - Data Warehouse Vision
   - Data Mart Vision
   - Use Cases Specification
   - Multidimensional Requirements Specification
   - Non-Functional Requirements Specification
   - Business Rules Specification
   - Revision Report

   In the work of (GRT11), it is stated that produced documents may be distinguished into releases corresponding to project milestones. According to the authors of this work, documentation must be "formal and light documentation" which leads to having a clear and non-ambiguous-to-date documentation.

   Moreover, according to (BDGM08), the main focus of requirements documentation in the Business Intelligence field is that it enables the achievement of requirements to be consistent and also traceable over the entire project. Plus, the authors claim that the final requirements documentation containing the all the specifications and including the multidimensional model will be used as a starting point for developers to create the system for instance, based on the common lexis (glossary) and the cross-referenced concepts related to the target business domain.

Requirements Artifacts:

Requirements Artifacts are composed of three essential types of documented requirements, namely: goals, scenarios and solution-oriented requirements.

1. **Goals:**

	Requirements Artifacts			Core Activities				
Investigated Literature	Goals	Scenarios	Solution-Oriented Requirements	Documentation	Elicitation	Negotiation	Management	Validation
[GR09]	*Collaboration of designers and end users*		✓		*Elicit goals and limitations*		*Prioritization*	
[PdC03]	*High-level requirement vision*			✓				
[PdC03]	**Data Warehouse Vision**			✓				
[PdC03]	**Data Mart Vision**			✓				
[GR09]	**Tropos**			✓				
[MSS]	**Tropos**			✓				
[PC02]	**Softgoal Inter-dependency Graph**			✓				
[IP01]	**Function Refinement Tree**	**Use Cases**		✓				
[MPT07]	**Goals' Classification**			✓				
[PdC03]	**Non-Functional Requirements Framework (NFR)**			✓				
[Kim98]	*Identify Customer Needs*			✓	Specialized Inteviews			
[BDGM08]	*Identify Customer Needs*							
[RB12]	*Identify Customer Needs*							
[PdC03]	*Customer Needs + Business Application Domain*			**Requirements Management Plan**				
[PdC03]	*Granularity in data marts, constraints for MD analysis, and rules for data exchange and load.*			**Requirements Management Plan**				
[PdC03]	*Requirements Management Planning Phase*							
[PdC03]	*Project Objectives*					*Agreement among all stakeholders*		
[GR09]	*Investigate available sources*					*Negotiation Stategy*		

Figure 3.8: Goals Artifact - Proposed Approaches in Literature

These refer to the stakeholder's intentions. Figure 3.8 shows the literature study of Goals in Requirements Engineering for Decision Support Systems. According to the work of (PdC03), there is a lack of higher level requirements vision during the design decisions that do not take into account important aspects which are crucial for the project's success including the user needs, multidimensional restrictions and the quality constraints. Therefore, is crucial to gather all these information and elicit the "User Needs". In addition, the authors claim that the Data Warehouse vision must be created in advance to describe the motivation and problem issues, general objectives, project scope, stakeholders' profile and other issues related to the Data Warehouse; whereas the Data Mart mission is in charge of collecting the high level

Data Mart user needs, features and actors that support these needs. After acquiring the goals, these are then documented by using several models. These may the following:

- *Tropos:* Tropos is based on i* framework and according to (MSS), it has been properly adapted and extended to fit the Data Warehouse specificities placing emphasis mostly in the early requirements which are directly related to the goals pointing why the data warehouse is needed to be developed. Tropos methodology proposed in (BPG04) is carried out in order to identify the business goals (see figure 3.9).

- *Softgoal Interdependency Graph:* Another approach proposed by the literature (PC02) is DW-NFR Framework (Data Warehouse - Non-Functional Requirements), which uses Softgoal Interdependency Graph (see figure 3.9). This graph represents the influence of interdependency of one softgoal on another. In order for a softgoal to be "satisfied," there must be sufficient positive evidence and little negative against the goal.

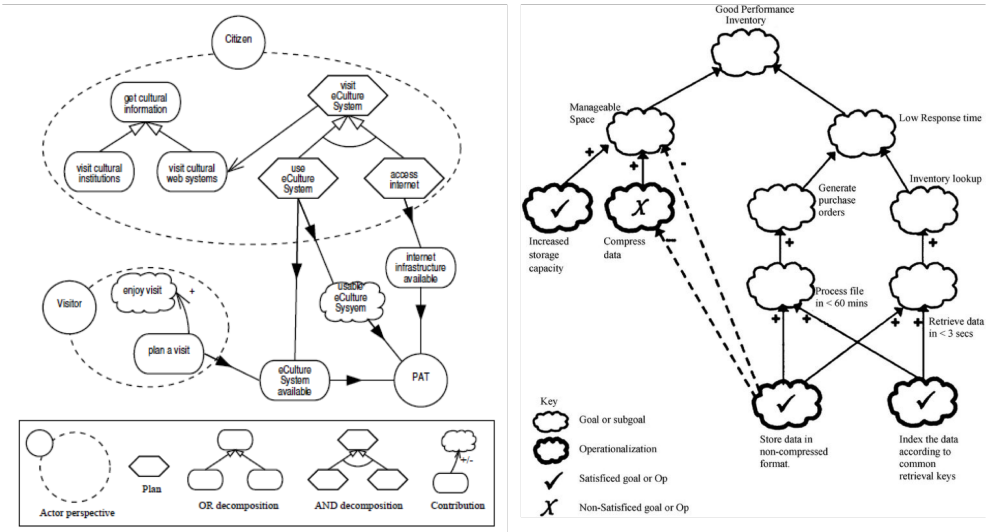


Figure 3.9: Tropos (Left) and Soft-goal Interdependency Graph (Right) reprinted from (Pro14) and (Pro12)

- *Function Refinement Tree:* External Interactions between the environment and the systems can be always divided into several functions. (IP01) proposes the use of the function refinement tree in order to organize in a refinement hierarchy being the leaf nodes the elementary functions. It gives an entry point to build the use cases.

- *Goal's Classification:* Furthermore, the work from (MPT07) proposes the classification of goals decision makers expect to fulfil with their envisaged Data Warehouse. According to this work, three kinds of goals are proposed depending on the level of abstraction:

  - Strategic goals: illustrates the top level of abstraction. They represent main objectives of the business needed to be changed. For example: "increase sales," "decrease cost," etc.

  - Decision goals illustrates the medium level of abstraction answering the question: "how can a strategic goal be achieved?". For example "determine some type of promotion" or "open new stores".

– Information goals: illustrates the bottom level of abstraction answering: “how can decision goals be achieved in terms of information required?”. For example “analyze customer purchases” or “examine stocks.”

Is important to notice that after the hierarchy is defined with its levels of abstraction, the information requirements can be directly obtained from the bottom level, meaning the information goals (MPT07); later, facts and dimensions may be discovered from these information requirements in order to specify the corresponding multidimensional model needed for the Data Warehouse.

- *Non-Functional Requirements:* This framework proposed by (PdC03) allows reusing the organization’s knowledge and methods (stored in design catalogues) for non-functional requirements (NFR). In addition, the authors exemplify that Data Warehouse-Extended NFR Framework (DW-ENF) applied for requirements such as indexing, disk space optimizers, loading schema and other key characteristics related to the Data Warehouse design. In addition, DW-ENF non-functional requirements list may also be used as a checklist to guide the developer of the system to consider all the quality requirements.

Note from the check-marks in figure 3.9 that after requirements are defined using the different goal models mentioned previously, these are then documented in the “Goals Documentation.” Before moving into the solution-oriented requirements, these must be checked in order to be in accordance with the business rules. A summary of Pohl’s framework instantiation with regards to goals may be seen in figure 3.10.

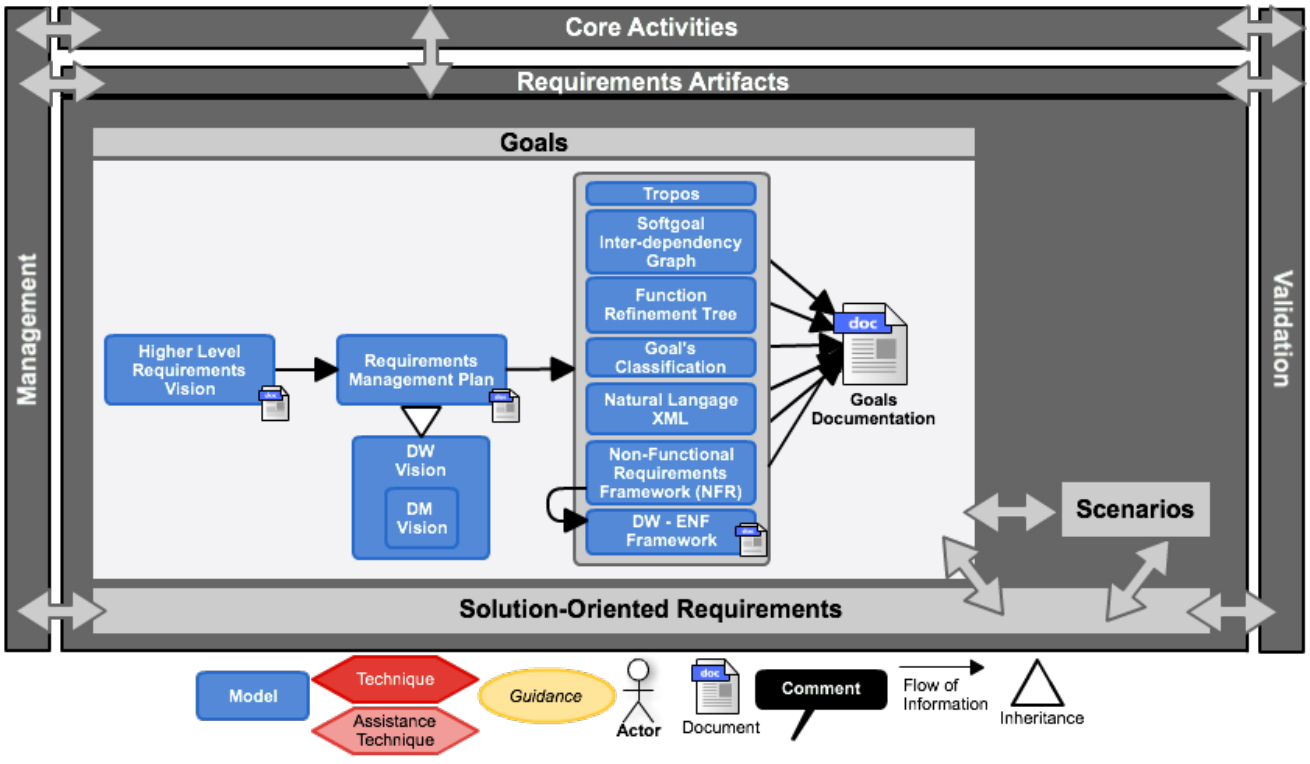


Figure 3.10: Goals - Pohl’s Framework Instantiation for Decision Support Systems

2. **Scenarios:**

	Requirements Artifacts			Core Activities				
Investigated Literature	Goals	Scenarios	Solution-Oriented Requirements	Documentation	Elicitation	Negotiation	Management	Validation
[PdC03]		**Use Cases**		✓		*Reusability of the agreed knowledge*		Requirements Conformance
[IP01]	**Function Refinement Tree**	**Use Cases**		✓				
[PdC03]		**Use Cases**		✓				
[PdC03]		**Use Cases**		**Use Cases Specification**			*Reuse of common behavior scenarios*	
[GOIM+00]		**Business / System Use Case**		✓				
[GOIM+00]		**Sequence Diagram**		✓				
[GOIM+00]		**Process Diagram**		✓				
[GOIM+00]		**Workflow Models**		✓				
[IBM09]		**-Use Case User Stories -Process Models -Domain Models -Workflow Models**		✓				
[PdC03]		*Determine the extent metrics can be operated*		✓	*Identify Facts, metric operations, and dimension hierarchies.*			

Figure 3.11: Scenarios Artifact - Proposed Approaches in Literature

Scenarios document concrete examples of the system's usage by illustrating the fulfillment or non-fulfillment of the goal. Figure 3.11 shows the literature study of Requirements Engineering in Decision Support Systems. The identified scenarios to be documented were the following:

- *Use Cases:* These may be used in order to specify scenarios either textual or graphic, for instance UML. According to (PdC03), the standard model may be used for the main Data Warehouse functionalities such as extract, transform and accessing the data. In addition , it is believed by the authors that "Querying" and "Loading" processes may be designed as a single use case using parametrized entries. Complementing (PdC03), (GOIM00) claim that the Use Case specification is used as the documentation that details the procedures required to implement the functionalities of the possible sequences and enable the reuse of behaviour shared among diverse Data Mart scenarios. These last authors place emphasis that a business use case must be created for each of the business process involved. In addition, a use case diagram is used to illustrate the context within the boundaries of the company.

- *Sequence Diagram:* This diagram represents diverse scenarios or objects acting in collaboration between one another to finally provide with an outcome (GOIM00).

- *Process Diagram:* Illustrate the work-flow performed to obtain some goal of the organization. Its been said by (GOIM00), that process diagrams illustrate the different roles and the data which is required and finally produced by each of the activities. According to the same work, process diagrams may be easily adapted to UML diagrammatic representations by using activity diagrams with vertical solid swim-lanes grouping activities from the same role.

- *Workflow Model:* The workflows mainly describe the procedural steps (business processes) which are done in the enterprise. According to (GOIM00), the workflow of the business processes can be modeled by means of activity diagrams illustrating the interactions between the roles. However, we would like to emphasize on the use of workflows for specifically Data Warehousing, for instance, the ETL workflows. (SS07) defines the representation of an ETL workflow as the means in which the data is loaded into the Data Warehouse, including: the identification of sources, extraction of information, transformation from heterogeneous sources, cleansing of data and loading), and other activities.

From the check-marks in figure 3.11, is noticeable that most of the scenarios identified are models (Use Cases, ETL Workflows, etc.) therefore, this outputs must be documented to provide input for the Solution-Oriented Requirements. A summary of Pohl's framework instantiation with regards to scenarios may be seen in figure 3.12.

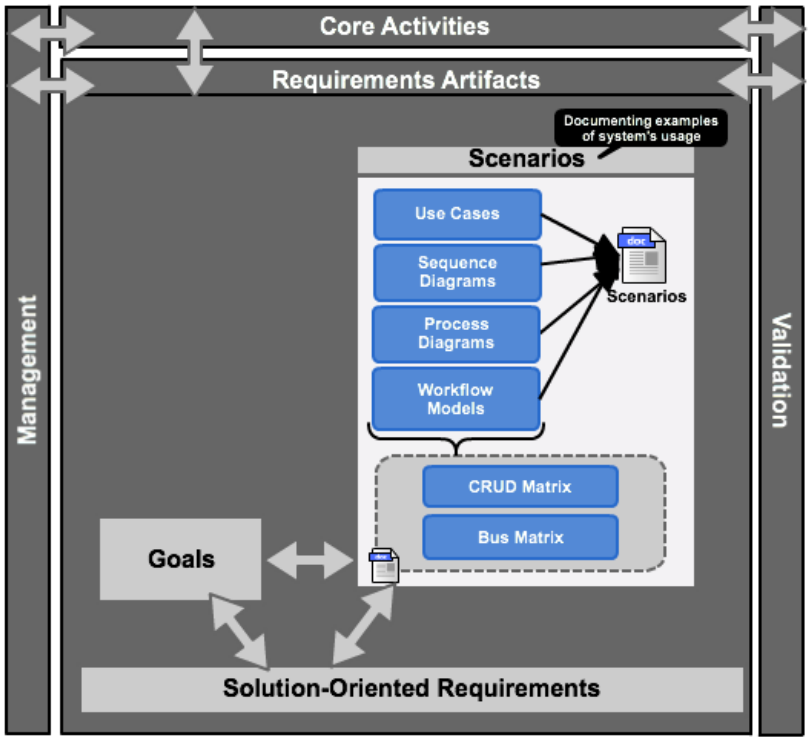


Figure 3.12: Scenarios - Pohl's Framework Instantiation for Decision Support Systems

3. **Solution-Oriented Requirements:**

	Requirements Artifacts			Core Activities				
Investigated Literature	Goals	Scenarios	Solution-Oriented Requirements	Documentation	Elicitation	Negotiation	Management	Validation
[ACNM06]			**Business Vocabulary**	✓				
[SWW11]			**Glossary**	✓				
[GR09]			**Glossary**	✓				
[PdC03]			**Glossary**	✓				
[BDGM08]			**Glossary**	✓				
[GOIM+00]			**Glossary**					
[ACNM06]			**Vocabulary Shared Ontology**	✓				
[Str02]			**MD Vocabulary**	✓				
[GR09]			**Conceptual Model**	✓	*Identify: Measures, Dimensions, Facts*			
[WS03]			**Conceptual Model**	✓				
[IP01]		✓	**CRUD Matrix Conceptual Schema**	✓				
[WS03]			**Information Map**	✓			✓	
[PdC03]			**Non-Functional Requirements Specification**	✓				
[PdC03]			**Multidimensional Requirements Specifications**	✓				
[PdC03]			**Business Rules Specification**	✓	*Elicit Business Rules*			
[Kim98]			**Bus Matrix**	✓				✓
[PdC03]			**DW - ENF Framework**	*Checklist for Quality Requirements*			✓	
[PdC03]			*Configurable Views*	✓				
[PdC03]			*MD restrictions and quality constraints*	✓				

Figure 3.13: Solution-Oriented Requirements Artifact - Proposed Approaches in Literature

Goals and scenarios are used to form solution-oriented requirements. These requirements refer to the documentation of the conceptual solution which satisfy the goals and the scenarios while also defining the basis for the developers to create the system. Figure 3.13 shows the literature study of Requirements Engineering in Decision Support Systems.

One of the most pointed out practices by the literature, is the creation of a shared terminology. The creation of this is done in order to avoid linguistic inconsistencies between stakeholders and IT people, the desire is expressed to homogenize used terms (e.g. dimensions) (SWW11). According to the literature, these may be expressed in different manners depending on the details and formalism:

- **Business Vocabulary:** At the highest level, a common "Business Vocabulary" is created starting from the goals in order to provide communication between business users and IT people having an integrated terminology and view of data (ACNM06).

- **Glossary (with cross-referencing):** On a deeper level, a project glossary is created which organizes the BI terminology related to problem domains (PdC03). In the glossary, the definitions are firstly identified along with abbreviations, synonyms, etc., in order to establish a general lexis which must be agreed by all the parties related to the project (BDGM08). In contrast to the vocabulary, the literature (GOIM00) proposes the use of a glossary having a cross-reference structure to control the traceability relationships from the business processes. This glossary will support the designer during the creation of the conceptual and design phase (GR09).

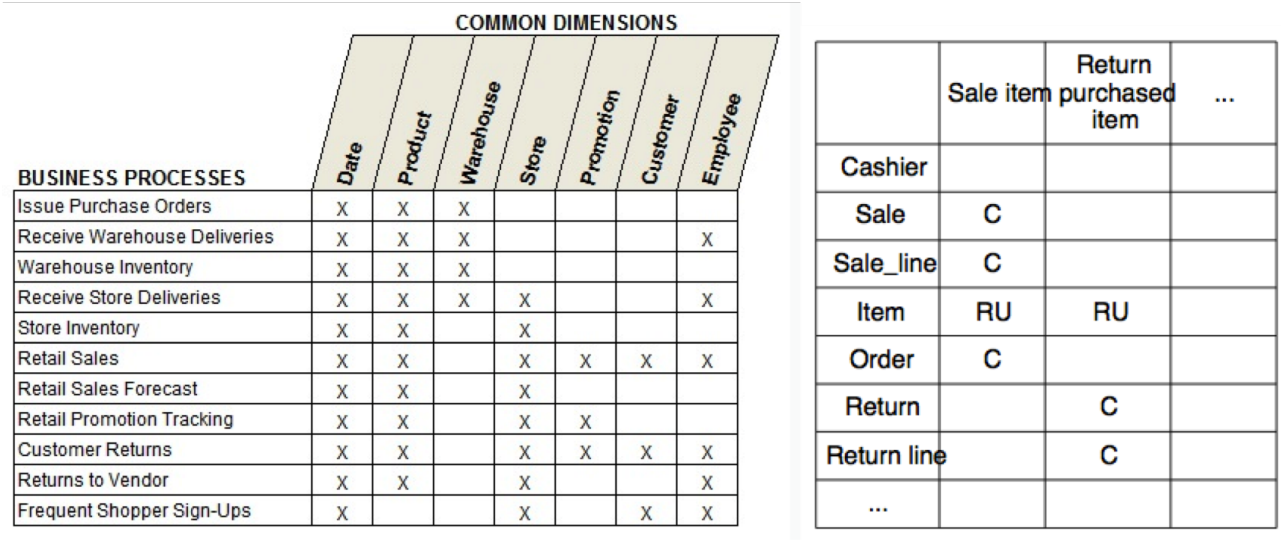
- **Shared Ontology:** Followed by this, a shared ontology is used as a common, agreed vocabulary of both stakeholders and developers. This ontology is considered as a superset of vocabulary, being sufficiently rich to describe the data source (ACNM06). The ontology includes terms used to denote the entities and their relationships (e.g. inheritance, formal structures of entities as attributes, and any constraints) (ACNM06). One of the main differences from the glossary and the ontology is that this is a machine-readable format while glossary may be illustrated by using a UML graphical representation.

- **Multidimensional Vocabulary:** Finally a "MD Vocabulary" focuses on the terms identified in the initial vocabulary and illustrates these as multidimensional concepts (e.g. facts, dimensions, measures) and their relationship among them (e.g. aggregate level) (Str02).

As it may be seen from the check-marks in figure 3.13, all these formalizations previously mentioned must be part of the documentation along with the business rules presented in the goals section. As one of the results of the requirements analysis, a complete and detailed document called "Information Map" should be created representing where data are sourced from (WS03). Afterwards, all this is sent to the creation of the "Multidimensional Model" and the "ETL Workflow."

Another two important inputs taken into account in the Solution-Oriented Requirements are the CRUD matrix and Bus Matrix. These stand in the middle of the scenarios and solution-oriented requirements due that they take into account the elicited goals and business processes in order to find a multidimensional semantics, however, these are still one step away before building a final conceptual model/ETL workflow. The literature describes these as follows:

- *CRUD Matrix:* It is also a Function Decomposition Table that is used to obtain the first version of the conceptual model. According to (IP01), this matrix by is formed by using the top rows as the Use Cases of the system and the left column representing the classes extracted from the sequence diagram. The authors specify that the table must be filled using the following letters: (C) instances that must be created, read (R), update (U) and delete (D) (see figure 3.14). This represents all the building blocks in order to build the multidimensional model just by adding the required class relationships.



BUSINESS PROCESSES	Date	Product	Warehouse	Store	Promotion	Customer	Employee
Issue Purchase Orders	X	X	X				
Receive Warehouse Deliveries	X	X	X				X
Warehouse Inventory	X	X	X				
Receive Store Deliveries	X	X	X	X			X
Store Inventory	X	X		X			
Retail Sales	X	X		X	X	X	X
Retail Sales Forecast	X	X		X			
Retail Promotion Tracking	X	X		X	X		
Customer Returns	X	X		X	X	X	X
Returns to Vendor	X	X		X			X
Frequent Shopper Sign-Ups	X			X		X	X

	Sale item	Return purchased item	...
Cashier			
Sale	C		
Sale_line	C		
Item	RU	RU	
Order	C		
Return		C	
Return line		C	
...			

Figure 3.14: Bus Matrix (Left) and Crud Matrix - Function Decomposition Table (Right) based on (Kim98) and (IP01)

- *Bus Matrix:* Another approach is proposed by (Kim98). This is a formal documentation that allows the identification of the relation of the measures (business processes) and the dimensions (e.g. group by, filter by). According to the authors, this matrix is helpful to achieve understanding among the business and IT people (see figure 3.14).
- *Natural Language:* In addition, (LB02) proposes that requirements may also be written in natural language, although these need to be translated into a formal specification language for execution. The author claims that after gathering the requirements, this document can be converted into Extensible Markup Language (XML) format in order to be machine-readable.

As the final result from the complete instantiation including the Core Activities and the Requirements Artifacts, is a conceptual model (multidimensionally oriented) is created. Therefore, the preliminary phases are crucial to create the model for the Data Warehouse / Data Mart given that facts must be identified, dimensions, measures by queries and interaction with the user (WS03). In addition to the conceptual model, by using the workflow created from the scenarios, the conceptual model, the business rules and the sources, the ETL workflow may also be created. Afterwards, (PdC03) proposes to store the documentation and all the details into the "Data Warehouse Requirements Specification."

A summary of Pohl's framework instantiation with regards to solution-oriented requirements may be seen in figure 3.15.

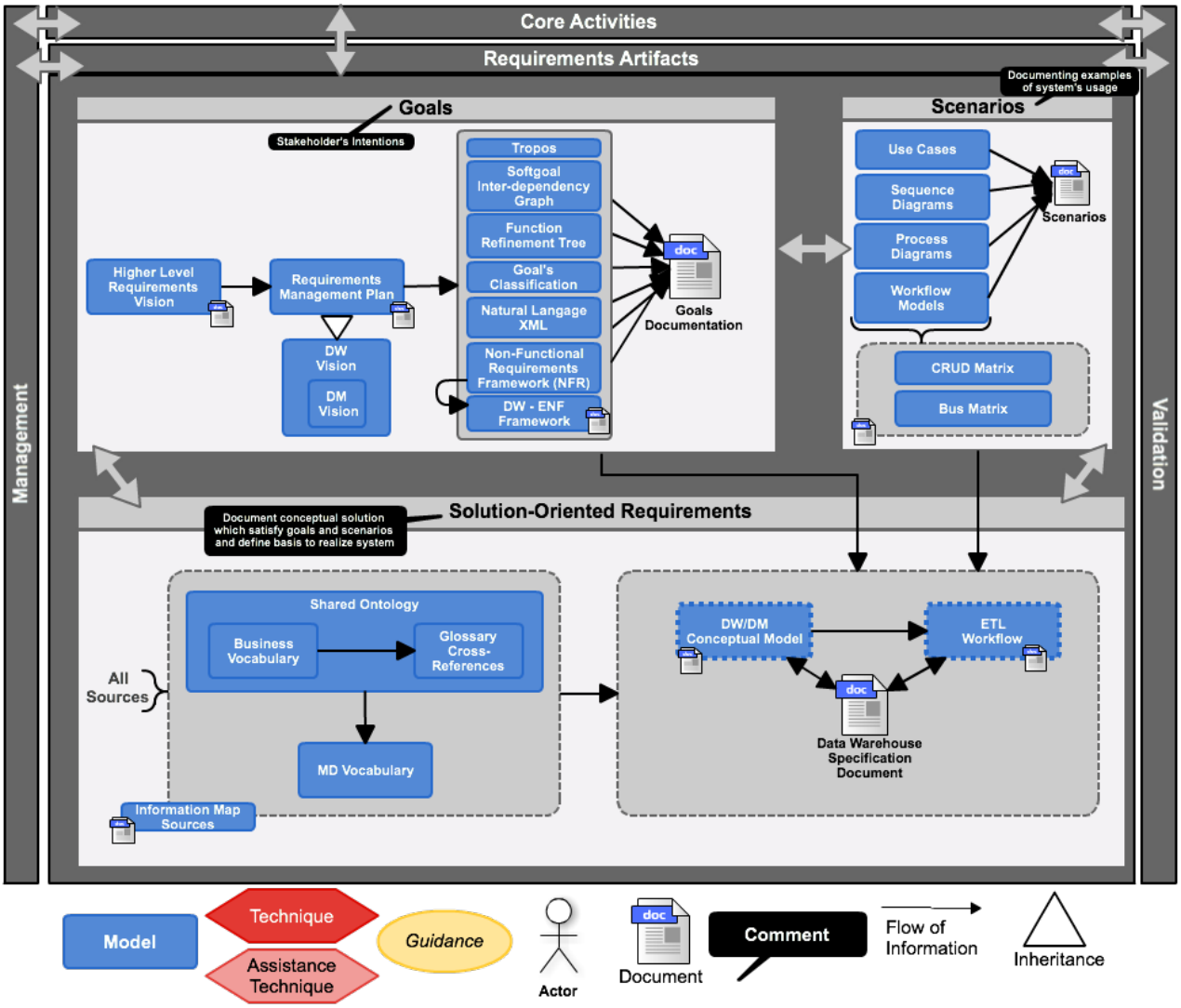


Figure 3.15: Solution Oriented Requirements - Pohl's Framework Instantiation for Decision Support Systems

Management:

Investigated Literature	Requirements Artifacts: Goals	Requirements Artifacts: Scenarios	Requirements Artifacts: Solution-Oriented Requirements	Core Activities: Documentation	Core Activities: Elicitation	Core Activities: Negotiation	Management	Validation
[Gar12]							*Agile*	
[RB12]							*Agile*	
[EBMJ12]							*Agile*	
[SWW11]							*Process Perspective*	
[Kim98]							*Program Requirements*	
[Kim98]							*Detailed project-specific requirements*	
[WS03]				*Evaluation Criteria*			*Prioritization*	
[GR09]	*Collaboration of designers and end users*		✓		*Elicit goals and limitations*		*Prioritization*	
[SWW11]							*Prioritization*	
[Kim98]	✓	✓	✓	✓			**2-by-2 Prioritization Grid**	
[IBM09]			✓	✓			**-Prioritization Matrix -Risk Mgmnt Plan -Change Mngmt Plan**	
[SWW11]				✓			*Traceability*	
[IBM09]				✓			*Traceability*	
[PdC03]			✓	✓			**Traceability Matrix with Cross-Reference**	
[PdC03]			✓				*Traceability and Change Management*	
[PdC03]							*Tools for Change Management*	
[PdC03]				✓			*Requirements Management Control*	
[GR09]							*Integration with other systems*	✓
[PdC03]			✓	**Data Warehouse Specification**	✓	✓	*Requirements Iteration*	

Figure 3.16: Management Cross-Sectional Activity - Proposed Approaches in Literature

The main goal of this activity is to observe the system and detect changes, manage the execution of the requirements and manage the requirements artifacts. Figure 3.16 shows the literature study of Requirements Engineering in Decision Support Systems. During the last decades, Decision Support Systems projects had an increasingly volatile business environment, meaning that organizations are demanding for 'lighter weight,' quicker and an agile software development process that can cope with the ongoing changes of the requirements during the project's lifecycle (Gar12). Agile requirements evolution is used due to its simplicity to change requirements and evaluate the consequences of the changes (EBMJ12). During the past years, according to (RB12), practitioners agree that Agile style development is most appropriate for BI systems, plus it has been the main approach used for a long-time. Figure 3.16 shows that "Agile," "Process Perspective" and "Program Requirements" expand across the complete requirements engineering life-cycle; reason why we have added a discontinuous lines across the Requirements Artifacts and the Core Activities.

The management of the requirements engineering activities mainly focus on the change management tools, traceability of requirements, the prioritization, and defining the level of refinement of these.

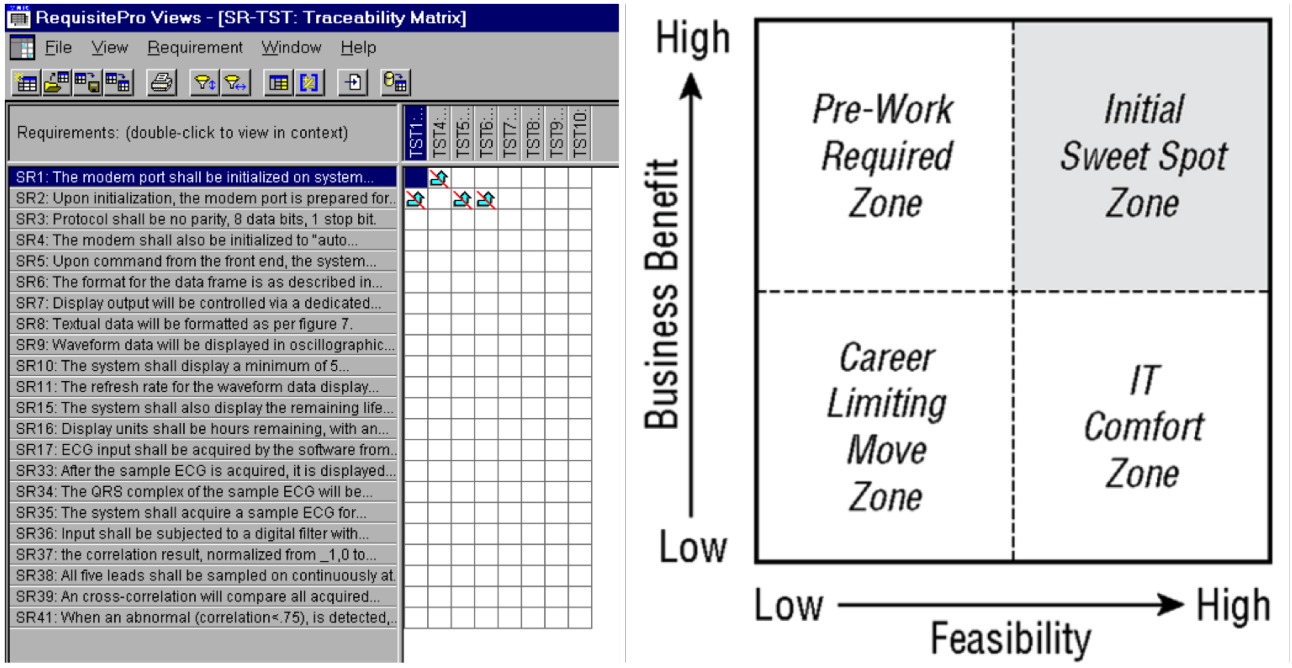


Figure 3.17: Requisite Pro - Traceability Matrix (Left) and 2by2 Priority Matrix (Right) based on (Kim98) and (Woo97)

- **Control Phase and Management Tools:** According to the work presented by (PdC03), "Requirements Management Control Phase" is done where all requirements are traced, refined and prioritized in accordance to the evaluation criteria. The authors emphasize that every change done will affect the database model in the Data Mart and the Data Warehouse visions; therefore most of the relational databases offer different tools (e.g. Microsoft SQL Server and Oracle 9i Developer Suite) in order to manage the affected attributes and discover the elements that were influenced with the change. In addition to the change management tools, the authors claim that all elicited requirements from the Data Warehouse or Data Mart usually traverse a several iterations, negotiation with stakeholders, registration and confirmed finally to a "Management Plan Document" which is then transferred to the "Data Warehouse Specification".

- **Requirements Traceability:** Traceability refers to having a change of history recorded (SWW11). In Data Warehouse environments, traceability must be carried out during the requirements phase in order to be able to measure what impact the change may have over the multidimensional design (PdC03). This last author believes that one of the main models used to achieve traceability is the "Traceability Matrices" to show the requirement's dependencies and provide cross-referencing between the different requirements (ex. Functionalities vs. facts; facts vs. dimensional attributes). Requisite Pro is a tool that aims to keep the traceability of the requirements (see figure 3.17). (IBM09) proposes diverse keys for requirements traceability such as: Unique Numbering, Referred To During Project, Key To Testing, Cross Referencing, Easy to Search For.

- **Prioritization of Requirements:** Prioritization of information requirements is an essential feature. Since there are limited resources that allow only selected requirements to be covered by the Data Warehouse system, evaluation criteria are firstly developed in order to assign priorities of what should be done first (WS03). (Kim98)'s work proposes to use the Two-by-Two prioritization grid in order to prioritize the requirements where the X-axis represents feasibility and Y-axis represents the potential business impact (see figure 3.17). In addition, the same work suggests that when prioritizing the requirements and the implementation process, the grid points that the requirements on the upper-right must be tackled in the initial phases of the development.

- **Requirements Level Refinement:** According to (Kim98), for the Data Warehouse / Business Intelligence requirements analysis there are two passes, first to gather the high-level program requirements followed by a project-specific requirements which are expressed in more details. These requirements may always be refined during the requirements iteration, and changes must be traced.

A summary of Pohl's framework instantiation with regards to management may be seen in figure 3.18.

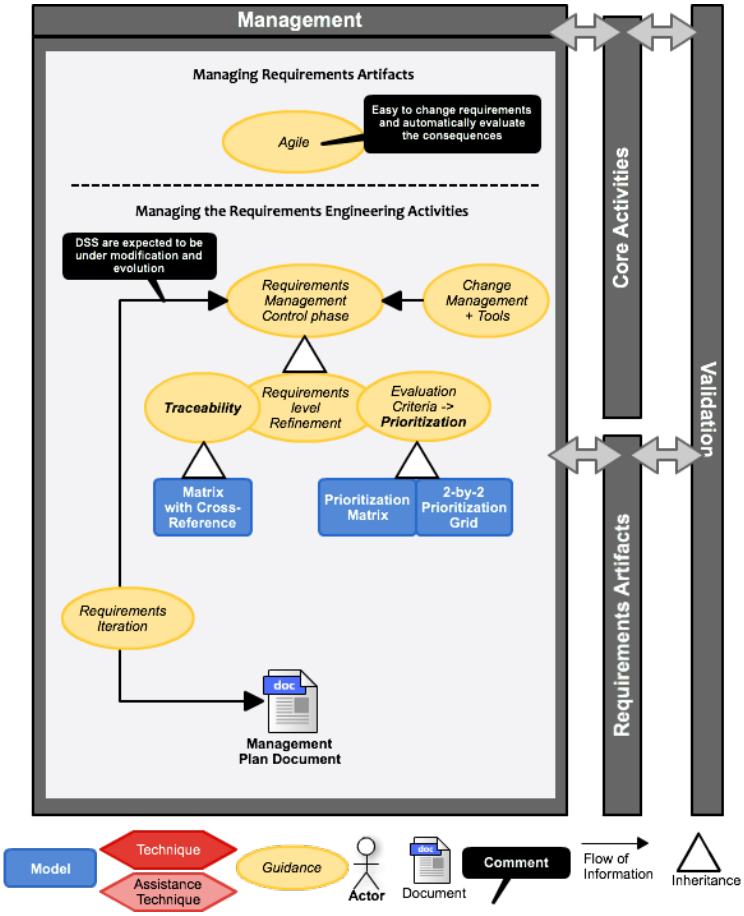


Figure 3.18: Management - Pohl's Framework Instantiation for DSS

Validation:

	Requirements Artifacts			Core Activities				
Investigated Literature	Goals	Scenarios	Solution-Oriented Requirements	Documentation	Elicitation	Negotiation	Management	Validation
[PdC03]			✓	**Data Warehouse Specification**				Requirements Conformance
[PdC03]		**Use Cases**		✓		*Reusability of the agreed knowledge*		Requirements Conformance
[SWW11]				✓				Prototype
[SWW11]				✓				Prototypes
[PdC03]				✓				Review Sessions + Prototyping
[PdC03]				✓				Prototypes
[PdC03]				✓				OLAP interface Prototypes
[SWW11]				✓				Interviews
[PdC03]				✓				Requirement's Checklist
[IBM09]				✓				-Quality Review -Peer Review -Customer Review -IT Review -Phase Gate -Project Sponsor -Requirement Presentation
[MLT08]			✓	**Conceptual Model**				**Normalization Process**
[PdC03]				✓				*Guarantee Summaizability*
[PdC03]				✓			✓	*Validate Derived Data*
[PdC03]								*Add external reviewers to validation process*
[PdC03]				*List of actions*				*When problems detected return*

Figure 3.19: Validation Cross-Sectional Activity - Proposed Approaches in Literature

Validation process is done in order to determine whether the input and output fulfil the quality criteria answering the question: Am I building the right system? whereas verification answers the question: Am I building the system right? Figure 3.19 shows the literature study for Requirements Engineering in Decision Support Systems.

- **Validation:** The first proposed manner to validate the activities by the literature is by involving external reviewers to the validation process as a means of giving an unbiased point of view about the Data Warehouse (PdC03).

- **Verification:** The second proposal is verifying the data is using the normalization process. This process is done with the main aim to obtain a multidimensional model that ensures summarizability (MLT). According to (PdC03), 'Summarizability' refers to guarantee the correctness of the aggregated results for combining facts and dimensions. In addition, the same work proposing the use of 'Summarizability' also states that Data Warehouse application must also verify the

derived data. The output of the normalization process is a multidimensional model constrained to the business rules and relationships that do not violate summarizability (MLT).

Moreover, diverse techniques are proposed in order to validate the activities done during the requirements analysis (notice from the checkmarks that all these must be documented):

- *Prototype:* Validation of the requirements may be carried out through the creation of a prototype (SWW11). For instance, the validation of the multidimensional model (solution-oriented requirements) may be carried out by an understandable prototype for the stakeholders, such as visualization graphs.
- *OLAP Prototyping:* These prototypes help to simulate OLAP behaviour for stakeholders to corroborate their ideas before being allocated to the final multidimensional solution. (PdC03)
- *Review Sessions:* According to (PdC03), review sessions combined with prototyping are very effective to identify and remove defects in the system before becoming part of the final Data Mart package delivered to the stakeholders.
- *Interviews:* Validation of the documented information requirements may also be done with interviews of IT and business staff (SWW11).
- *Data Warehouse Requirement's Checklist:* (PdC03) defined a checklist for Data Warehouse requirements, tailoring specific questions regarding aggregations, facts, documentation quality, completeness, etc., (see figure 3.20).
- *Other techniques include:* Quality Review, Peer Review, Customer Review, IT Review, Phase Gate, Project Sponsor and Requirement Presentation (IBM09).

Item	Description
Automatic aggregation	Do all dimensional levels lead to a complete summarizability approach, in terms of the multidimensional model elaborated?
Facts and dimensions representation	Are all stakeholders' analytical needs represented in terms of a multidimensional schema?
Facts and dimensions connection	Is the entire set of dimensional levels properly associated in all levels to the basic set of facts being analyzed?
Integration completeness	Are all integration requirements and procedures defined as to correctly incorporate external information from source providers?
Documentation quality	Do all defined documents serve as tools to accomplish all user needs under established quality standards?
Requirements conformity	Can we truly "drill" across fact tables by navigating through conformed dimensions without incurring in data loss or inconsistence?

Figure 3.20: Requirements Checklist for Data Warehouse, based on (PdC03)

When problems are located during the validation process, immediately the validation team attaches a document with the document 'list of actions' to each of the errors or defects identified (PdC03). Afterwards, the development process shall return

to the specification phase (e.g. Core Activities, Requirements Artifacts) in order to solve the issues. A summary of Pohl's framework instantiation with regards to validation may be seen in figure 3.21.

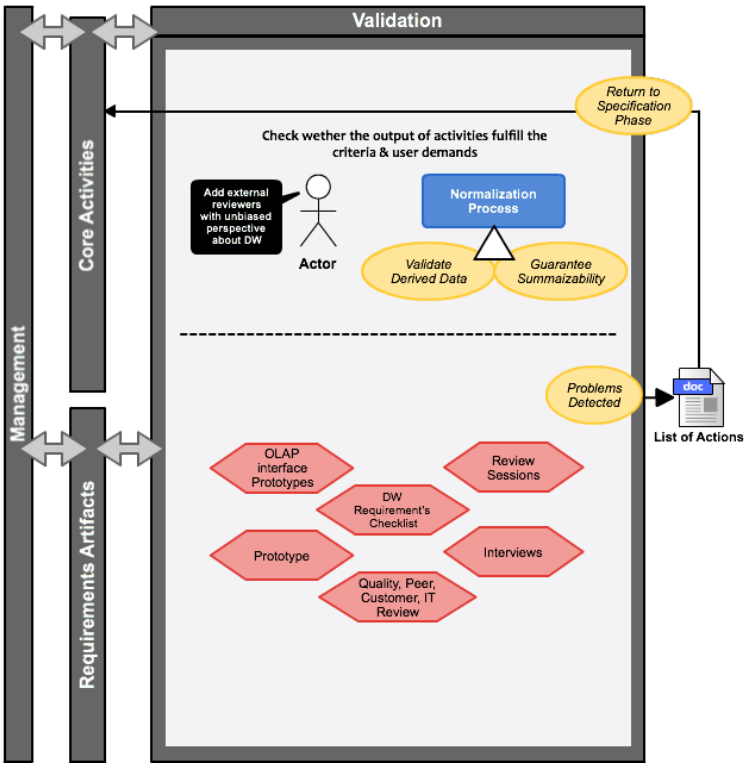


Figure 3.21: Validation - Pohl's Framework Instantiation for DSS

3.4 Complete Graphical Representation

Finally, the proposed models, techniques and guidance were adhered to Requirements Artifacts, Core Activities and Cross-Sectional Activities to create the complete vision of the literature in the framework. Figure 3.22 presents the complete representation of the Core Activities and Requirements Artifacts practices from the Decision Support Systems Requirements Engineering literature. While figure 3.23 presents the Management and Validation practices. These instantiations will be used in the following two chapters in order to propose our approach and subsequently validate it in the Chagas case study.

As seen in both figures 3.22 and 3.23, Models are represented in blue, techniques in red and guidance in yellow. Documentation is taken throughout the complete process ending with the "Data Warehouse Requirements Specification" containing all the details for designers to start the implementation process. It is crucial to note that arrows between the Core Activities and the Requirements Artifacts are iterative, meaning that it is possible to go back and forth during the activities. In addition, the two cross-sectional activities also affect all the process for instance, when the conceptual model is ready, it can be validated using interviews with the designers and business users in order to verify everything is according to the stakeholder's needs. Although if problems are detected in the validation phase, then the team must return to the specification phase.

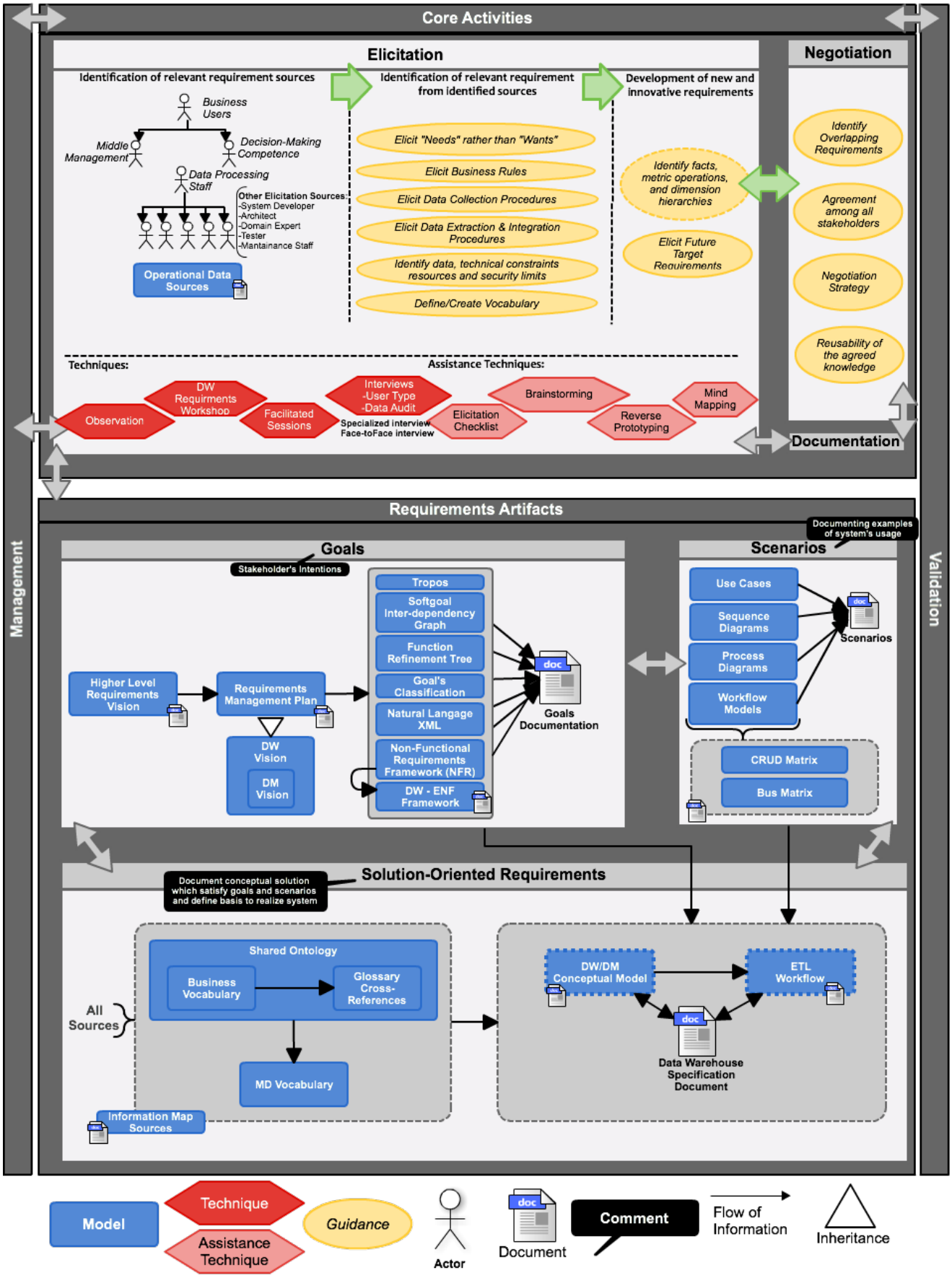


Figure 3.22: Pohl's Framework Instantiation for Decision Support Systems

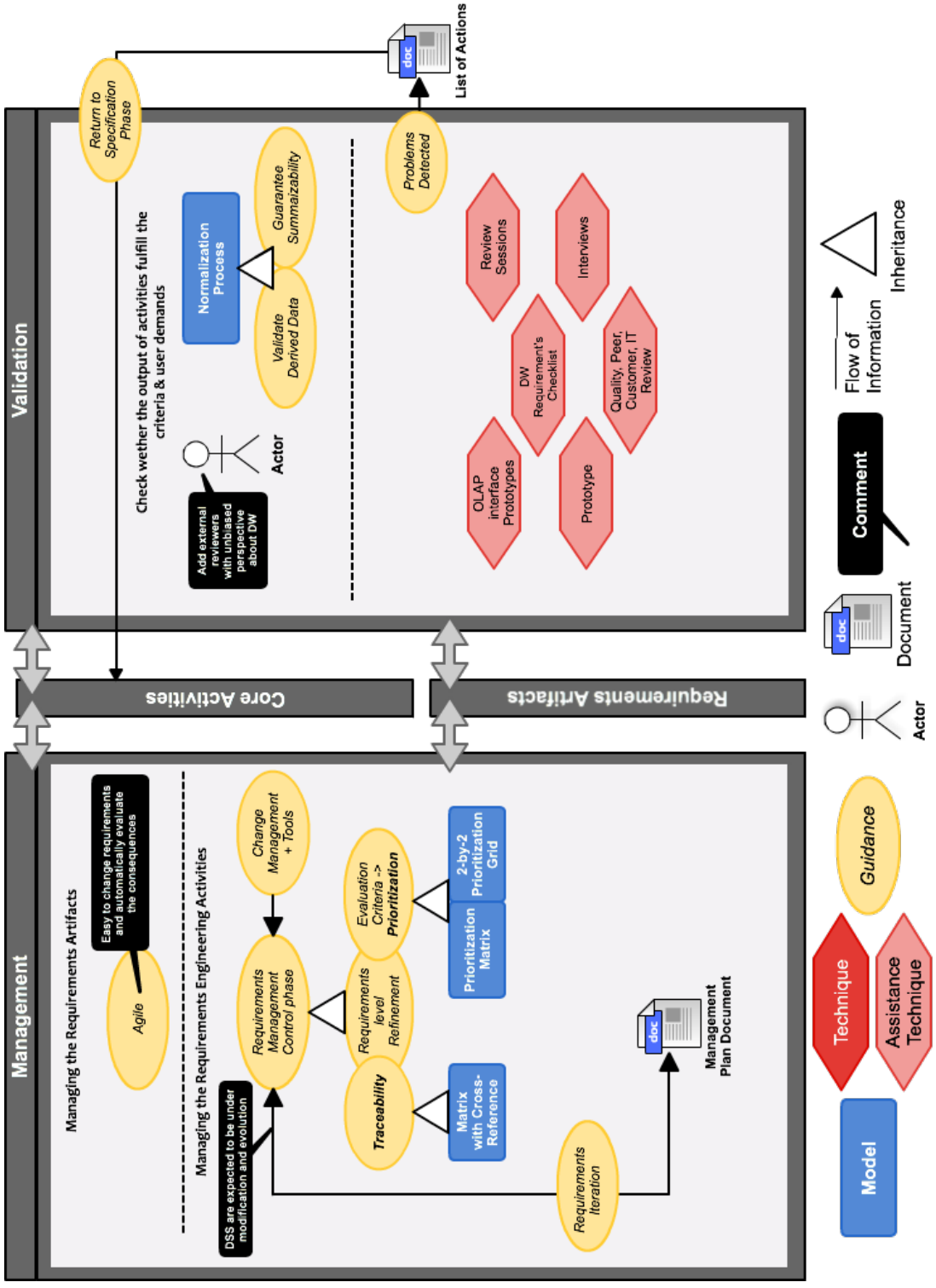


Figure 3.23: Pohl's Framework Instantiation for Decision Support Systems

3.5 Chapter Conclusions

This chapter at hand consolidates the literature classification of Requirements Engineering for Decision Support Systems. The chapter lend support to claim that that leading Information Systems professionals lack of an adequate approach to determine information needs specifically for Decision Support Systems (see chapter 4).

Some of the points worth to highlight from the literature classification are the following:

- From the information gathered in the literature, it is suggested that during the **elicitation** three relevant sources are usually identified: the business users, the data processing staff (IT), and the data sources. A closer look at the literature indicates that this last source in Decision Support Systems may need to integrate diverse heterogeneous sources; hence, this demands to establish several procedures to control how the data is collected and transformed.

- During the **elicitation**, in Decision Support Systems it is not easy for stakeholders to express their requirements. In response to this, business needs must be extracted by asking, "what do you do and why?" and not asking "what do you want in your system?". From these elicited requirements, we may start identifying the needed multidimensional concepts (e.g. facts, metrics, and dimension hierarchies) to be included in the Decision Support System.

- As it may be seen in this chapter, the literature shows that **negotiation** activity is addressed by relatively few approaches and usually consist of the same activities like any traditional software system. Nevertheless, what does actually change is the concept of the conflict (i.e. stakeholder's may or may not agree on the granularity of the data, facts, dimensions needed).

- According the literature, the **documentation** activity in Decision Support Systems enables the achievement of requirements to be consistent and traceable over the entire project.

- The available literature seems to propose plenty of models to document **Goals** for Decision Support Systems, such as: Tropos, Softgoal-Interdependency Graph, Function Refinement Tree, Goals' Classification, etc. When documented, these must be checked in accordance to the specified business rules. However, there is a lack of grasping a higher-level requirements vision during the design of a Decision Support System.

- Several authors propose the use of **scenarios** by using Use Cases (just as any software system). Other alternatives include sequence diagrams, process diagrams, workflows, etc. Although, Use Cases are traditionally used in any software system and not just in Decision Support Systems.

- One of the most addressed failures in Decision Support Systems presented in the literature is the existing poor communication between the business and the IT users. Given this, the literature proposes the use of a vocabulary since the early start of the project aiming to homogenize terms and provide better communication between IT and business users; whereas the glossary provides cross-references between the terms (e.g. synonyms). This process is mainly done manually, although automation may be considered if the requirements are formalized in machine-readable format. Hence, the glossary may be later transformed into a shared ontology (super-set of vocabulary) to automatize the creation of the

Solution-Oriented Requirements by building the multidimensional vocabulary which is illustrated multidimensional concepts (e.g. facts, dimensions, measures) and their relationship among them.

- In contrast to other systems, according to the literature the final step in the **Solution-Oriented Requirements** of Decision Support Systems a Data Warehouse and ETL Conceptual Models are built out of the all the requirements given by the stakeholders. However, business needs are collected (e.g. interviews, etc.) to be then manually translated into a conceptual MD and ETL Designs; this manual process tends to be error-prone, that demands several rounds of redesign to satisfy all the business needs stated by the stakeholders. Hence, this process may be automatized in such a way that from obtaining the requirements in a structured manner, we may automatically create the respective MD and ETL design considering all the needs (more details available in chapter 4).

- During the **management** activities Decision Support Systems, we must consider the traceability and prioritization of requirements. In addition, the literature proses several techniques to **validate** the outcome, prototypes, OLAP prototyping, and Interviews may be carried out. Another way to validate the data is using the normalization process ensuring summarizability and guarantee the correctness of the aggregated results.

Part III

Contributions

CHAPTER

"The goal is to turn data into information, and information into insight."
– Carly Fiorina

4 RE4DSS: A Systematic Approach of Requirements Engineering for Decision Support Systems

Decision Support Systems have emerged as a dominant technology capable of integrating heterogeneous sources into an analytical fashion to facilitate and provide better decision-making process. As described in the previous chapter of this thesis, the development of these systems is rather different from the development of traditional operational systems in charge of supplying data to the warehouse. Particularly we mentioned that, Decision Support Systems are expected to: deal with diverse domain terminology causing poor communication between business users and IT people, integrate heterogeneous sources demanding complex procedures to control the integration and transformation phases, produce high-quality documentation, create analytical results expected to be explored across multiple levels (e.g. dimensions), provide a validated and verified output, provide traceability of sources; and a comprehensible MD Design is mandatory. In addition to the aforementioned characteristics, at the early stages of Decision Support System projects, typically, business needs are manually collected (e.g. interviews, etc.) to be then translated into a conceptual MD and ETL Designs; this manual process tends to be error-prone that demands several rounds of redesign to satisfy all the business requirements. On the basis of the evidence currently available, it seems fair to suggest that leading Information Systems professionals lack of an adequate approach to determine information needs specifically for Decision Support Systems.

For this reason, in this chapter we propose a systematic approach of Requirements Engineering tailored to Decision Support Systems: we have named it "RE4DSS" *(Requirements Engineering for Decision Support Systems)*. Our block-oriented approach *RE4BI* aims to improve these manual practices by systematizing the requirements process, and accompanied by a system called GEM (Generating ETL and MD designs) we provide a semi-automatic means translating each of these requirements to finally produce a validated and verified ETL and MD designs (more information regarding the GEM may be found in Solution-Oriented Requirements Section). Fundamentally, our iterative, block-oriented approach, RE4DSS, is based on the Requirements Engineering framework blocks proposed by Klaus Pohl and is coupled with a set of models and techniques, guiding the developer throughout the Decision Support System Requirements Engineering process.

To pursue our main goal, we have used the literature classification from the previous chapter to address the complexities of Requirements Engineering for Decision Support Systems. Our motivation to create this approach is found in the following

chapter which describes the contributions of our approach to the specifications of a Data Warehouse project conducted by the WHO to create an information surveillance system to control/eliminate Chagas disease (see chapter 5). The following sections describe our approach starting with a complete overview in Section 4.1. Subsequently each of the elements in our approach is mapped to our reference framework: Section 4.2 focuses in the Core Activities, Section 4.3 in Requirements Artifacts, Section 4.4 includes the description of the GEM system and its role for the Solution-Oriented Requirements plus the Validation and Management activities in our approach.

4.1 RE4DSS Approach Overview

This approach, was created based on the practices from our literature study done on chapter 3. As it may be seen in figure 4.1, RE4DSS employs three *Core Activities*, namely: elicitation, negotiation and documentation. **Elicitation** starts with the main sources, business users (which include the middle management, decision-makers), the data processing staff and in addition, all the source data stores. **Negotiation** is done in order to avoid conflicting ideas within the stakeholders and **documentation** activity is done throughout the complete approach, especially during the Requirements Artifacts. According to the literature, elicitation has been found to be the most distinguished activity for Decision Support Systems out of these three, while documentation and negotiation activities tend to resemble more like traditional software systems. These activities are used as a base in order to elicit, negotiate and document the Requirements Artifacts.

These artifacts are: Goals, Scenarios and Solution-Oriented Requirements. One of the most important tasks of elicitation activity is to elicit the **goals** starting from the high-level requirements. Goals can be modeled and refined by using a "Goal Classification" tree. This Goal Classification is divided into three levels: strategic, decision and information goals. At the lowest level, information goals (requirements) must be then classified as one of these three: *Exploration Requirements*, *Domain Requirements* or *Quality Properties*. *Exploration Requirements* are the functional requirements used at the very end of the process for future tuning and to enrich the Data Warehouse (e.g. visualization dashboards). *Domain Requirements* (other functional requirements) and *Quality Properties* (also known as non-functional requirements) are then transformed into Structured Requirements in XML format to later enrich the operators from the ETL Design. In addition, from this last branch (information goals) we also extract the **scenarios** (Use Cases) that come from the *Exploration Requirements* and the *Domain Requirements* (functional requirements).

One of our main contributions for our approach tailored to Decision Support Systems, is using a Vocabulary aiming to homogenize the terminology involving the poor communication between business users and IT. However, this document cannot be processed by the system; consequently, we propose the use of Shared Ontology being in a machine-readable format. Within this Shared Ontology, the sources are mapped into an Ontology Source (we have proposed the use of OWL). From this, the Domain Ontology is created and enriched with the Vocabulary document extracted from the Use Cases. This Vocabulary is iteratively updated at every step in our approach and the final stage is sent to the final **documentation**.

With this, all three machine-readable inputs for the GEM are created, namely: the Domain Ontology, the Source Mapping, and an XML file containing Structured Requirements (read more about GEM in Solution-Oriented Requirements section).

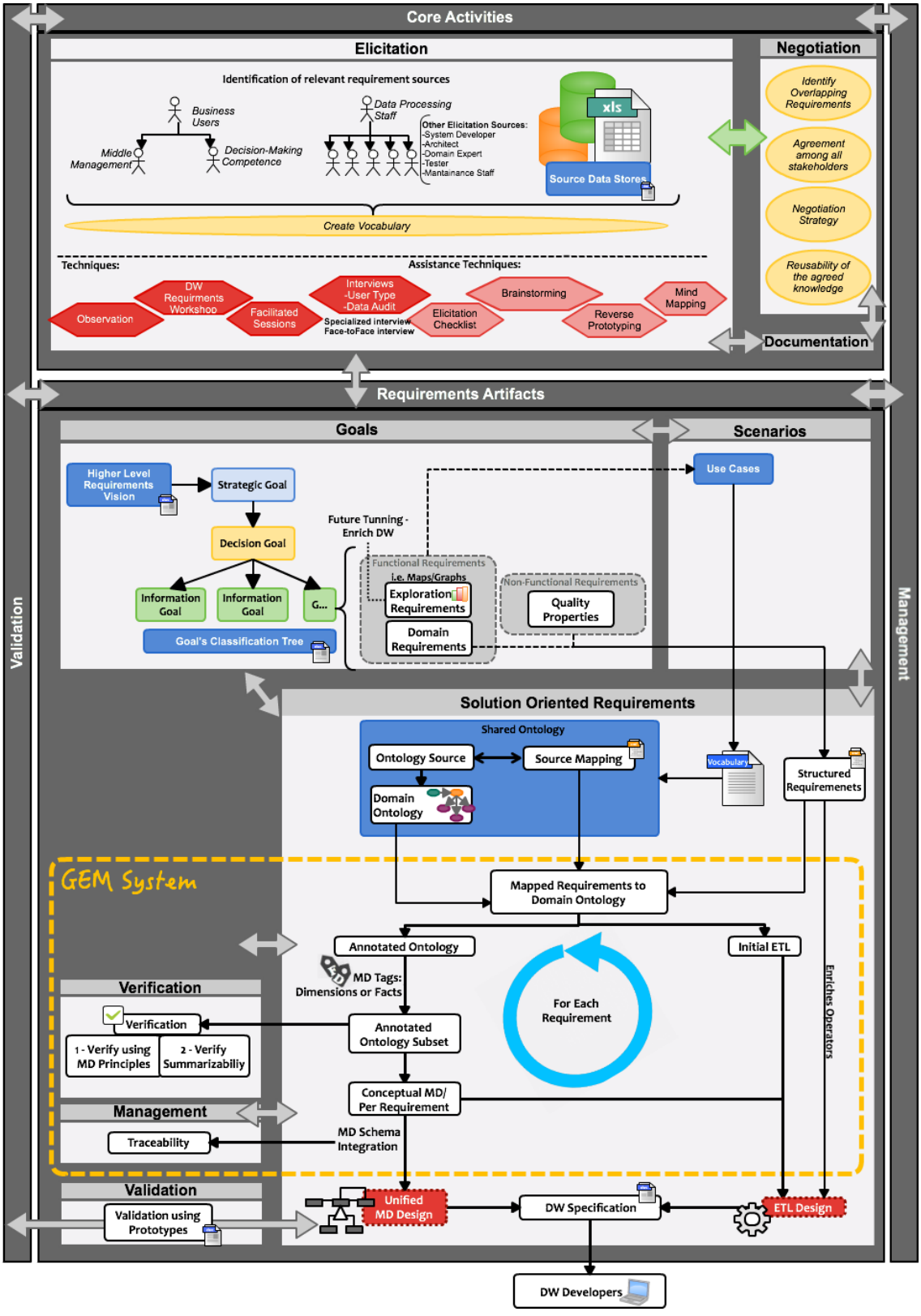


Figure 4.1: Requirements Engineering 4 Decision Support Systems (RE4DSS) Approach

In the **Solution-Oriented Requirements** (see yellow delimitation line in fig 4.1), the GEM system takes action and starts by mapping the requirements to the Domain Ontology. When everything is mapped, this is then divided into two-paths, one to create the ETL Design (right) and one to create the Conceptual MD (left). On the creation of the Conceptual MD side, an Annotated Ontology is created where concepts are classified as dimensional or factual using tagging. Then the result is automatically **verified** based on the tagging according to the MD Design principles and summarization. After being verified by GEM, the individual Conceptual MD from each requirement is then integrated to create a Unified Schema. Is important to note that traceability of the requirements and changes must be recorded during the **management**. Finally, the unified MD Design and the ETL Design are **validated** along with the stakeholders using prototypes. If the verification and validation succeed, the MD and ETL Designs are sent to the final documentation stage: Data Warehouse Specification, which is later automatically used as an input for the DBMS (Database Management System) or ETL tool in order to start the implementation.

The process is explained in more details in the following subsections: Core Activities, Requirements Artifacts, Validation and Management.

4.2 Core Activities:

1. **Elicitation**

   Our systematic approach starts at the very earliest step of a Decision Support System collecting the stakeholder's needs in order to finally translate these into an ETL and MD Design. Our approach suggests that the very first step entitled 'elicitation' is performed using two inputs (see figure 4.2), namely - Business Requirements (extracted from business users and data staff processing) and Source Data Sources:

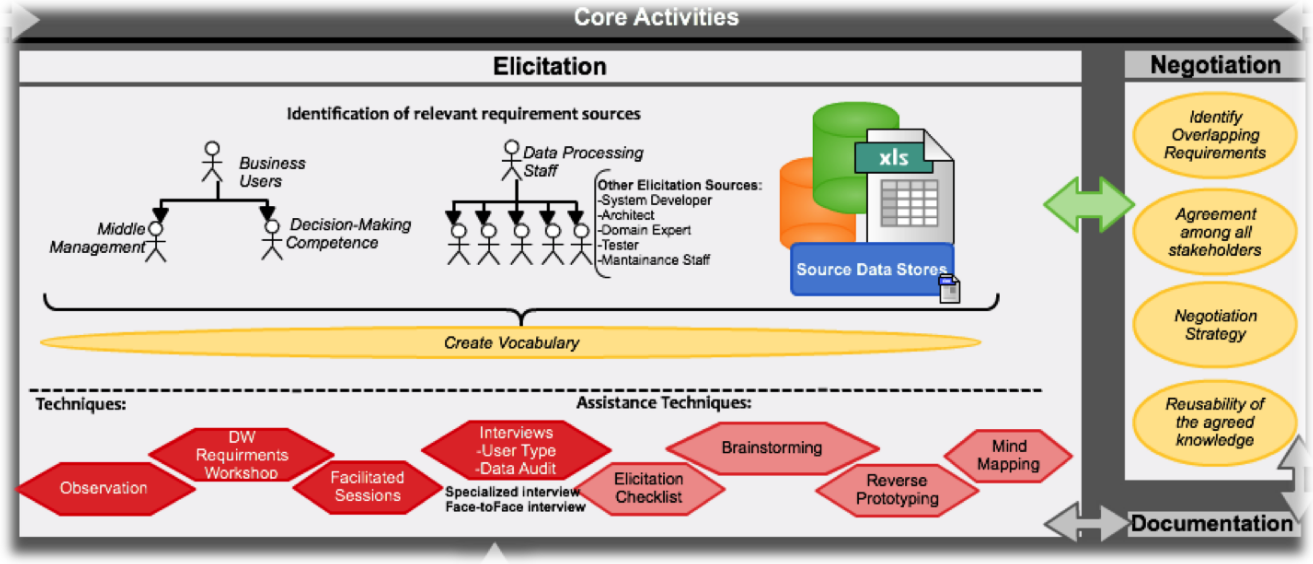


Figure 4.2: Elicitation Process (RE4DSS) Approach

a) **Elicitation of Business Requirements:**

In typical Decision Support Systems, business requirements are the starting step for designing and building the system. The requirements must be fully understood in order to transform these into the appropriate MD model and ETL Design. (Notice that we speak about "needs," and not about what the customer wants).

b) **Elicitation of Source Data Stores:**

Not only requirements from the Business Users are elicited, but there are also Data Stores from where information may also be extracted. These may be identified by interviewing the IT and Data processing staff in the organization for instance. These may include various sources that may be in different formats such as Excel, XML or operational databases. All the available sources must be taken into account; Then in the following steps these will be refined and depending on the user's goals, only certain sources will be selected (see Goals subsection). In addition, the structure of the sources (from where the data is extracted) is transformed and finally is loaded into the target schema (described in Solution-Oriented Requirements).

As mentioned in the previous chapter, some of the techniques proposed by the literature to elicit requirements of Decision Support Systems are the following:

- **Interviews**
- **DW Requirements Workshops**
- **Observation**
- **Facilitated Sessions**
- **Assistance Techniques:** Other techniques that can be used in order to support the basic techniques mentioned above are: Brainstorming, Elicitation Checklist, Mind Mapping and Reverse Prototyping.

Considering that Decision Support System projects involve business people using business terminology and source data stores that contain different terminology given by the data processing staff, a Vocabulary should be created in order to bridge the terminology. This vocabulary is creating at the very initial step of the project and must be updated throughout the complete Requirements Engineering process, including during the creation of goals and scenarios. Vocabulary will be further explained in the Solution-Oriented Requirements.

2. **Negotiation**

   The negotiation is intended to be the solution activity when diverse stakeholders provide different wishes that may be in conflict (see figure 4.2). So, a mutual agreement must be provided among all the stakeholders in order to provide a single integrated solution that encompass all the stakeholder's objectives. Conflict management involves identifying, analyzing, resolving and documenting the conflict. Some of the suggested negotiation techniques are Win-to-Win and Interaction Matrix. Nevertheless, negotiation activity does not vary much in comparison to other traditional software systems.

3. **Documentation**

   Documentation is an important activity to establish a common reference and promote communication among the stakeholders. The main aim of this activity is to document the elicited requirements according to the rules. This activity is crucial when supporting training of new employees and to preserve expert knowledge for future employees. Rules for documentation may be either: 1-General Rules (e.g. define layout, headers, version history), 2-Documentation Rules (e.g. rules to ensure quality of the template for the use of elicitation and negotiation), or 3-Specification Rules (more restrictive than documentation rules e.g. rules for subsequent development activities like the use of requirements specification language). Considering this activity is related to the documentation of the Requirements Artifacts, (for instance Goal's documentation, Scenario's documentation - Use Cases, etc) this activity will be further explained in the Requirements Artifacts section.

4.3 Requirements Artifacts:

1. **Goals**

   By using the different elicitation techniques presented in the Core Activities, we can identify a high-level requirements vision. Due to the lack of a high-level requirements vision during the design decisions, we have included the main aim of our stakeholders, during the elicitation (e.g. interview). Hence, it is crucial to ask: "what do you do (and why)"; and not asking "what do you want" in the system. With this, we extract the main need of the stakeholders. For instance, the CEO may request: *"I need a system to increase the sales."*

   Followed by having the high-level requirements vision, out from all the models proposed by the literature (see the previous chapter for more details), we have decided to use Goal Classification Tree proposed by (vL04) because starting from a high-level vision, it provides a structured manner of classifying the sub-goals in different levels of abstractions that business users expect to fulfil by the envisaged Decision Support System. Another reason of utilizing this model is that from the refinement of goals, we have found that from the last level of the tree (information goals) we may identify additional information about: the needed sources, classify these as different types of requirements for future use, and decide whether to create use cases.

   Having a high level requirements vision, we will then refine this main Goal by using the model Goal's Classification where goals are classified in three kinds depending on the level of abstraction (see figure 4.3):

   a) *Strategic Goal:* This goal represent the highest level of abstraction being the main objective of the Decision Support System. This is defined as the changes from the current situation into a better situation. An example of a strategic goal is the one given in the previous example "increase sales."

   b) *Decision Goals:* These goals are the subdivisions of the strategic goals. These are the medium level of abstraction intended to answer the question "how can the defined strategic goal be accomplished?". Some examples would include "launch new product" or "open new store in a different city".

   c) *Information Goals:* In the last subdivision level of the hierarchy (leafs), we try to answer the question "how can the decision goal be accomplished?". Examples may include "Import customer purchases," "Examine stock from suppliers," "Visualization sales maps," "Data in sales data mart must be updated every 2 seconds".

   Is important to note that Business Rules from the organization must also be taken into account at this step in order to produce a better result when defining the goals. After the Goal Classification Tree is created (see figure 4.4), the Information Goals are the point of reference to start the classification of the requirements, that can be either a functional requirement (which includes *Domain Requirements* and *Exploration Requirements*) or *Quality Properties*. Later, for each of the functional requirements, a Use Case must be designed.

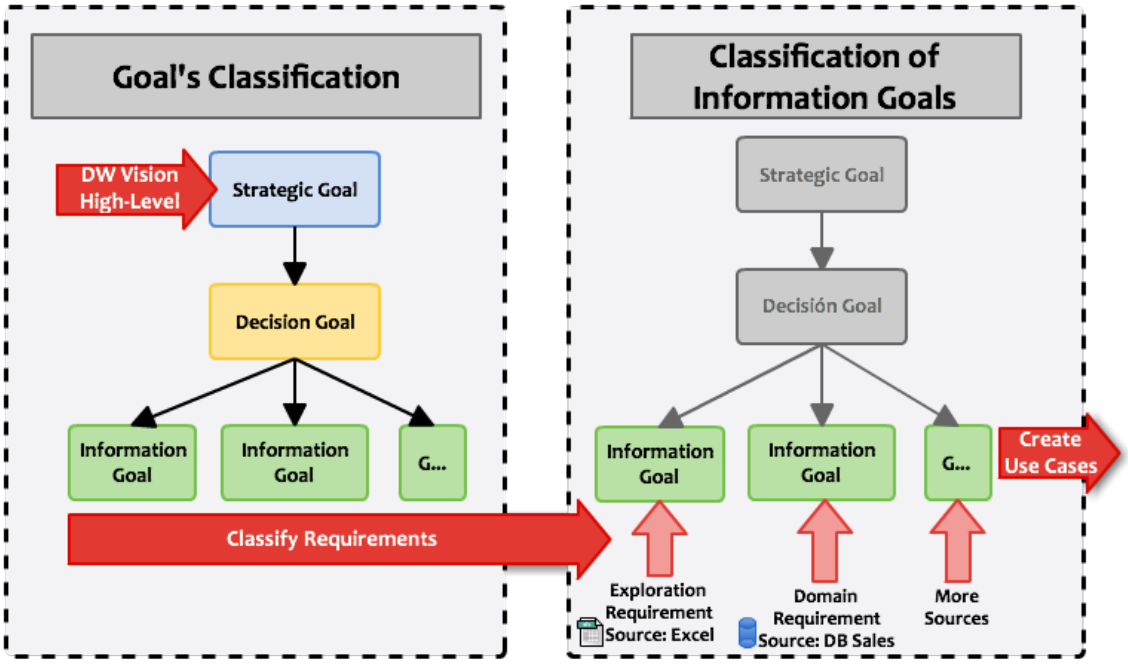


Figure 4.3: Goal Classification Tree Levels - Extracting Information Goals and Creating Use Cases

Each of the Information Goals must be classified into one of these definitions (see figure 4.4). (Is important to note that the following definitions of the requirements terminology are out of our study's scope; these definitions are only used to specify the different outcomes of the Use Cases):

- *Domain Requirements:* This kind of requirements specify essentially what the system should do (also known as functional requirements). These requirements usually specify a behaviour or function. Although *Domain Requirements* exclude any *Exploration Requirements*. Considering the examples provided in the Goal's Classification, a domain requirement would be: "Import customer purchases"and "Examine stock from suppliers".

- *Exploration Requirements:* Any domain requirement that requires direct user's interaction such as visualization. Considering the examples provided in the Goal's Classification, an exploration requirement would be: "Visualization sales maps."

- *Quality Properties:* The *Quality Properties* specify how the system should behave as a constraint upon the system's behaviour (notice that quality properties are also known in the literature as non-functional requirements, however we have decided to follow the same terminology as (Poh10)'s terminology. In this document, quality properties refer to the quality attributes of the system, for instance scalability, and data integrity. Considering the examples provided in the Goal's Classification, a quality property would be: "Data in sales Data Mart must be updated every 2 seconds."

The final result of the classification of the requirements are the following (considering that requirements are expressed in

a structured way): *Exploration Requirements* such as visualization will be utilized in the future for tuning and enriching the Data Warehouse. Example of these may include the way stakeholders want to visualize and interact with the data for decision-making. *Domain requirements* and *Quality Properties* are collected and then expressed as Structured Requirements in an XML file where multidimensional concepts are identified (measures and its dimensions).

In summary, the three outputs of the Goal's Classification Tree are the following (see outgoing arrows from figure 4.4):

- Identifying the needed sources from the elicited sources using the lowest level (information goals) of the Goal's Classification Tree. For instance, if the information goal states the following "Examine stock from suppliers," then we will need to utilize the source that contains the complete stock from the company. This may be a database, excel file, or any other type of source.
- Begin the classification of *Quality Properties*, *Exploration Requirements*, and *Domain Requirements*
- From each of the information goals (except *Quality Properties*) it creates an input for the scenarios (Use Cases). (See the section Scenarios for more details).

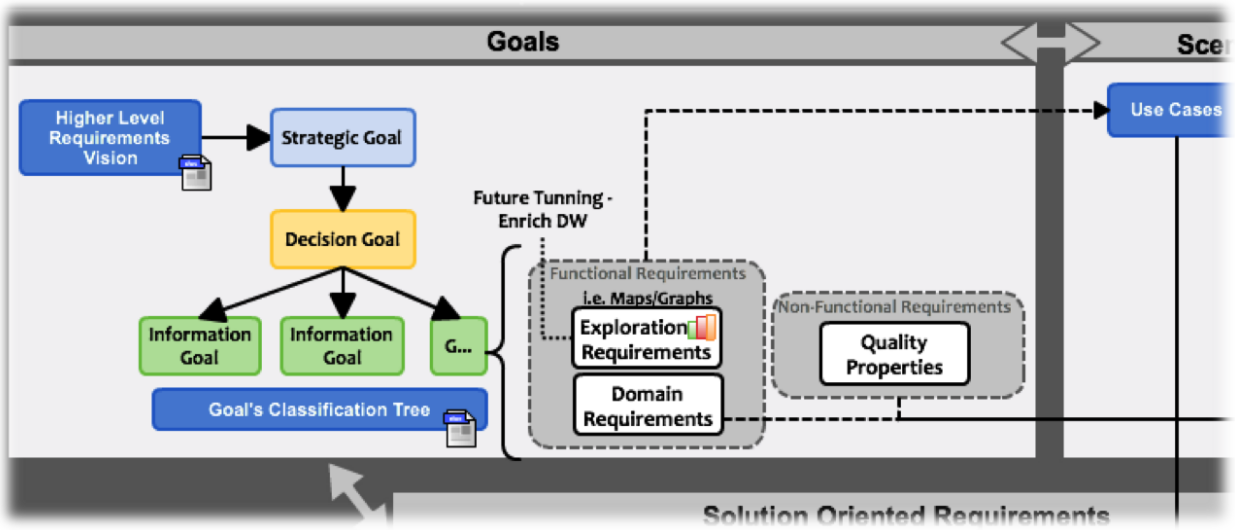


Figure 4.4: Goal's Classification Process and Scenario Use Cases - (RE4DSS) Approach

2. **Scenarios**

After creating the Goal's Classification Tree and classifying the information goals (leafs), from each one of the functional requirements (*Exploration Requirements* and *Domain Requirements*) we now create a Use Case (see figure 4.3). Given the complexity of Decision Support Systems, we have used Use Cases because these allow to describe complex systems systems in an easier and understandable manner by simply pointing what the system is intended to do for the business users. In addition, Use Cases directly relate to the negotiation activity, because these encourage to achieve a common and documented agreement between the system's requirements, and identify which parts from the proposed system's manual processes may be automated. It is possible to use a reference template to document the Use Case

including attributes in columns (ID, name, author, version, change history, goals, actors, precondition, post-condition, etc.). Straight from the Scenarios, the Vocabulary will be complemented with new concepts/terms identified in the Use Cases in order to enrich our Shared Ontology (see example in chapter 5).

3. **Solution-Oriented Requirements**

Goals and scenarios are the basic foundations for developing Solution-Oriented Requirements. By joining together these three together, we may define the foundation and reference for the developers to start implementing the system.

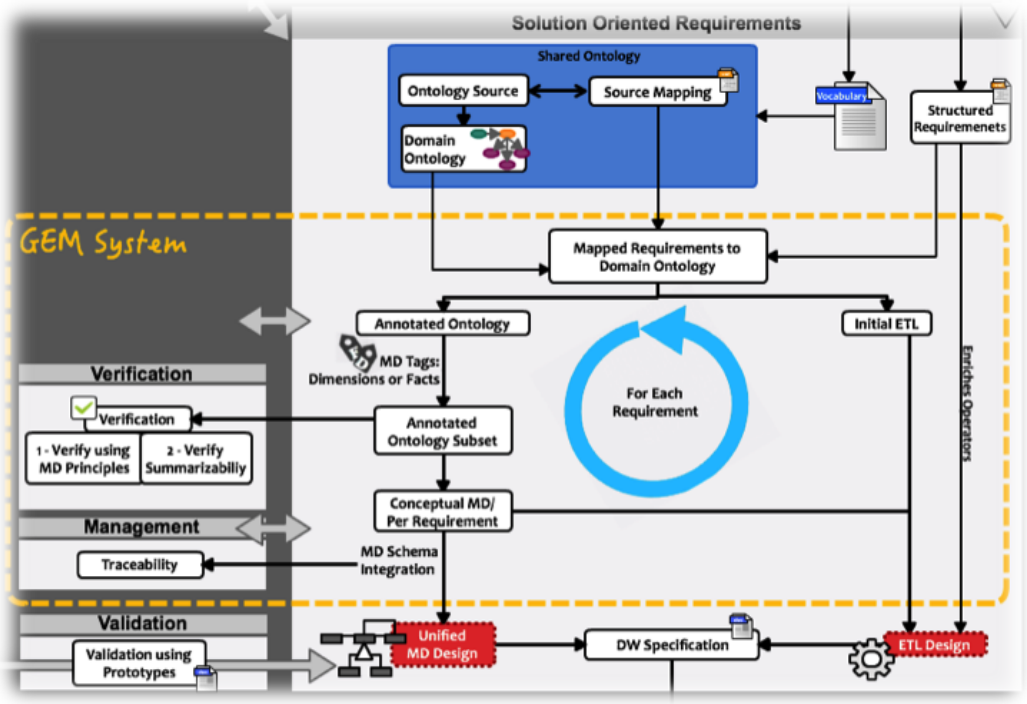


Figure 4.5: Components of the Solution Oriented Requirements - (RE4DSS) Approach

In this section, we will explain the different components (see figure 4.5) in the Solution-Oriented Requirements of our approach. These are organized in the following subsections:

a) Structured Requirements

b) Shared Ontology

   i. Vocabulary

   ii. Ontology Source & Domain Ontology

   iii. Source Mapping

c) GEM System

   i. Mapping Requirements to Domain Ontology

ii. Annotated Ontology Subset

iii. Validation & Verification

iv. Management

d) Final Output - Unified Schema and ETL Design (iterative)

In order to complete the three inputs to automatically generate from the elicited requirements the respective ETL & MD Designs using the GEM system, we must obtain the following first two bullets, namely: (1) The Structured Requirements (XML format) and the Shared Ontology (composed of a (2) Domain Ontology and the (3) Source Mappings). These are explained in details in the following sections.

Structured Requirements

This first input, Structured Requirements, are obtained from the *Domain Requirements* and the *Quality Properties*. Considering that requirements are expressed in a structured way, *Domain requirements* and *Quality Properties* are collected and then expressed in an XML file where multidimensional concepts are identified (measures and its dimensions). Dimensions being the perspectives used to analyze the data (view data in a specific level of detail). Measures being a summary of the analyzed data. For instance "examine income per customer." An example of the extract of the XML is shown in figure 4.6. The first two examples in the figure relate to the dimensions and measures of the *Domain requirements*. In addition, the third example in figure 4.6 relates to the *Quality Property* where the freshness that requires that the corresponding revenue is updated at least every hour. Other examples may include: Scalability, Availability, Reliability, Data Integrity, etc.

```
<dimensions>
    <concept id = "customers"/>
        <dimensions>

<measures>
    <concept id = "income"/>
<measures>


<measures>
    <concept id="revenue">
          <q_requirement> p_priceATRIBUT * p_discountATRIBUT</q_requirement>
          <nfr kind="freshness" format="HH24:MM:SS">&lt;01:00:00</nfr>
    </concept>
</measures>
```

Figure 4.6: Domain Requirments and Quality Properties illustrated as Structured Requirements in XML

Shared Ontology

In this section, we will focus on explaining the second and third inputs (Domain Ontology and Source Mappings) for the GEM system which reside inside our Shared Ontology. One of our main contributions in RE4DSS is having both, a Vocabulary document and a Shared Ontology. The Vocabulary document provides just a wordbook to enhance the communication between the business users (domain users), stakeholders, IT people, and the sources. In comparison with the Vocabulary, the Shared Ontology is not a document, but a machine-readable format providing an integrated view of a domain terminology and data that describes any data source; the shared ontology provides a cross-reference structure including the data stores, and must also maintain the traceability relationships. Overall, the main and most important difference between these two is that the Vocabulary is just a wordbook documentation (unable to be read by any system) which is used to enrich with new terms the Shared-Ontology that indeed is specified in machine-readable format being an input to the GEM system (see figure 4.5). These components are all explained in the following subsections.

a) **Vocabulary**

The output of the Use Cases is used to complement the created Vocabulary, which will later enrich the concepts in the Domain Ontology. Since there are business people involved in the decision-making process which certainly use business terms for their everyday communication, we must expect that the input business requirements are also expressed using this business terminology, but from the other side, the sources created by the Database specialist do not match the same terminology. Consequently, we found a need to bridge this gap between the business and the IT terminology (see figure 4.7).

The Vocabulary is intended to avoid linguistic inconsistencies between stakeholders and IT people, the desire is expressed to homogenize used terms (e.g. dimensions, facts, summarizability, etc), provide definitions, acronyms, abbreviations, etc. All these must be part of the documentation along with the business rules presented in the goals section. In summary, the Vocabulary is utilized to define the terminology and enrich the corresponding ontology (Domain Ontology).

b) **Ontology Source**

After identifying the available data sources and selecting the sources needed for the goals to be achieved by the use of Goal's Classification, now we are ready to represent the structure of the sources from which the data will be extracted, transformed and loaded into the target schema. The semantics and the constraints of the selected source are captured in terms of an ontology (see figure 4.8):

An ontology is shortly defined as the specification of a conceptualization. The ontology proposed to be used in our approach is the Web Ontology Language (OWL). This is used to explicitly represent the meaning of the terms in vocabularies and the meanings of those terms. This ontology is intended to be used when the information in the documents need to be processed by applications (automatic processing of information). The process of construction of an ontology is also led by the enrichment of the Vocabulary which is extracted from the Use

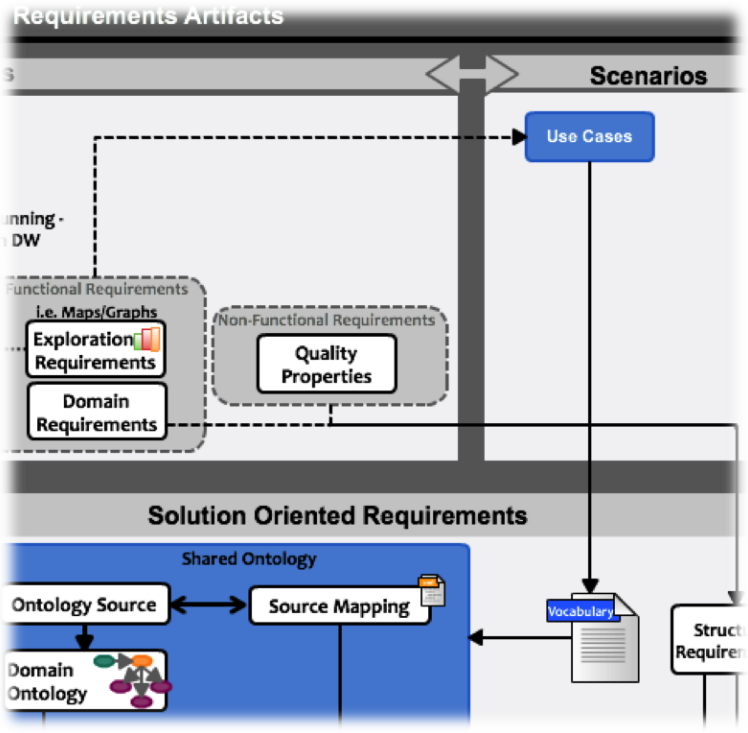


Figure 4.7: (RE4DSS) Approach - Vocabulary Specification

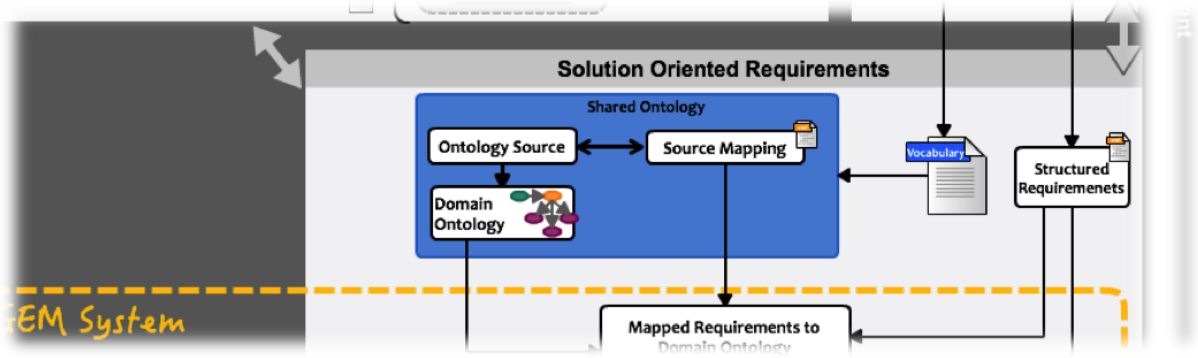


Figure 4.8: Ontology Source and Source Mapping - (RE4DSS) Approach

Cases. OWL ontology may represent diverse types of data stores; let us provide the representation of a relational data source (e.g., tables, attributes, cardinalities etc.) Example extracted from (Jov11), (see figure 4.9):

- **Source table:** The data source table “Region” is represented as an ontology class (owl:Class) (see example in figure 4.9).
- **Attributes of the source tables:** The different attributes from the table(s) are represented in as different datatype properties in the ontology, for instance r_regionkey INTEGER (primary key) and r_name VARCHAR. The domain class represents the source table of the given attribute while the range represents the datatype of the attribute (Jov11). The Datatype Property name is a Vocabulary term agreed of the domains to

construct the ontology and is also used when integrating new business requirements from the stakeholders; this is created with the name of the table as prefix and attribute as suffix plus the word ATRIBUT. (See example in figure 4.9).

- **Concept Relationships:** Sources may also consist of relationships (references) between tables, such as "Nation" includes the reference to the source table "Region" (for example: a Foreign Key in relational databases). Therefore in this example the ObjectProperty is introduced in the ontology (see example of Source Associations in figure 4.9).
- **Cardinalities between Relationships:** The relationship between two tables, must have defined their cardinalities. In the example, the tables Region and Nation cardinalities are defined to be 1..1 and 1..* respectively. This cardinality must be represented as subclasses (restrictions) inside the class.

Source Table:

```
<!--definition of the ontology class that represents source table Region-->
<owl:Class rdf:ID="Region">
...
 </owl:Class>
```

Attributes of the Source Table:

```
<!--Definition of the attribute r_regionkey, i.e., primary key, of the table Region -->
<owl:DatatypeProperty rdf:about="#Region_r_regionkeyATRIBUT">
   <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#int"/>
   <rdfs:domain rdf:resource="#Region"/>
</owl:DatatypeProperty>
```

```
<!--Definition of the attribute r_name of the table Region -->
<owl:DatatypeProperty rdf:about="#Region_r_nameATRIBUT">
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
  <rdfs:domain rdf:resource="#Region"/>
</owl:DatatypeProperty>
```

Source Associations:

```
<!-- Definition of the IsFrom relationship between the tables Nation and Region -->
<owl:ObjectProperty rdf:about="#IsFrom">
   <rdfs:domain rdf:resource="#Nation"/>
   <rdfs:range rdf:resource="#Region"/>
</owl:ObjectProperty>
```

Figure 4.9: Relational Data Source represented by means of OWL Ontology. Example extracted from (Jov11)

c) **Source Mapping**

The Source Mappings represent the structured manner to define the mappings which relate the Ontology Source with the real data stores. The mappings are intended to be defined in an XML structure. This means that for each of the concepts in the ontology may have its mapping inside the XML structure. Considering the example presented in Ontology Source, there are different kinds of mappings:

- **mapping of an ontology class**
- **mapping of an ontology datatype property** (see example in figure 4.10)
- **mapping of an ontology object property**

```
<OntologyMapping sourceKind="relational">
    <Ontology type="property">
        http://www.owl-ontologies.com/unnamed.owl#Region_r_nameATRIBUT
    </Ontology>
    <Mapping>
        <Tablename>region</Tablename>
        <Projections>
            <Attribute>r_regionkey </Attribute>
            <Attribute>r_name </Attribute>
        </Projections>
    </Mapping>
</OntologyMapping>
```

Figure 4.10: Mapping of an ontology datatype property. Example extracted from (Jov11)

The final of the Ontology Source and the Source Mapping conclude on creating a Domain Ontology which represents the knowledge based on the concepts within the domain enriched with a shared Vocabulary to denote the different properties. This will be used along with the Structured Requirements (XML) as input for the GEM system (Solution-Oriented Requirements).

After the three inputs from the GEM system (see figure 4.11), the following sections describes our accompanying GEM system aiming to facilitate the production of ETL and MD Designs with a semi-automatic means.

GEM System - Generating ETL and Multidimensional Designs

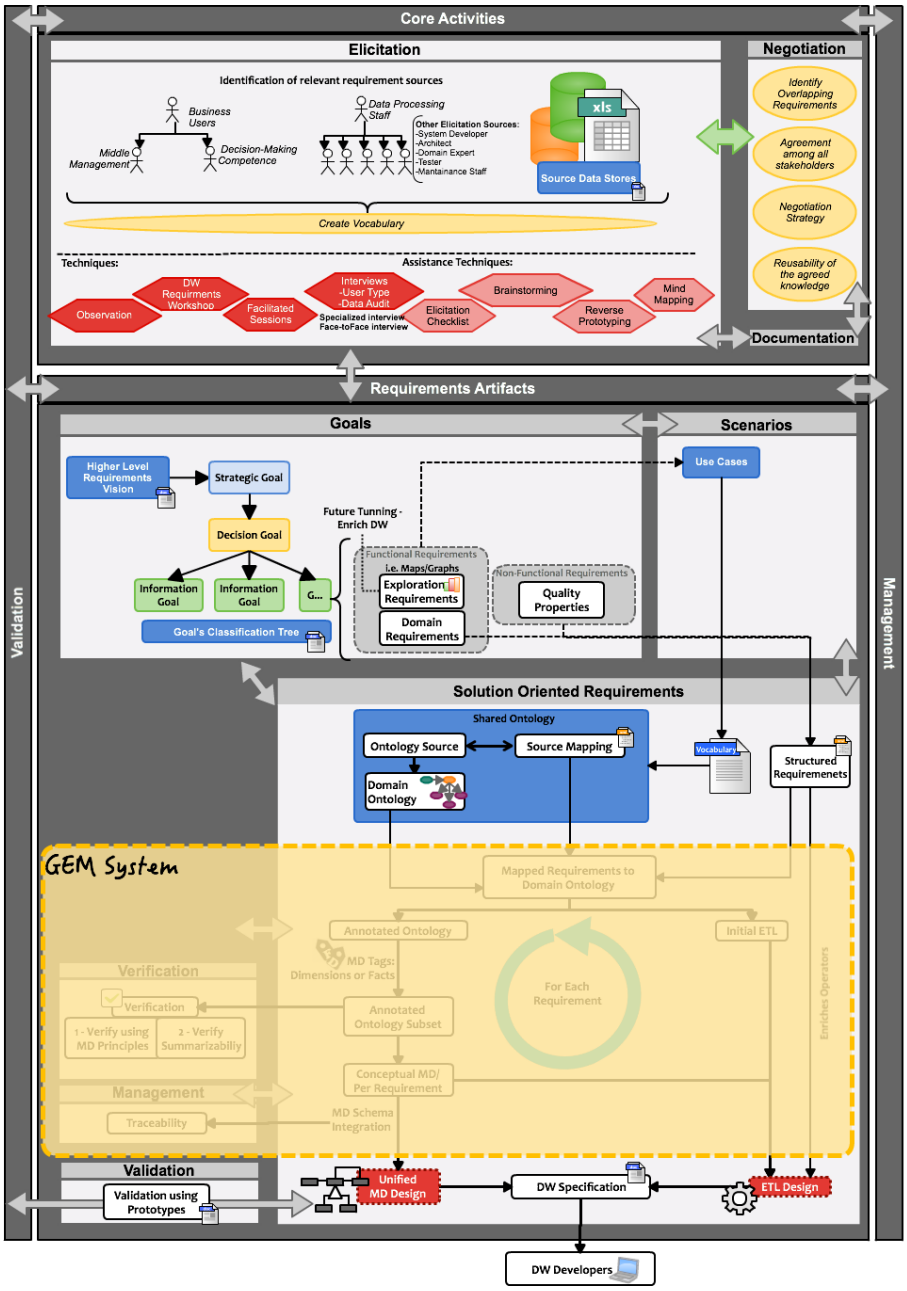


Figure 4.11: Solution-Oriented Requirements and GEM System - (RE4DSS) Approach

At the early stages of Decision Support System projects, typically, business needs are manually collected through a series of practices (e.g. interviews, etc) to be then translated into a conceptual MD and ETL designs. By being a manual process, it tends to be error-prone, that demands several rounds of reconciliation to finally satisfy all the business requirements from the user. In response to this, in 2011, professors Alberto Abelló and Oscar Romero from ESSI department (Departament d'Enginyeria de Serveis i Sistemes d'Informació) at Technical University of Catalonia (UPC), together with Alkis Simitsis presented, in (RSA11), the GEM framework. GEM represents

the system that semi-automatically produces both the Data Warehouse MD Design and the ETL Process Designs of the resulting data warehousing system, concerning both, the set of requirements and source data stores as the inputs.

The GEM system is a great fit for our study due to the following reasons:

i. GEM helps RE4DSS to semi-automate the MD and ETL design processes (corresponding to the Solution-Oriented Requirements)

ii. RE4DSS elicit different requirements from the business users and source data stores which must be documented one by one. Hence, GEM considers each these requirement separately, just as and incrementally builds the unified MD and ETL schema that satisfies the entire set of requirements.

iii. GEM also covers the Verification and Management activities for Solution-Oriented Requirements (see further subsections). During the verification, GEM approves whether the summarization is correct and makes MD sense, and also verifies the completeness assuring the union of the requirements must constitute the final entire set. Management activity keeps the traceability related to the information about data sources, the transformations performed when migrating the data from the sources, and about each of the mappings to the integration schema; this is called lineage.

figure 4.12 shows the delimitation where the GEM starts (yellow line).

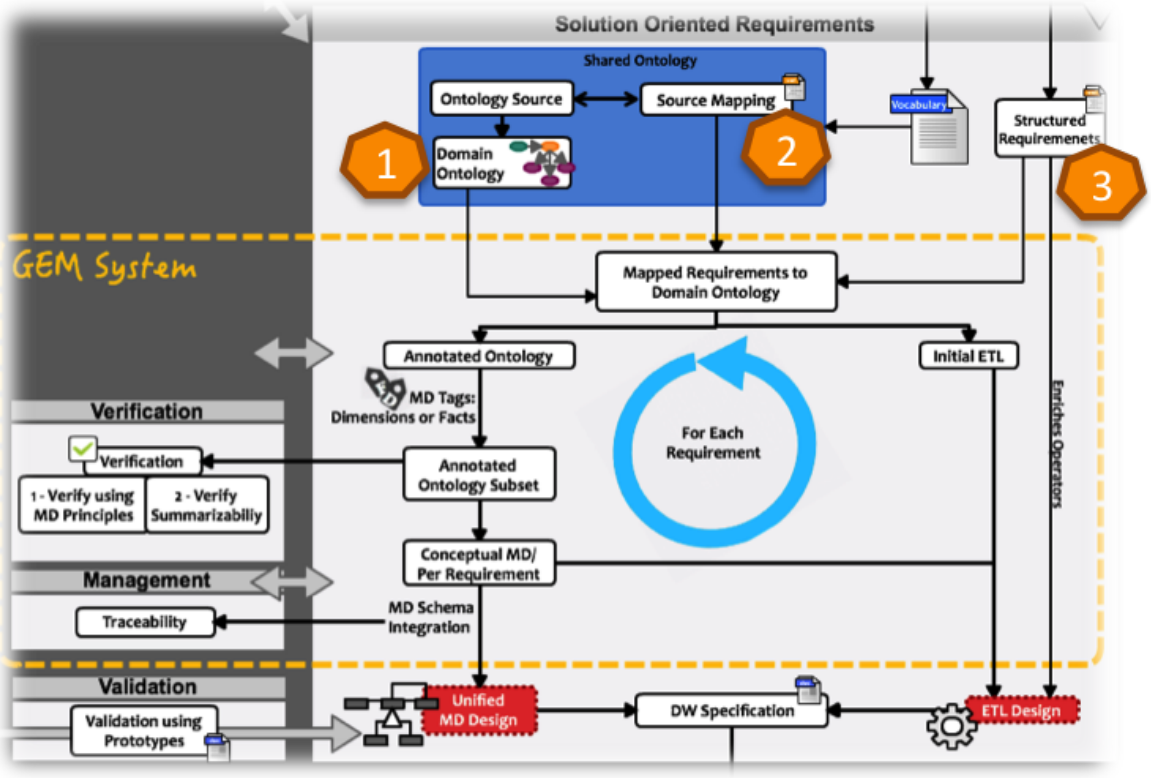


Figure 4.12: Inputs for GEM system - (RE4DSS) Approach

In summary, GEM system used in this approach, takes three inputs mentioned in the previous subsections:

i. **Two components from the Shared Ontology:**

   A. The **Domain Ontology** from the source data stores whose semantics, characteristics and constrains are represented in a tailored ontology.

   B. **Source Mappings**, expressed as an XML, which represent the information if an ontology concept is mapped to the real source data and how it is mapped. (See #2 in figure 4.12)

ii. **Structured Requirements** after using our RE4DSS approach, these are also expressed in an XML file, which represents the analytical query needed for the organizational decision making process. (See #1 in figure 4.12)

The following subsections provide an explanation of how the GEM system is used in our approach to create the final Solution-Oriented Requirements, namely: ETL and MD Designs. Notice that given the requirements in a structured and machine-readable format by RE4DSS, the GEM system provides the generation of the respective ETL and MD Designs in a semi-automatic manner. Although in the following section we give a brief introduction of the GEM system, it is not in our scope to explain the system in a detailed manner. More information regarding the GEM may be found in the work from the authors (RSA11).

These steps are divided into the following subsections are:

- Mapping the Requirements to Domain Ontology
- Annotated Ontology Subset
- Validation & Verification
- Management
- Final Output - Unified Schema and ETL Design (iterative)

i. Mapping Requirements to Domain Ontology

During the previous steps, we have covered the different inputs we can use for our Solution-Oriented Requirement. In this section, we will explain if there is a mismatch among these business requirements in the XML, and the data source by checking the concepts in the ontology if these are mapped to the sources. (See figure 4.13):

This mapping starts with the following steps. First the system is in charge of reading the input XML which includes the *Domain requirements* and *Quality Properties* from the business requirements and extracts the individual concepts. The ontology used for representing the data stores is then used to search for this concepts. Consequently, these are then tagged with a multidimensional role including these labels: Levels, Measures, or Descriptors. Although errors may occur in the process, and then a verification algorithm proposed in the work from (RSA11) must take place (see figure 4.14):

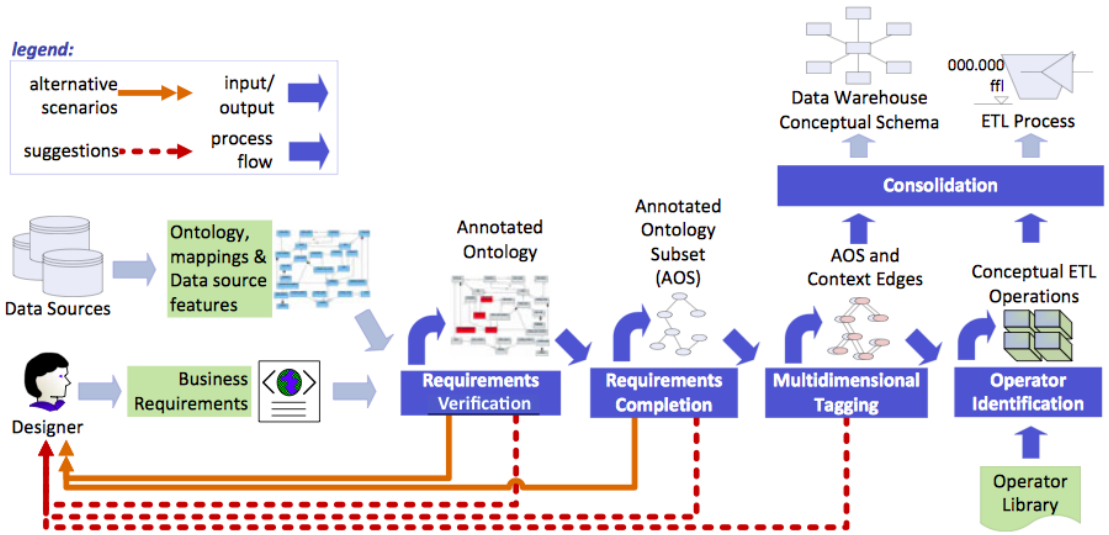


Figure 4.13: GEM System Architecture - Reprinted From (RSA11)

1. **if** the tagged concept is mapped to the sources **then** no further action is needed
2. **else if** the tagged concept is involved in a concept taxonomy **then**
   (a) **if** any of its subclass(es) has (have) a mapping **then** we annotate the tagged concept with the ETL operations 'renaming' and 'union'
   (b) **else if** any superclass has a mapping **then** we use the general concept mapped and annotate the required concept with ETL operations 'renaming' and 'selection'
      i. **if** *discriminant function* has not been specified in the input XML file **then** user feedback is required
      i. **if** the tagged node has several superclasses **then** 'minus' or 'intersection' are also considered (see Section 6 for details)
3. **else if** exists a (transitive) one-to-one association to a mapped concept **then** suggest it as a potential synonym
   (a) **if** the suggestion is accepted **then** the f-req is updated with the synonym concept
4. **else** the concept is not available in the data sources

Figure 4.14: Requirements Verification Method. Reprinted from (RSA11)

- In step 1, if it is identified that the tagged concept is already mapped to the source, then we do not need any other activity.
- When concepts are not directly mapped, in step 2(a), the system must identify which of the concept's subclasses are directly mapped and ask the user to provide new mappings according to the suggestions. These suggestions contain map ontology subclass concepts and operations the user needs to derive new mappings. On the other hand 2(b) focuses on identifying the superclasses.
- In step 3, if there are not mapped subclasses or superclasses, then it is inferred that possible synonyms of a given concept exist, meaning one-to-one associations. These are identified by using transitivity.
- After these steps are completed, the process either may continue or states "the concept is not available in the data sources" (step 4).

The end of this process represents a set of ETL operations needed to extract the concepts for the data sources in the initial ETL. This will be then included in the final ETL Design. On the other hand, it also produces

an Annotated Ontology.

ii. Annotated Ontology Subset

After considering all the business requirements, the output Annotated Ontology is completed in its already tagged concepts with additional information regarding the sources. This is done by mainly identifying intermediate concepts (with labels) that are not stated in an explicit manner in the business requirements but needed for the *Domain requirements* and *Quality Properties* (RSA11) (see figure 4.15):

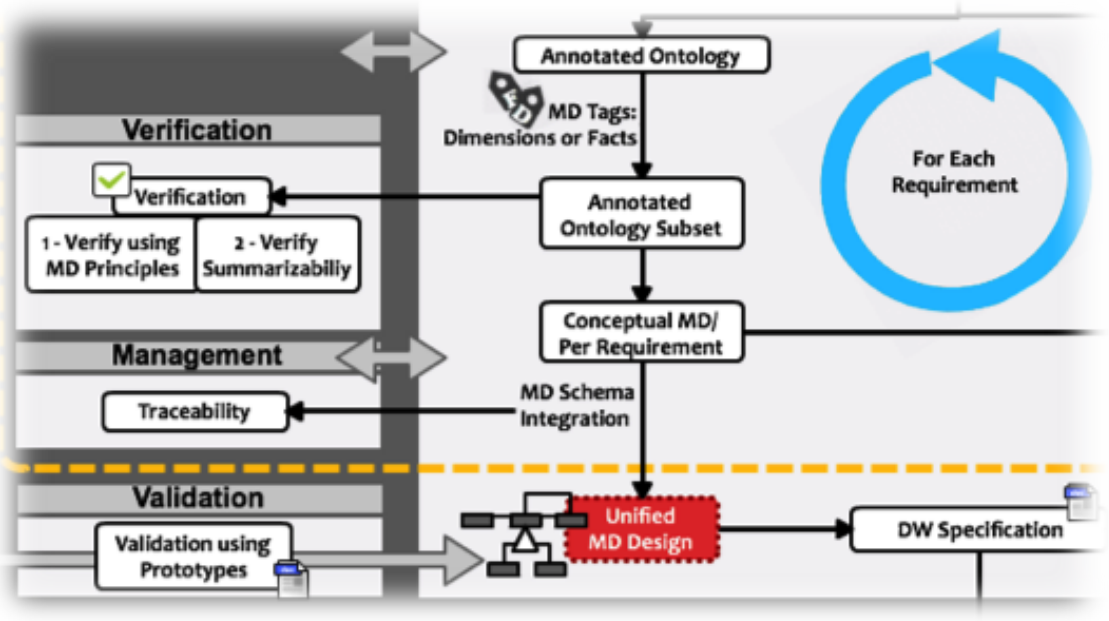


Figure 4.15: Annotated Ontology Subset & Verification - (RE4DSS) Approach

The completion process starts by a pruning process where concepts that are not mapped neither tagged are pruned. In addition, many-to-many association (e.g.*-*) are also pruned. The output of this process is an Annotated Ontology Subset (AOS). This may be represented as a graph where the ontology concepts are nodes, and the associations are edges. The challenge is trying to identify the paths between the nodes associated to the tagged concepts. This is done by the following algorithm proposed in the work (RSA11) (see figure 4.16):

- 1 (a) starts by identifying edges that relate to the tagged concepts directly.
- 1 (b) identifies edges reaching the tagged concepts (seed edges are in the right).
- Then apply transitivity starting from the tagged concepts.
- 2(b) explores new edges where right-end matches the left-end seed edge.
- Some of the restrictions are that the path cannot be explored again or a node is reached representing a tagged concept, and the path exploration is done.

1. **foreach** edge e **in** O **do**
   (a) **if** *right*_left_concepts(e) **are** tagged **then** paths_between_tagged_concepts ∪=e;
   (b) **else if** *right*_concept(e) **is** tagged **then** max_length_paths ∪=e; *//Seed edges*
2. **while** size(max_length_paths) != ∅ **do**
   (a) paths := ∅;
   (b) **foreach** path p **in** max_length_paths **do**
      i. extended_paths := explore_new_edges(p, O); *//only considering edges not in p*
      ii. **foreach** path *p1* **in** extended_paths **do**
         A. **if** *left*_concept(*p1*) **is** tagged **then** paths_between_tagged_concepts ∪=*p1*;
         B. **else** paths ∪= p1;
   (c) max_length_paths := paths;
3. **return** paths_between_tagged_concepts;

Figure 4.16: Algorithm - Defining Paths Between Concepts. Reprinted from (RSA11)

After this is accomplished, the complete AOS including the new concepts needed to answer the functional requirements (*Domain requirements* and *Quality Properties*) must be checked, so the complete graph makes Multidimensional (MD) sense (read "Validation & Verification" section for more details). In order to guarantee the MD Design principles, the "factual and dimensional nodes must be related properly" (RSA11). If not, any issue encountered in the graph must be solved.

d) **Validation & Verification:**

Verification aims to confirm the following question, Am I building the right MD Design for the purpose of the DSS? The important matter is to avoid is error propagation; this is an error when the artifacts (Goal Classification and Scenarios) is not detected until the release of the system. This can lead to high unexpected costs. In addition to this verification question, the system may be validated by asking: Am I building the Decision Support System right? To summarize, verification is done in an automatic manner by GEM, whereas the validation is done using prototypes jointly with the stakeholders.

In order to verify that concepts (tags) produced in the Solution-Oriented Requirements make MD sense, two main aspects are verified: First, the placement of data in a multidimensional space, and second, the correct summarizability of the data (see figure 4.17).

i. The first aspect proposed in the work of (RA10) verifies whether the factual data is arranged in a multidimensional space. For instance, each of the instances of the factual data must be classified to a particular point the analysis dimension. The authors claim that by having one fact and several dimensions to analyze gives rise to a multidimensional schema. Every instance of the fact is related to at least and at most one instance of a dimension, while every dimension instance may be related to many instances of the fact.

ii. Whether the summarization is correct by examining following conditions proposed in (MLT09):

   A. disjointness - the sets of objects to be arranged must be disjoint

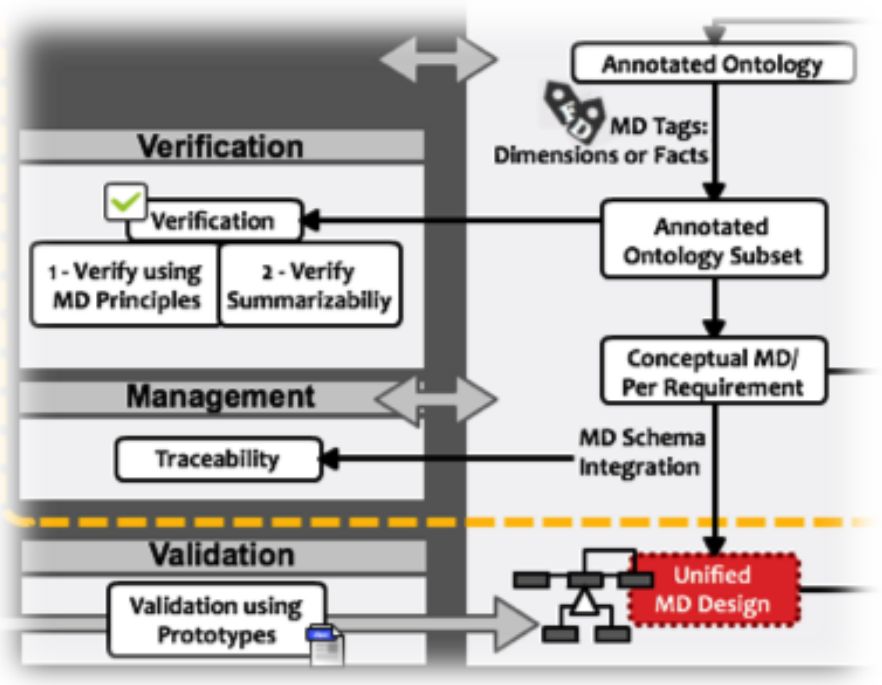


Figure 4.17: Verification, Management and Validation - (RE4DSS) Approach

B. completeness - the union of the requirements must constitute the entire output, meaning all requirements must be integrated to fulfill all the requirements in the MD and ETL Designs.

C. compatibility of the dimension with the type of the measure being arranged and the aggregation function

Mapped many-to-many associations (*-*) violate the three summarization necessary conditions. Therefore, they cannot be considered for the resulting multidimensional design. Moreover, if fail occurs during verification, there must be alternative solutions proposed. The pre-process starts by deriving new MD knowledge from the non-tagged concepts considered to be a factual or dimensional role (RSA11). Next, these tags are verified to see if these are sound and make MD sense. Then, all potential MD tagging are determined and how this may affect the output schema. For the tagging combinations, alternative annotations are created. In the end, only the AOS which makes sense are considered.

In addition to the automatic verification done by the GEM system, we have added a validation activity at the end of our approach in order to now validate the output ETL and MD Design (see figure 4.17). This is a joint activity with the stakeholders where negotiation can be done with the use of prototypes/mock-ups in order to validate the output before the implementation.

The following subsection explains the management process. Afterwards, the final output is presented in the last subsection where each of the resulting MD designs causes the new invocation of the Operation Identification to generate the final ETL design.

e) **Management:**

Management of the Decision Support System requirements encompass lineage to keep traceability capturing information about different aspects such as:

- **Traceability of Data Sources:** Keeping traceability of the sources is crucial in order for the developers/stakeholders perceive the origin of the data. For instance, stakeholders may require to separate the official governmental data sources or other agencies with empirical bricks from the unofficial data obtained by uncertified public websites, this may be done by tracing the origin of the data sources.
- **Traceability of the mappings to the integration schema (including schema versioning):** There have been several approaches which focus on keeping the trace of the changes maintaining different versions of the schema before being unified. One of these is (BEK04) which stores and maintains alternative versions of the schema to simulate different business scenarios. GEM along with ORE module considers each requirement separately and incrementally build the unified MD schema satisfying the entire set of requirements (JRSA12b) (see following section for more details). In addition, it does the same for the ETL process, besides the multidimensional schema produces a work-flow taking the ETL for each requirement and tries to find the overlapping between them to make the schema.
- **Traceability of metadata):** The job from GEM and with the complementary module called ORE, is that in all stages, it maintains and keeps a structure called "Traceability of Metadata"; this enables to systematically trace everything from the MD design which has been integrated so far (JRSA12b). This same work explains that ORE works in an iterative manner where the traceability of the metadata keeps growing every time a new requirements is integrated, along with its feedback if provides the main solid base for the final MD Schema. One of the advantages of usin ORE is that it is possible to avoid overloading the MD Design when is producing the schema (e.g. all alternatives of MD interpretations for a single requirement, however only one of the alternatives is chosen to create the final MD Schema).

4.4 Final Output - Unified Schema and ETL Design

Followed by the automatic verification process by GEM, the final output is a Solution-Oriented Requirement. After the initial ETL Process, the ETL operations which are needed to support the dataflow are identified and complement the design with additional information (from the source data stores to the MD target data stores). During the final step, all the above stages are meant to run once for each input business required. These are then conciliated into a unified MD design, and single ETL Process design (see figure 4.18); this is done using two modules: ORE (mentioned in the previous section for metadata traceability) and COAL modules. ORE module, introduced in (JRSA12b), provides a semi-automatic method that considers each requirement separately and incrementally build the unified MD schema satisfying the entire

set of requirements. COAL (JRSA12a) does the same for the ETL process, besides the multidimensional schema produces a work-flow taking the ETL for each requirement and tries to find the overlapping between them to make the schema.

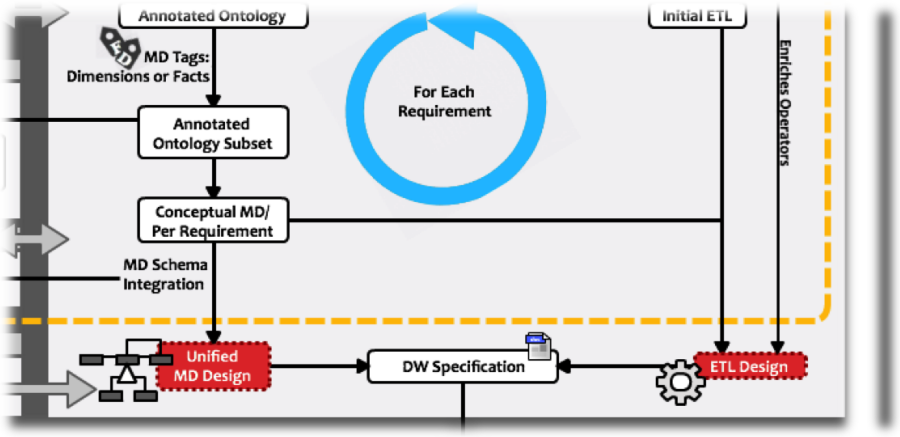


Figure 4.18: Unified MD design and ETL design - (RE4DSS) Approach

The final documentation is entitled the “Data Warehouse Specification” which includes the final and integrated MD design and the ETL design considering all the requirements. This final document will be used as an input for the DBMS or ETL tool in order to start the implementation.

CHAPTER

5

"Men should stop fighting among themselves and start fighting insects."
— Luther Burbank

Case Study:
Data Warehouse Requirements for the global WHO surveillance system to eliminate Chagas disease

This final chapter presents the contributions of our RE4DSS approach introduced in Chapter 4, employed to a real case study provided by the World Health Organization (WHO), a United Nations (UN) agency. The project consists to advance in disease control by creating a Chagas Information Database (CID) system for surveillance to raise awareness on the Chagas disease storing and analyzing strategic data regarding the disease. This chapter initializes with the description of the Chagas Disease, along with its background. This is followed by the methodology used in order implement our RE4DSS approach into the Chagas case study with the assistance of the GEM system.

5.1 Background - Chagas Disease

The WHO is responsible for providing leadership on world-wide health matters. In 2010, WHO launched the First Report on Neglected Tropical Diseases (NTD) (SDC10), that included 17 diseases: Chagas disease, Buruli ulcer, taeniasis cysticercosis, etc.

According to the official WHO website (Sta14), the Chagas disease (also known as the Human American Trypanosomiasis) is a mortal illness caused by the protozoan parasite, also called Trypanosoma cruzi (see figure 5.1). The website mentions that the transmission to humans usually occurs through contact with faeces of vector insects (triatomine bugs), including the ingestion of contaminated food, transfusion of infected blood, congenital transmission, organ transplantation or laboratory accidents (WHO13). People infected with T. cruzi are located mostly in the endemic areas of 21 Latin

Figure 5.1: Blood-sucking insect responsible for the Chagas disease. Reprinted from: (WHO13)

American countries, (see map in figure 5.2): Argentina, Belize, the Bolivarian Republic of Venezuela, Brazil, Chile, Colombia, Costa Rica, Ecuador, El Salvador, French Guyana, Guatemala, Guyana, Honduras, Mexico, Nicaragua, Panama, Paraguay, Peru, the Plurinational State of Bolivia, Suriname and Uruguay (WHO13).

Chagas disease began by attacking the poorest populations in Latin America, however with the time, these wild insects started adapting to the domestic environment. According to the official website, Chagas disease can be easily transmitted by six different transmission routes, mainly vectorial, oral, congenital (mother to child) and blood transfusion routes, but also by organ transplantation and laboratory accidents. In addition to the danger caused by the triatomines, the Chagas disease has been now detected in the United States, Canada, 17 European countries (see map in figure 5.2), and some Western Pacific countries, mainly Australia and Japan (WHO13).

At present, it has been reported around 7 to 8 million people worldwide infected, and still, there is no available vaccine to prevent from the infection (Sta14). The main concern is that Chagas disease has been internationally recognized as a critical medical and social issue due to the number of cases of infected people that keeps increasing worldwide from year to year. For this reason, countries are progressively realizing the need to to control/eliminate the Chagas disease by interrupting its transmission.

With the objective to advance in disease control, in 2010 the World Health Assembly (WHA) approved the resolution "Chagas disease: control & elimination", with the following key objectives mentioned in (Who10):

- Tackling of all transmission routes
- Integration of care of patients with acute and chronic clinical forms of Chagas disease
- Recognition of an increased number of cases of Chagas disease in countries where the disease is not endemic regions

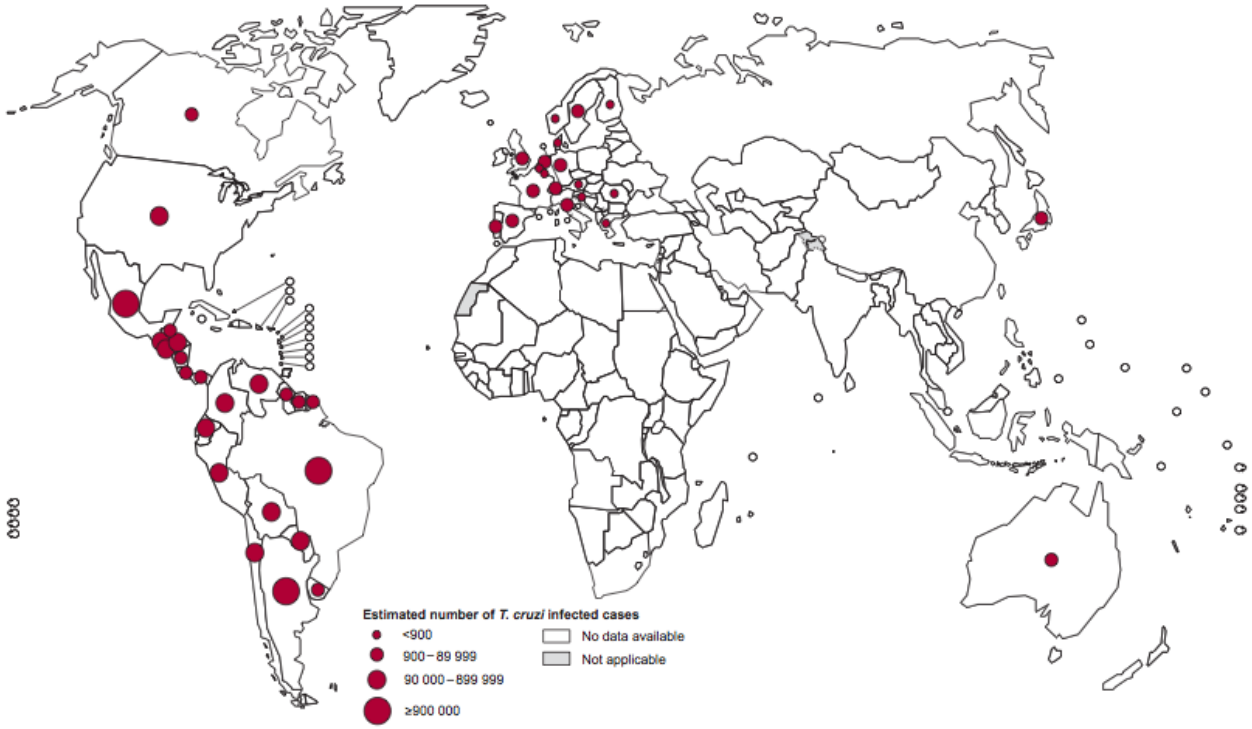


Figure 5.2: Global distribution of cases of Chagas disease, based on official estimates, 2006 - 2010. Reprinted from: (WHO13)

Finally in June 2013, the WHO approved the Programme on Control of Chagas disease. This aimed to achieve the elimination of the chagas disease transmission by **creating an information and surveillance system to facilite the access to heterogeneous sources, creating interactive data, providiung disease statistics, visualizing maps and diagrams with current patients.**

Our main objective for this case study is to implement and validate our approach RE4DSS during the Requirements Engineering cycle of the global WHO information surveillance system to control/eliminate Chagas disease.

5.2 Methodology

A case study is presented based on the employment of our RE4DSS approach along with the GEM system (explained in chapter 4) (see figure 5.3). We start by describing the core activities in section 5.3, mainly the elicitation activity and outlining the problems faced during the interviews, questionnaires and observation. During these activities, negotiation was done throughout the complete project in order to achieve agreement among all the stakeholders from the WHO. Documentation of the Requirements Artifacts was generated during the whole project in order to have evidence of the goals, scenarios and Solution-Oriented Requirements.

After describing the Core activities, we will detail the Requirements Artifacts. By using the different elicitation techniques, the stakeholder's intentions were documented as goals that were later refined into a Goal Decomposition Tree. Information Goals were extracted from the leafs from this tree in order to obtain the classification of requirements.

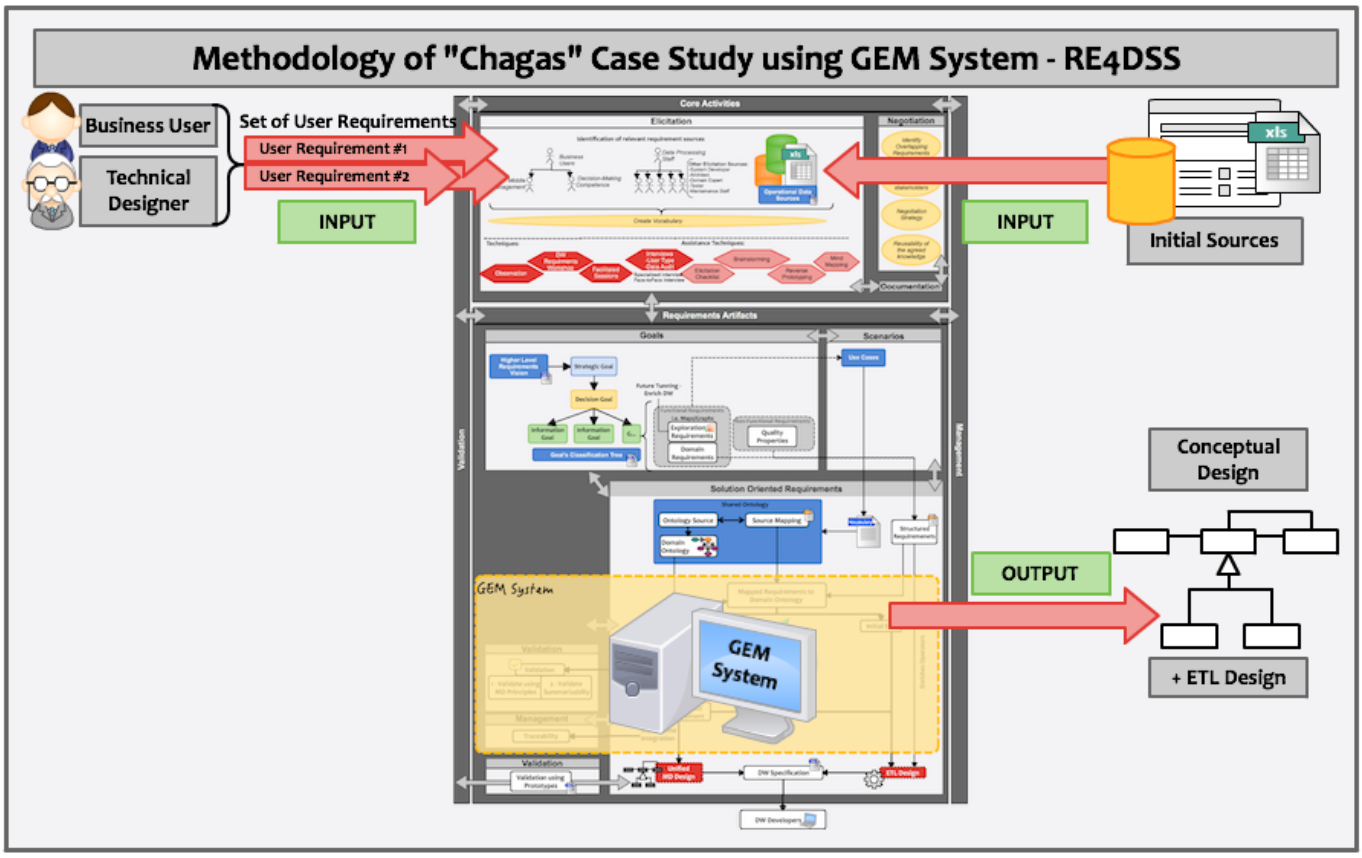


Figure 5.3: Chagas Case Study Methodology

Subsequently several requirements were used either for tuning the visualization of the Data Warehouse or to create the Scenarios that in this case we described by means of Use Cases based on Cockburn's practical guiding (Coc01). Followed by this, a vocabulary created since the beginning of the process is then used to enrich the Shared Ontology. The sources from the WHO, at this stage, are represented by means of an OWL ontology. With this information in hand, the Ontology, Source Mappings and the Structured Requirements are introduced into the GEM system which will finally generate the ETL and MD Design. Verification and Management activities are done inside the GEM system, such as summarizability and traceability of requirements. Finally, the ETL and MD design begin are created, and validated with the stakeholders using prototypes before being implemented.

The following sections demonstrates how the RE4DSS is employed in the Chagas project. Therefore, the following sections will be divided into the basic elements proposed on Klaus Pohl framework which includes: Core Activities, Requirements Artifacts, Validation, and Management. In addition, we have added the Verification process.

5.3 Core Activities

a) **Elicitation**

At the very earliest stage of the data warehouse design project, the main objective was to identify the relevant sources to start eliciting the requirements.

Sources

The elicitation sources identified were divided in two, namely: Requirements extracted from the source data stores and requirements from the WHO stakeholders (Health Ministry Officers, Researchers, etc.). During the elicitation activity, several software analysts from the UPC (Polytechnic University of Catalonia) participated (see figure 5.4).

i. **Source Data Stores**: Sources included three different inputs, from excel sheets to open data files. This information is in the following sections captured in terms of OWL ontology while also integrating it with our domain vocabulary. When new sources arrive, mapping to the other sources must be done. The sources used were the following:

   A. **Excel Kiss Bug Inspection and Spraying:** Data about the raids organized by researchers and / or governments to inspect / analyse / spray dwellings to control the kiss bug presence in lived areas. Source: Excel file.

   B. **MetaTri Database:** It is a geolocated vectorial-oriented database. Stores information about where kiss bugs can be found in South America. Source: Microsoft SQL Server database. This is a database created by Jorge Rabinovich (Argentinian entomologist) (VN14). This database contains information regarding all the known species in the world that may transmit the Chagas disease.

   C. **FAO Climate and flora:** Data about the climate and flora associated to any point in the world. Source: Open data files to be downloaded from FAO.

ii. **WHO Stakeholders (Business Users)**: These sources may be identified by using the proposed techniques of our approach (described later in the text). During the study, elicitation was performed through Interviews, Observation, Questionnaires and Workshops in order to collect the requirements from the business users and needs. The identified critical business users (specialist in Chagas disease and triatomine bugs) were the following: Health Ministry Officer, Researcher and WHO (see figure 5.4). Roles are described below (Description of business users entirely extracted from (VN14)) :

   - **World Health Organization:** The WHO actor interacts with the system in order to visualise, manage, analyse and import/export information. On the one hand, WHO has access to the entire information system database, therefore, it is able to consult, but not modify, any type of information related to any

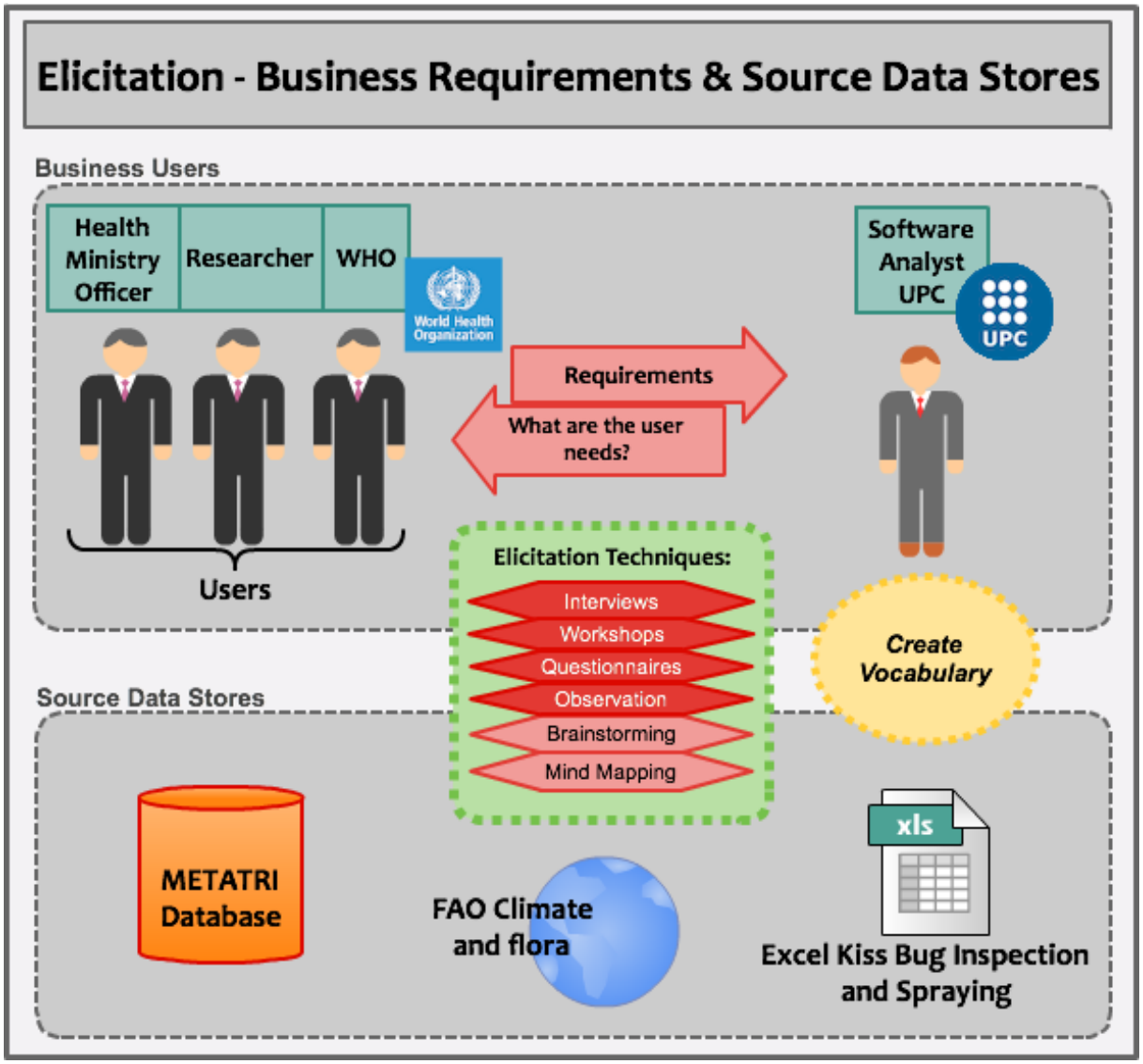


Figure 5.4: Elicitation of Business Users and source data stores. Adapted from: (RSA11)

country. Additionally, WHO actor will be able to manage assessments, regulations and certifications. On the other hand, it can generate and visualise maps and diagrams with any type of information gathered in the system. Finally, it will be able to import and export any type of information provided by countries or researchers by uploading it with an Excel file.

- **Health Ministry Officer:** The HMO actor, in a similar way as the WHO actor, interacts with the system in order to visualise, manage and import/export information. Health Ministry Offcers are only allowed to provide, modify, and delete information related to his own country, therefore, he will not be able to consult, modify and delete information from other countries. Information he can manage range from healthcare, through transmission interruption, to systemic and normative. In particular, he can manage the following information: treatments, diagnosis, insecticide applications, triatomine bug

studies, inspections, regulation and systemic information. In addition, he will be able to generate and visualise maps and diagrams with the information he provided to the system. Finally, he will be able to import and export any type of information by uploading it with an Excel file.

- **Researcher:** The Researcher actor, in a similar way as the other actors, interacts with the system in order to visualise, manage, import/export, and analyse information. Researchers can provide information related to healthcare, and transmission interruption. In particular, he can manage the following information: treatments, diagnosis, insecticide applications, triatomine bug studies and inspections. In addition, he will be able to generate and visualise maps and diagrams with the information he provided to the system. Finally, he will be able to import and export any type of information by uploading it with an Excel file.

Elicitation Techniques

Once identified the sources, four different techniques were used to extract the to elicit the requirements (see figure 5.4). These are described below:

- **Interviews**: The UPC software analysts conducted various interviews to the Chagas experts. These interviews were exploratory interviews conversation-like where the interviewee elicited the opinion regarding some issues. For instance, software analysts interviews the Chagas experts to understand the variables needed to be taken into account when defining the MD Design (e.g. insect species, countries, spray). Interviews were done according to Kimball's recommendation doing face-to-face or voice-to-voice interviews (Kim98).
- **Workshops**: In order to refine the high-level goals, several intense workshops took place for 2-3 day sessions. During the workshops, assistance techniques were also used: brainstorming (to define the high-level goal) and mind mapping.
- **Questionnaires**: Considering that face-to-face interviews were not always possible, occasionally questions were sent from the software analysts to the Chagas experts via email.
- **Observation**: During this technique, the observer must elicit the requirements by studying the existing systems. Even though there is no current information system to eliminate the Chagas Disease, some similar projects were also analyzed within the WHO context.

Creating Vocabulary

Once identified the techniques to elicit the sources, and the elicitation of requirements started, problems commenced to bloom. One of the most common difficulties faced at the beginning of this project, which is very common during the Decision Support Systems is that the software analyst from the UPC were not familiar with the domain of the project. Consequently, the first meetings were only focused on understanding what the Chagas disease means, how it is spread, its transmission, the treatment, etc.

As mentioned in the previous chapter, since there are Chagas specialist involved in the decision-making process which certainly use medical and domain terms for their everyday communication, we must expect that the input requirements are also expressed using this medical and domain terminology such as "trypanosomiasis and Triatominae"; but from the other side, the sources do not always match the same terminology, for instance, the Chagas bug has different name variations such as: kiss-bug, vinchuca, barbeiro, chinche, chipo etc. For this reason, there was a need to bridge this gap between the Chagas experts and the software analysts terminology by creating a Vocabulary (see figure 5.5) in order to identify the real business needs, avoid linguistic inconsistencies between Chagas specialist and software analysts, to homogenize used terms (e.g. definitions, acronyms, abbreviations in data sources), but most importantly, the output of the vocabulary was mainly meant to enrich the concepts in the Domain Ontology. The vocabulary was updated throughout the complete project, and it must provide a cross-reference structure.

Attribute	Definition
Intradomicile Infested	The dwelling's intradomicile has at least one adult living inside.
Intradomicile Colonized	The dwelling's intradomicile has at least one nymph living inside.
Peridomicile Infested	The dwelling's peridomicile has at least one adult living inside.
Peridomicile Colonized	The dwelling's peridomicile has at least one nymph living inside.
Number of adult insects Intradomicile	Number of adult insects found in the ddwelling's intradomicile
Number of nymph insects Intradomicile	Number of nymph insects found in the dwelling's intradomicile
Number of adult insects Peridomicile	Number of adult insects found in the dwelling's peridomicile
Number of nymph insects Peridomicile	Number of nymph insects found in the dwelling's peridomicile
Infrastructure promoting risk (Domicile)	House structure (roof, walls or floor) at risk of colonization, facilitated by the use of organic materials or presence of cracks and crevices.
House hygiene promoting risk (Domicile)	Lack of house hygiene with disordered accumulation of domestic goods.
Hygiene practices oromoting oral transmission risk (Domicile)	Lack of hygiene practices and sanitation and lack of good manufacturing practices.

Figure 5.5: (RE4DSS) Approach - Chagas Case Study Vocabulary Creation. Reprinted from (VN14)

b) **Negotiation**

Usually on software projects, different stakeholders have different wishes and needs that may conflict each other. And so, conflicts must be resolved in order to achieve an agreement regarding the system's requirements. From the literature and the Chagas project, we have found that the negotiation activity is similar in traditional software projects and Decision Support Systems, the main difference is the concept of the negotiation. For instance, in Decision Support Systems negotiation may be regarding specific granularity for the data, visualization of results, etc.

The main conflict faced while the negotiation activity in the Chagas project was an interest conflict (defined by (Poh10)) where different interests of stakeholders contradict. This conflict involved to either: (1) collect as much information as possible with many details (e.g. details about how many insects were found in each dwelling inspection control) but taking the risk that the information provided by the system would not be accurate due to the level of details, or (2) to have the minimum information possible to obtain the maximum criterion to eradicate the Chagas disease.

To solve this problem, negotiation using mock-ups were created to finally achieve an the agreement on using the minimum information possible. Some of the suggested negotiation techniques by the literature are Win-to-Win and Interaction Matrix (as explained in the previous chapter). Nevertheless, negotiation activity does not vary much in comparison to other traditional software systems.

c) **Documentation**

Documentation was generated during the complete project in order to describe the Requirements Artifacts, namely the goals, scenarios and solution-oriented requirements. This focus of requirements documentation in the Business Intelligence field enables the achievement of requirements to be consistent and also traceable over the entire project (BDGM08). This includes any output from the models, techniques and assistance techniques (e.g. Interviews, Workshops, Use Cases, Observation, Conceptual design, Vocabulary, Constraints, Business Rules, Sources, Ontology, Traceability Matrices, Negotiation, etc.). In addition, diverse templates may be used in order to document these.

Examples of Decision Support Systems documentation include:

- Use Cases Specification
- Project Vocabulary
- Data Warehouse Vision
- Requirements Management Plan.
- Data Mart Vision
- Multidimensional Requirements Specification
- Non-Functional Requirements Specification
- Business Rules Specification
- Revision Report

Any documentation must be done according to the rules and quality standards. During the Chagas project, documents used the [IEEE 830-1998] software requirements specification standard (see figure 5.6).

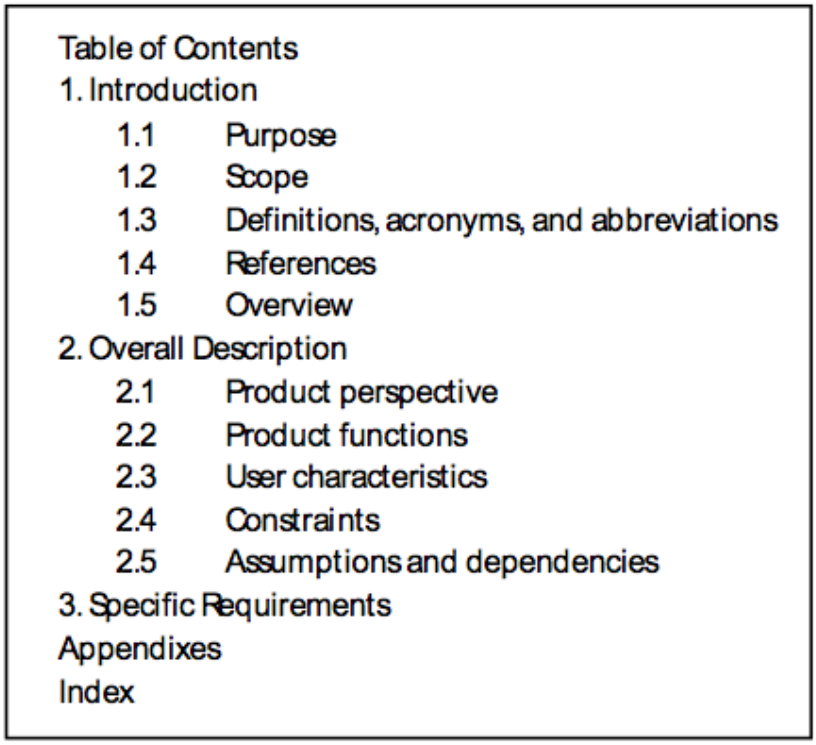


Figure 5.6: (RE4DSS) Approach - Standard structure for software requirements specification according to [IEEE Std 830-1998]. Reprinted from (Poh10)

5.4 Requirements Artifacts

The term "requirements artifacts" stands for a documented requirement using a specific format (Poh10). In this section we differentiate between three different of requirements artifacts according to our RE4DSS approach, namely goals, scenarios and solution-oriented requirements carried out during the CID project.

a) **Goals**

Goals refer to the documentation of stakeholder's intentions about the objectives. Goals state what exactly is expected or required from the system. In order to identify the main aim of our stakeholders, during the elicitation (e.g. interview) is important to ask: "what do you do (and why)"; and not asking "what do you want" in the system (Kim98). With this, we extract the main need which will be the input at highest level of our Goal Classification Tree in our further steps. Goal Decomposition is used to refine the system vision into objectives to be fulfilled by the Decision Support System.

For the scope of our case study, requirements were gathered in the form of user stories free-text and were refined to be structured during the negotiation process with the Chagas Specialists. The following running example provides an extraction of elicited requirements stated by the Health Ministry Officer from the WHO:

Running Example - Elicited High-level Vision Requirement:

Health Ministry Officer: *"We need a system to control/eliminate the Chagas Disease".*

After identifying the main objective, other objectives related to the main objective may now be elicited. From this main goal presented above, we may narrow into the following goals:

Running Example - Strategic Goal Decomposition:

i. **Researcher:** *"I need to import Excel files containing inspection data in the system automatically so that this data is delivered to the WHO"*
ii. **Health Ministry Officer:** *"I need to visualize maps showing the presence / absence of triatomines per species, genus and geographical area"*
iii. **Health Ministry Officer:** *"I need to generate graphics showing the temporal variation of evaluated (inspected) houses by department every two years for La Rioja (Argentina)"*
iv. **Health Ministry Officer:** *"I need graphics from infested dwelling to be updated every 2 days for La Rioja (Argentina)"*

Having identified the higher-level requirement's vision and its sub-goals, a Goal Classification Tree is now built (see figure 5.7). This tree is built according to the level of abstraction, namely Strategic Goals, Decision Goals

and Information Goals. To refine the (**Strategic Goal**) "I need a system to control/eliminate the Chagas Disease", it is intended to elicit the answer of the following question "how can our main/strategic goal (create a system to control/eliminate Chagas disease) be achieved?". After answering the question, it is easy to realize that the Chagas specialists needed a Decision Support System where they could store their data representing single and integrated view of the Chagas disease characteristics (including dwellings, countries, infected individuals, etc.) intended to be exploited by Chagas specialists. These are classified as (**Decision Goals**). We keep on refining the decision goals, until reaching the last subdivision of the tree (leafs). In this last subdivision the question must answer "how can the decision goal be achieved?", answers to this question are classified as (**Information Goals**), for instance, "Visualize Maps of Infested Dwelling".

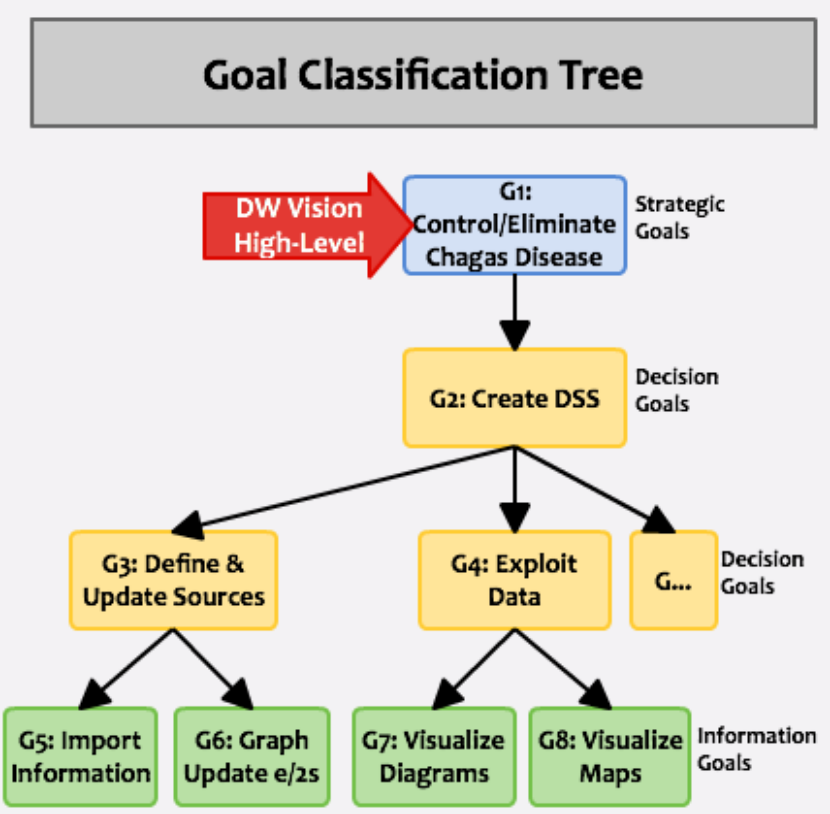


Figure 5.7: Goal Classification Tree of Chagas Case Study - RE4DSS

Business Rules were also taken into account during the goals elicitation in order to produce a better result when defining the goals. After the Goal Classification Tree is created, the Information Goals were the point of reference to start the classification of the information requirements, that can be either *Exploration Requirements*, *Domain Requirements* or *Quality Properties*. Each of the Information Goals must be classified into one of these definitions (see figure 5.8), for instance, "Import Information" is classified as a *Domain Requirement*, "Graph Update every 2 seconds" is classified as a *Quality Properties*, and finally "Visualize Diagrams and Maps" are classified as *Exploration Requirements*. From this classification, it is easy to exploit each of the requirements. Hence, the output is the following:

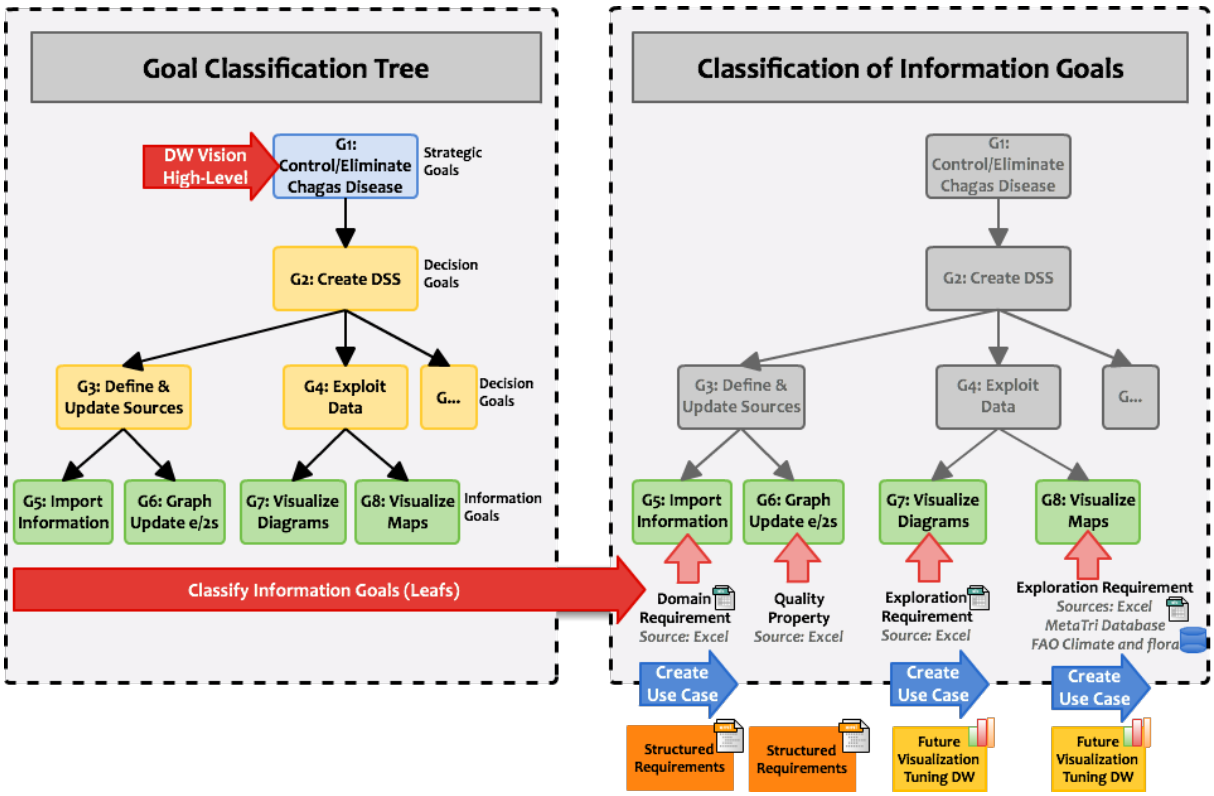


Figure 5.8: Goal Classification Tree of Chagas Case Study Identifying Requirements - RE4DSS

i. *Exploration Requirements* and *Domain requirements* are used to create the Use Cases (see Scenarios section) that will later be used to enrich our Vocabulary (presented in elicitation activity).

ii. *Exploration Requirements* such as visualization for maps and graphs will also be utilized in the future for tuning and enriching the Data Warehouse depending on the WHO, Researcher and HMO needs. (See "Final MD and ETL Design" section at the end of this chapter for more information of how these requirements are finally exploited)

iii. *Domain requirements* and *Quality Properties* are expressed as Structured Requirements in an XML file where multidimensional concepts are identified (e.g. measures and its dimensions). (See "Solution-Oriented Requirements" section for more information).

From this goal classification is now easy to identify the sources needed for each of the information requirements. Making easy the task utilized only the required sources of information needed to build the Chagas system without using unneeded sources. In addition, this fact matches the negotiation activity where it was agreed among the end-users from the WHO and the software analysts from the UPC to have the minimum information possible to obtain the maximum criterion to eradicate the Chagas disease.

After goals are elicited, Use Cases are then created in the following section as scenarios.

b) **Scenarios**

Scenarios document concrete examples of the system's usage by illustrating the fulfillment or non-fulfillment of the goal (Poh10). During the scenarios, we have created a Use Case for each classified *Exploration Requirements* and *Domain requirements* (see figure 5.8). The Use Cases aim to detail the process needd to be followed in order to implement the functionalities of the Decision Support System and its interaction with the actors. It is possible to use a reference template to document the Use Case including attributes in columns (ID, name, author, version, change history, goals, actors, precondition, post-condition, etc.). During this case study, scenarios are described by means of (Coc01).

In the following examples we will provide two Use Cases extracted from the Information Goals from our previous example, namely, Use Case Import Information and Use Case Visualize Diagrams (Use Case examples extracted from (VN14)):

Running Example - Use Case Import Information:

Use Case Name: Import Information

Active Actor: HMO user

Trigger: HMO user indicates export information

Preconditions: The HMO user must be identified and authenticated.

Stakeholders and interests

i. HMO user: obtaining and importing information about diagnosis, treatments, triatomine bug studies, inspections, **insecticide applications** and systemic and normative information from the system in order to evaluate and analyse it.

Main Success Scenario

i. User indicates import information
ii. System presents the different types of information that can be imported: Inspection information, Triatomine Bug information, Insecticide Application information, Diagnosis information, Treatment information, Systemic information and Normative information.
iii. User selects the information to import
iv. System shows all available geographic objects in the system
v. User selects the geographic objects which he wants to extract information from and then confirms the exportation
vi. System requests the import file name
vii. User enters the import file name

Running Example - Use Case Visualize Diagrams:

Use Case Name: Visualise Diagrams

Active Actor: WHO user

Trigger: WHO user indicates visualise a diagram

Preconditions: The WHO user must be identified and authenticated.

Stakeholders and interests

i. WHO user: visualise diagrams about information gathered in the CID system, which includes: Inspections, Diagnosis, Treatments, Fumigations, Triatomine bug studies, Climatological data, Systemic information and normative information.

Main Success Scenario

i. User indicates to visualise a diagram

ii. System presents all different types of information that can be visualised in a diagram: Inspection variables, Diagnosis variables, **Insecticide Application** variables, Triatomine Bug variables, Climatological variables, Treatment variables, Systemic variables and Normative variables.

iii. User selects all variables he wants to visualise in the diagram (e.g. Climatological variables, Treatment variables, Systemic variables and Normative variables)

iv. System shows all available countries and their first administrative divisions

v. User selects the geographic objects and then confirms the creation of the diagram

vi. System processes all information and generates the diagram - User repeats steps 3,4 and 5 for each diagram he wants to generate

Outcome A final diagram visualization with the information selected by the WHO stakeholder has been generated.

Straight from the Use Cases obtained in the Scenarios, the Vocabulary will be complemented with new concepts/terms identified in order to enrich our Domain Ontology (See Solution-Oriented Requirements). By our Use Cases presented in Scenarios, we may identify new concepts such as “Insecticide application and information variables” used in both Use Cases (see bold font in Use Cases presented above).

To illustrate this, we have added these concept to our vocabulary along with its definitions and derivations (considering that it does not exist in our vocabulary) (see figure 5.9). This document main purpose is later be used to enrich the domain ontology in the Solution-Oriented Requirements. Notice that the vocabulary is not only enriched by the Use Cases, but this must be iteratively enriched during the complete life-cycle of the project during also the elicitation of goals for instance.

Attribute	Definition
Insecticide applied in Intradomicile	House that received an insecticide application in the intradomicile
Insecticide applied in Peridomicile	House that received an insecticide application in the peridomicile
Type of insecticide	Type of the insecticide: Pyrethroid and Organophosphate
Active ingredient	Active ingredient of the insecticide: deltamethrin, permethrin, alfacypermethrin, betacypermethrin, lambdacyalothrin, Cyfluthrin, Fenithrotion, Malation and Parathion
Insecticide application technique	Technique used when dwelling was intervened: spray, fumigant and canister.

Figure 5.9: Complementing Vocabulary with Use Cases - RE4DSS

c) **Solution-Oriented Requirements**

Goals and scenarios are used to form: solution-oriented requirements. As the final result of the Solution-Oriented Requirements, a conceptual model is created (See figure 5.10). Therefore, the preliminary phases are crucial to create a conceptual design for the Data Warehouse / Data Mart; such as identifying facts, dimensions, measures during the elicitation process. In addition to the conceptual model, the ETL work-flow is also created.

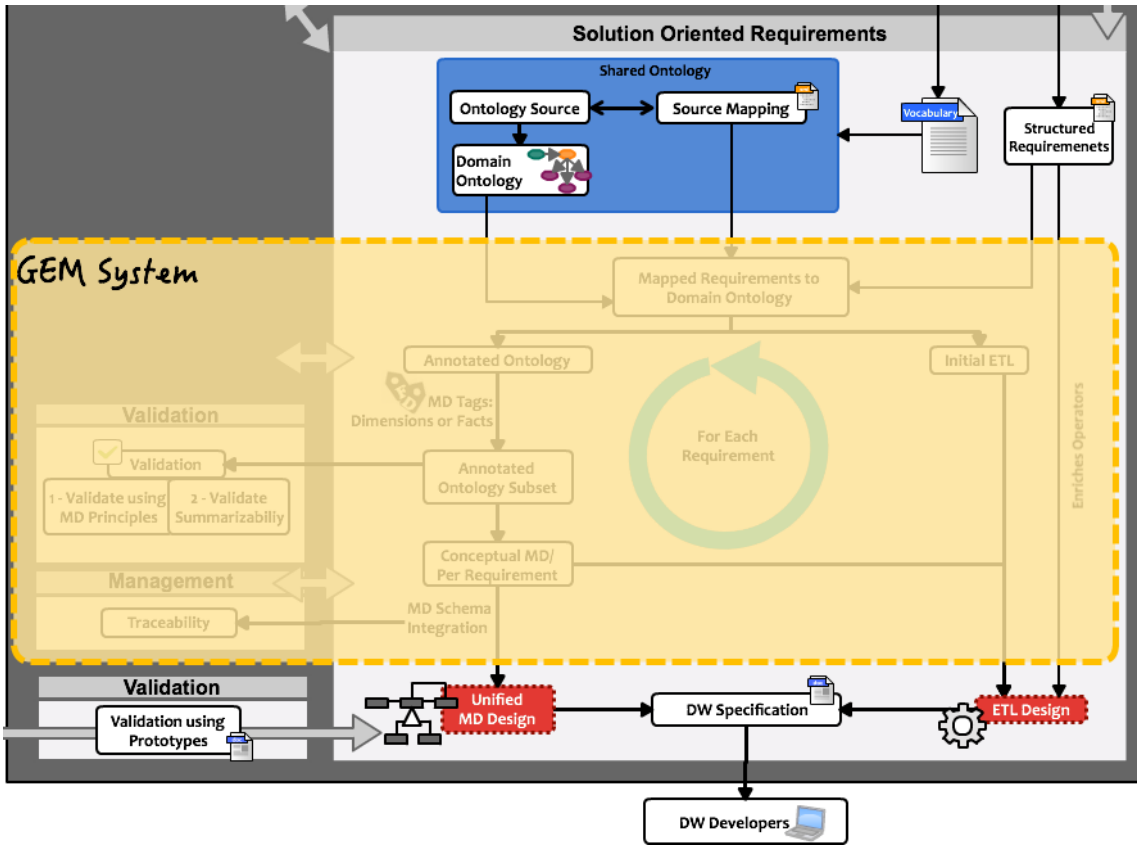


Figure 5.10: Solution-Oriented Requirements & GEM - RE4DSS

Afterwards, all details must be recorded into the final documentation"Data Warehouse Requirements Specification" which is used as the automatic input for the DBMS and also handed to the developers with the aim to create the data warehouse for the information and surveillance system to eliminate Chagas.

Given that the Solution-Oriented Requirements of our approach are comprised of various components including the GEM, this section defines the input files needed for the system to generate respective ETL and MD Designs. Recalling from the previous chapter, GEM framework expects three different files at its input mentioned in the previous subsections:

i. **Structured Requirements** XML file containing requirements extracted from the *Domain Requirements* and *Quality Properties* in the Goals.

ii. **Two components from the Shared Ontology:**

   A. A **Domain Ontology** represented with OWL classes and is later enriched with Vocabulary.

   B. **Source Mappings**, XML file containing mappings of the ontology concepts.

In order to achieve an easier understanding, this section follows the next structure matching the sequence of our approach (See figure 5.10):

i. Structured Requirements

ii. Shared Ontology

   A. Ontology Source & Domain Ontology Complemented With Vocabulary

   B. Domain Ontology

   C. Ontology Validation

   D. Source Mappings

iii. Complementary GEM System

   A. Verification & Management

   B. Final MD & ETL Design

iv. Validation

Structured Requirements

The first input for the GEM system is the Structured Requirements. As mentioned previously in this chapter, these are obtained from the *Domain Requirements* and the *Quality Properties*. Considering that requirements are expressed in a structured way (RSA11), *Domain requirements* and *Quality Properties* are collected and then expressed in an XML file. Consequently multidimensional concepts are identified (measures, dimensions, etc.). Dimensions being the perspectives used to analyze the data (view data in a specific level of detail). Measures being a summary of the analyzed data.

After the goals are collected, it is noticed that the identification of multidimensional concepts inside of these requirements is very simple. For instance:

Running Example - Requirements Presented in XML:

Functional Requirement: "I need to generate graphics showing the temporal variation of evaluated (inspected) houses by department in Argentina."

<measures><concept id = "InspectedHouses"/><measures>

<dimensions><concept id = "Departments"/><dimensions>

<dimensions><concept id = "Countries"/><dimensions>

The XML structure contains different parts which have different tags, see figure 5.11:

Levels:

```
<dimensions>
    <concept id="Species_s_name"/>
</dimensions>
```

Measures:

```
<measures>
    <concept id="EvaluationCoverage">
        <function>
            Inspection_i_inspectedhousesATRIBUT div Inspection_i_evaluatedhousesATRIBUT
        </function>
    </concept>
</measures>
```

Slicers:

```
<slicers> <comparison>
    <concept id="Inspection_i_dateATRIBUT" />
    <function>extracty_year</function>
    <operator>&lt;=</operator>
    <value>2010</value> </comparison>
</slicers>
```

Figure 5.11: XML Defining Structured Requirements: Levels Measures and Slicers - RE4DSS

In addition to the *Domain Requirements*, we may also identify the different tags in *Quality Properties*. This may include: reliability, freshness, security, usability, data integrity, and many other. In the following example, we show that the freshness of the EvaluationCoverage requires to be updated every hour at least (See figure 5.12). Is important to note that quality properties may affect different levels of design. For instance, freshness affects how often the ETL process should run to keep the data warehouse updated according to the requirements, therefore, it should be forwarded to the physical level (Jov11).

In the following sections, we will now focus on the Shared Ontology, which consist of the Ontology Source and the Source Mappings.

```
<measures>
    <concept id="EvaluationCoverage">
        <function>
            Inspection_i_inspectedhousesATRIBUT div Inspection_i_evaluatedhousesATRIBUT
        </function>
        <nfr kind="freshness" format="HH24:MM:SS">&lt;01:00:00</nfr>
    </concept>
</measures>
```

Figure 5.12: XML Defining Structured Requirements: Quality Properties - RE4DSS

Shared Ontology:

Ontology Source

After obtaining the available data sources during the elicitation activity with the WHO stakeholders and selecting the sources needed for the goals to be achieved, the next step involves representing the structure of the sources from which the data will be extracted, transformed and then loaded into the target schema. As mentioned before, the ontology proposed to be used in our approach is the Web Ontology Language (OWL) where the semantics and the constraints of the data sources are captured. The process of construction of the Source Ontology is enriched with the concepts found in the Vocabulary document (complemented with the Use Cases) in order to create a Domain ontology. Let us provide the representation of a relational data source (e.g., tables, attributes, cardinalities etc.) Example extracted from (Jov11), (see figure 4.9): Even though the OWL ontology is capable of representing various types of data stores, for this thesis, the examples of representing relational data sources are provided (e.g., tables, attributes, cardinalities etc.), these examples extracted from the Chagas project contain the following information (see figure 5.13):

i.
- Table **Dwelling** with the attributes:
  - d_dwellingkey INTEGER (primary key)
  - d_latitude INTEGER
  - d_longitude INTEGER
- Table **Inspection** with the attributes:
  - i_inspectionkey INTEGER (primary key)
  - i_infestedIDPD INTEGER
  - i_sprayed INTEGER
  - i_dwellingkey INTEGER (foreign key)

The reference i_dwellingkey represents the relationship between the **Dwelling** and **Inspection** tables including the cardinalities 1..1 for the table **Dwelling** and 0..*MANY for the table **Inspection** (see figure 5.13). This knowledge is represented inside the OWL ontology:

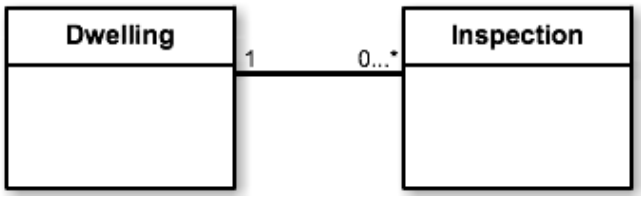


Figure 5.13: Representing Relationships Between Dwelling and Inspection Tables - RE4DSS

- **Source Tables:** The source tables, namely Dwelling and Inspection, are represented as ontology classes using OWL (see figure 5.14).
- **Attributes in Source Tables:** The attributes are represented with different names using Datatype properties. The domain class is the ontology class representing the source table of the attribute, and the range represents the Datatype attribute (see figure 5.15) and (see figure 5.16).
- **Concept Relationships:** As it may be seen from figure 5.13 presented before, the table Inspection has a reference to the source table Dwelling (an example of this in relational databases would be using a foreign key). This relationship is explicitly represented in the ontology by introducing the ObjectProperty. Inside of this, the domain class represents the referencing source table (Inspection), and the range is the one that represents the referenced source table (Dwelling) (see figure 5.17).
- **Cardinalities between Relationships:** In addition, cardinalities must be also be represented. For the relationship between Dwelling and Inspection tables, the defined cardinalities were the following: 1..1 and 0..*. In order to represent the cardinality restriction in the OWL ontology (requiring an integer value), the integer value of '-1' was used to represent 'many' (see figure 5.18).

```
<!--definition of the ontology class that represents source table Dwelling-->
<owl:Class rdf:ID="Dwelling">
...
</owl:Class>

<!--definition of the ontology class that represents source table Inspection-->
<owl:Class rdf:ID="Inspection">
...
</owl:Class>
```

Figure 5.14: Representation of Source Tables in OWL Classes - RE4DSS

During the complete Requirements Engineering process, the Vocabulary is updated whenever new terms arise. So, in order to complement the Ontology Source, we can now use those concepts found from the Use Cases in the Vocabulary and use these to enrich the Domain Ontology. This will finally used as input for the GEM.

```
<!--Definition of the attribute d_dwellingkey, i.e., primary key, of the table Dwelling-->
<owl:DatatypeProperty rdf:about="#Dwelling_d_dwellingkeyATRIBUT">
    <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#int"/>
    <rdfs:domain rdf:resource="#Dwelling"/>
</owl:DatatypeProperty>

<!--Definition of the attribute d_latitude of the table Dwelling-->
<owl:DatatypeProperty rdf:about="#Dwelling_d_latitudeATRIBUT">
    <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#int"/>
    <rdfs:domain rdf:resource="#Dwelling"/>
</owl:DatatypeProperty>

<!--Definition of the attribute d_longitude of the table Dwelling-->
<owl:DatatypeProperty rdf:about="#Dwelling_d_longitudeATRIBUT">
    <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#int"/>
    <rdfs:domain rdf:resource="#Dwelling"/>
</owl:DatatypeProperty>
```

Figure 5.15: Attributes Representation of Dwelling Table using OWL - RE4DSS

```
<!--Definition of the attribute i_inspectiongkey, i.e., primary key, of the table Inspection-->
<owl:DatatypeProperty rdf:about="#Inspection_i_inspectionkeyATRIBUT">
    <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#int"/>
    <rdfs:domain rdf:resource=”#Inspection"/>
</owl:DatatypeProperty>

<!--Definition of the attribute i_infestedIDPD of the table Inspection-->
<owl:DatatypeProperty rdf:about="#Inspection_i_infestedIDPDATRIBUT">
    <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#int"/>
    <rdfs:domain rdf:resource="#Inspection"/>
</owl:DatatypeProperty>

<!--Definition of the attribute i_sprayed  of the table Inspection-->
<owl:DatatypeProperty rdf:about="#Inspection_i_sprayedATRIBUT">
    <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#int"/>
    <rdfs:domain rdf:resource="#Inspection"/>
</owl:DatatypeProperty>

<!--Definition of the attribute i_dwellingkey of the table Inspection that is actually the
foreign key referencing to the table Dwelling→
<owl:DatatypeProperty rdf:about=”#Inspection_i_dwellingkeyATRIBUT">
    <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#int"/>
    <rdfs:domain rdf:resource="#Inspection"/>
</owl:DatatypeProperty>
```

Figure 5.16: Attributes Representation of Inspection Table using OWL - RE4DSS

```
<!- -Definition of the BelongsTo relationship between tables Inspection and Dwelling-->
<owl:ObjectProperty rdf:about="#BelongsTo">
    <rdfs:domain rdf:resource="#Inspection"/>
    <rdfs:range rdf:resource="#Dwelling"/>
</owl:ObjectProperty>
```

Figure 5.17: Source Associations “BelongsTo” Between Tables Dwelling and Inspection - RE4DSS

```
<!-- complete definition of the concept that represent source table Inspection-->
<owl:Class rdf:ID="Inspection"> ...
<!--definition of the maximum cardinality restriction on the property BelongsTo -->
<rdfs:subClassOf>
  <owl:Restriction>
    <owl:onProperty>
      <owl:ObjectProperty rdf:ID="BelongsTo"/>
    </owl:onProperty>
  <owl:maxCardinality rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
  >-1</owl:maxCardinality>
  </owl:Restriction>
</rdfs:subClassOf> ...
<!--definition of the minimum cardinality restriction on the property BelongsTo -->
<rdfs:subClassOf>
  <owl:Restriction>
    <owl:minCardinality rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
  >0</owl:minCardinality>
  <owl:onProperty>
    <owl:ObjectProperty rdf:about="#BelongsTo"/>
  </owl:onProperty>
  </owl:Restriction>
</rdfs:subClassOf>...
```

Figure 5.18: Cardinalities Between Dwelling and Inspection Tables 1..1 and 0..* - RE4DSS

ii. Domain Ontology

As mentioned previously in this chapter, the Chagas project resulted very complex since Chagas is a world-wide disease, and the system must be compatible with all the regions in the world. For instance, one of the challenges faced during the project is that the system gathered information related to dwellings and their peridomicile, however, "peridomicile" does not provide the same meaning in Spain and in Argentina (North-West region). This concluded to be a massive challenge for the technological and biomedical side. Given the situation, the Vocabulary had to include information and definitions related to the Chagas disease, the relationships within the concepts of Chagas disease and WHO, and the meaning of the terms used by the specialists.

For this reason, we have created the Shared Ontology complemented with the vocabulary created from the Use Cases. One of the main contributions from RE4DSS is the proposing not only a Vocabulary document of the homogenized terms with acronyms and synonyms, but to really make this Vocabulary useful, this Vocabulary document is used to create and enrich the Shared Ontology presented in a machine-readable format enabling to automatize the creation of the **Solution-Oriented Requirements** (e.g. by being a direct input to the GEM system) and finally build the the MD design.

For the sake of the representation of the domain ontology, we have used a *"UML like diagrammatic representation"* intended to be machine-readable in contrast to the vocabulary document. The representation is structured in the following packages for readers' easier comprehension (description of packages extracted from (VN14)). (See example in figure 5.21):

- **Health-care Package:** This package contains information regarding the health-care indicators. This package is divided into three sub-packages: treatment package, diagnosis package and drug distribution package. It mainly gathers the information from the diagnosed and treated patients in individual and collective cases.
- **Transmission Interruption Package:** Transmission interruption package is in charge to control and eliminate the Chagas disease by focusing on the vectorial transmission routes. It contains the information regarding the dwelling insecticide application and triatomine bug studies (See figure 5.19).
- **Normative Package:** The normative package contains information regarding the regulations, certifications and assessments. Regulation class represent the concepts of the regulation defined by the WHO that may be from these different types: laws, recommendations and regulations.
- **Systemic Package:** This package provides information regarding the activities and procedure that the countries are implementing in order to control/eliminate the Chagas disease. Countries may provide information about the helping associations for people affected by the disease including the name, city, date, creation of association, etc.
- **Wildlife Package:** This package was added because future plans include forecasting of the evolution of the Chagas disease with regards to the fauna, flora and climatology. For this, the METATRI database was created by Jorge Rabinovich. This database consist of the information regarding the species in the world transmitting the Chagas disease and the climatology.

Model - Intervention Hierarchy - Vectorial.png

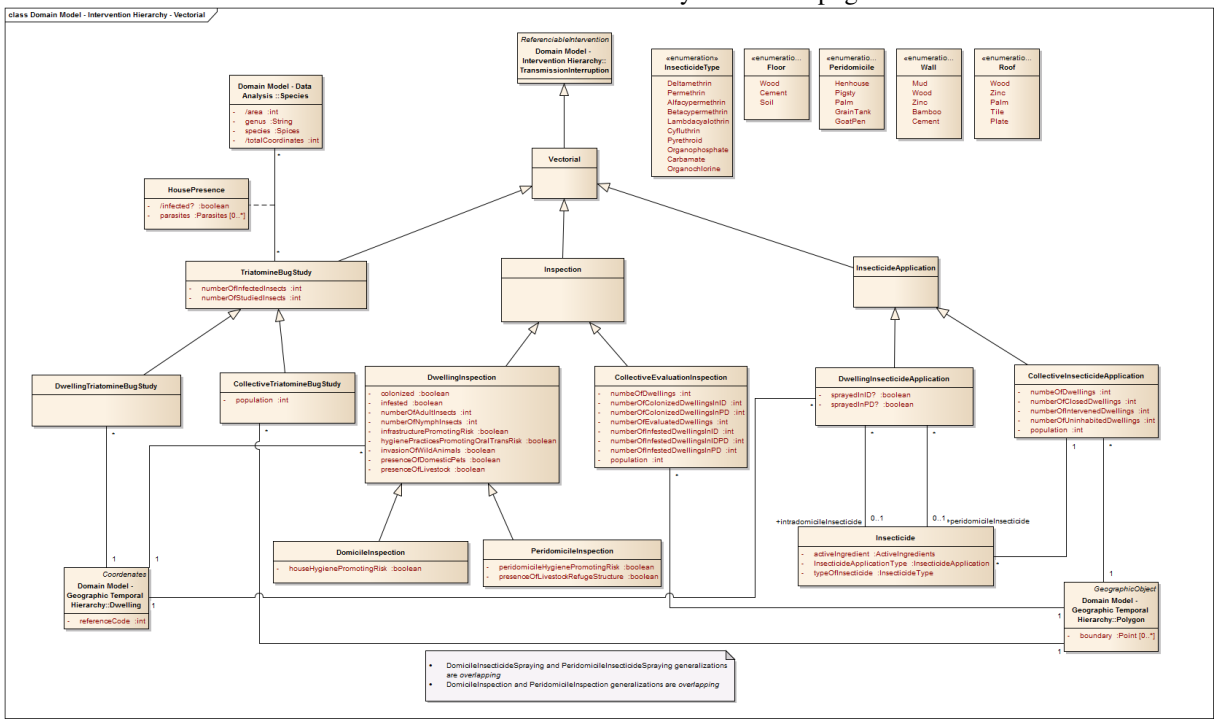


Figure 5.19: Vocabulary Representation - Vectorial Intervention Hierarchy - RE4DSS Reprinted from (VN14)

Model - Geographic and Temporal Hierarchy.png

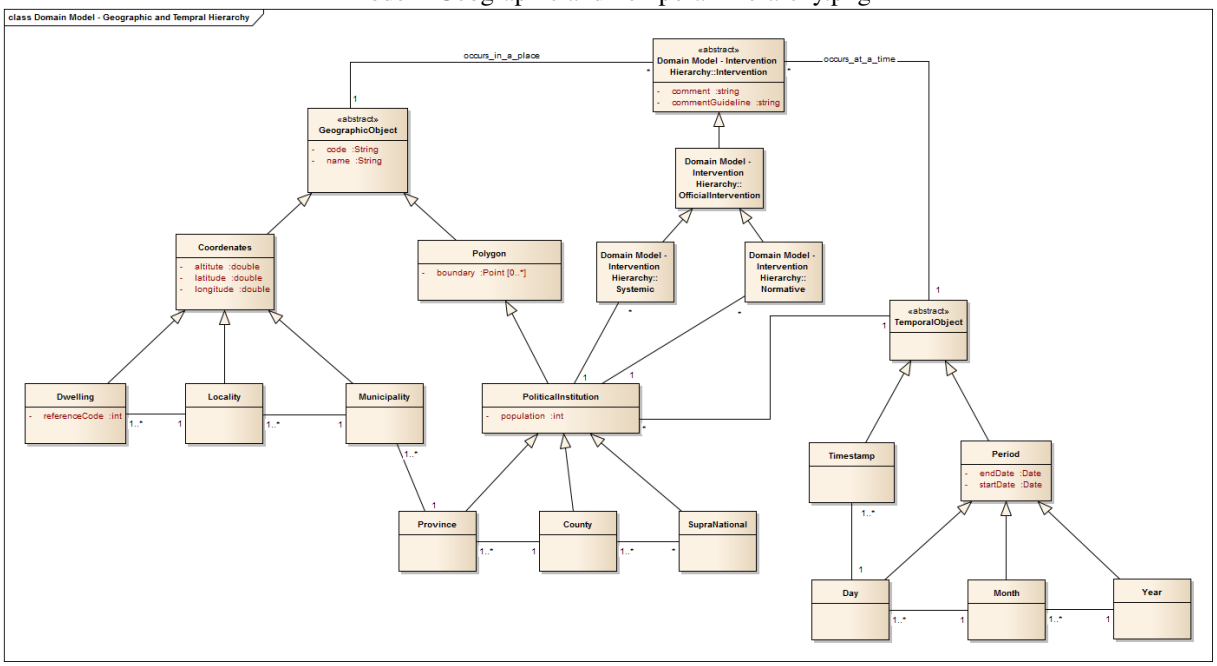


Figure 5.20: Vocabulary Representation - Geographic and Temporal Hierarchy - RE4DSS Reprinted from (VN14)

Model - Intervention Hierarchy.png

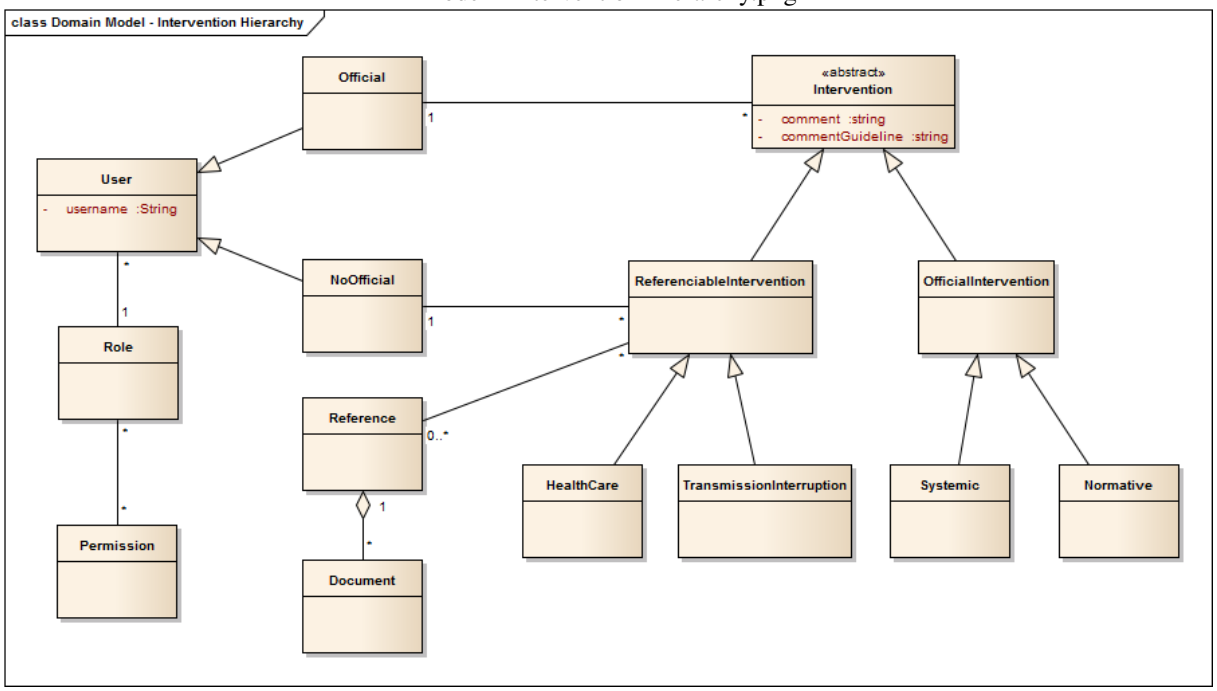


Figure 5.21: Vocabulary Representation - Intervention Hierarchy - RE4DSS Reprinted from (VN14)

Given these packages (see annexe to visualize all packages represented as vocabularies), is important to note that any piece of data entered to the system is characterized by geographical and temporal objects with the aim to visualize the information in maps and diagrams (See figure 5.20).

iii. Ontology Validation

After creating a "UML like diagrammatic representation" of the Shared Ontology, it was still not easy for the WHO stakeholders to understand the agreed common Vocabulary displayed in the UML. Given the situation, in order for them to understand and validate the agreed vocabulary in the UML, we have used prototypes in order to them to have some idea of how the input formats should look like and if the terminology did match to their requirements. Considering that this was a prototype, suggestions were always taken into account (using the negotiation activity) in order to improve the Shared Ontology and to provide a better system fitting their needs to control/eliminate the Chagas. The following example (See figure 5.22) illustrates the Individual Dwelling Inspection where the entomologists may enter information regarding the insects found during the inspection, presence of pets, and more information about the animals and the house.

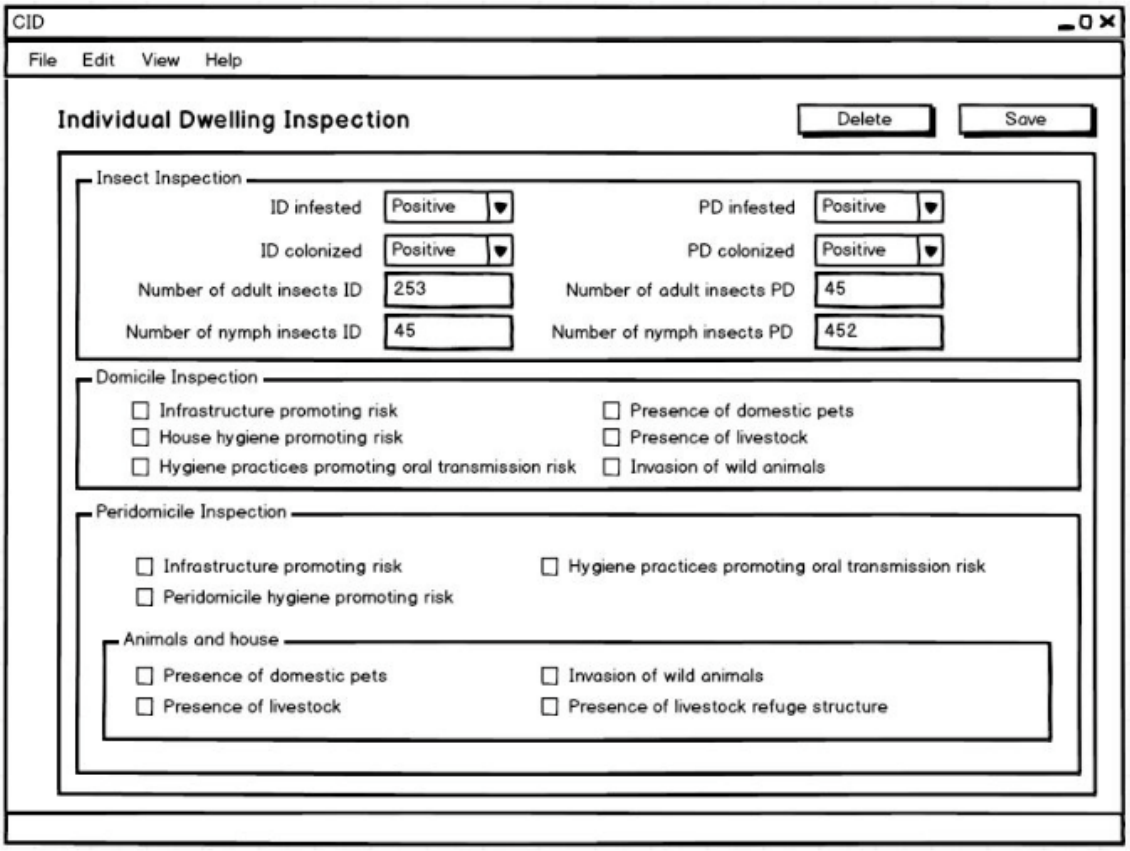


Figure 5.22: Prototype Validation of Vocabulary Shared Ontology - Individual Dwelling Inspection."

iv. Source Mappings

This is the second input needed for the GEM system where the source mappings are defined relating them to the ontology representation of the sources done in the previous step. The source mappings are defined in this case as an XML structure. This consists in mapping each of the concepts from the ontology inside the structure. There are different mappings used to differentiate the kinds of attributes that are in the sources. (See figure 5.15)

A. **Mapping of an ontology class to the source table:** Is identified by the content of the Ontology tag. The ontology concept defined before as (datatype property) which represents the primary key is identified in the (RefOntology tag). In addition, the mapping includes a projection of the attribute representing the primary key (i_inspectionkey). (See figure 5.23)

B. **Mapping of an ontology datatype property to attribute of source:** In this example we have already includes the ontology (datatype property), and it also includes a mapping to the data source table (TableName) and the attribute in the source table (d_latitude). It also contains the attribute representing the primary key of the table that this attribute (d_latitude) belongs, which in this case is d_dwellingkey. (See figure 5.24)

C. **Mapping of an ontology object property to the corresponding datasource relationship:** In the following example, ontology object property represents the association (BelongsTo), in the OntologyTag. The tablename which uniquely defines the association instances is (dwelling). In the projection, the attributes representing both primary keys of the referencing and referenced source are included from both tables. (See figure 5.25)

D. **Mapping of an ontology datatype property:** Contains the RefOntology tag representing the source table referenced by the defined attribute. (See figure 5.26)

```
<OntologyMapping sourceKind="relational">
    <Ontology type="concept">http://www.owl-ontologies.com
        /unnamed.owl#Inspection</Ontology>
    <RefOntology type="property"> http://www.owl-ontologies.com
        /unnamed.owl#Inspection_i_inspectionkeyATRIBUT</RefOntology>
    <Mapping>
        <Tablename>inspection</Tablename>
        <Projections>
            <Attribute>i_inspectionkey</Attribute>
        </Projections>
    </Mapping> </OntologyMapping>
```

Figure 5.23: Source Mapping - Mapping of an ontology class - RE4DSS

```
<OntologyMapping sourceKind="relational">
    <Ontology type="property">http://www.owl-ontologies.com/
        unnamed.owl#Dwelling_d_latitudeATRIBUT</Ontology>
    <Mapping>
        <Tablename>dwelling</Tablename>
        <Projections>
            <Attribute>d_dwellingkey</Attribute>
            <Attribute>d_latitude</Attribute>
        </Projections>
    </Mapping>
</OntologyMapping>
```

Figure 5.24: Source Mapping - Mapping of an ontology datatype property - RE4DSS

```
<OntologyMapping sourceKind="relational">
    <Ontology type="property">
    http://www.owl-ontologies.com/unnamed.owl#BelongsTo</Ontology>
    <Mapping>
    <Tablename>inspection</Tablename>
        <Projections>
            <Attribute>i_dwellingkey</Attribute>
            <Attribute>i_inspectionkey</Attribute>
        </Projections>
    </Mapping>
</OntologyMapping>
```

Figure 5.25: Source Mapping - Mapping of an ontology object property - RE4DSS

```
<OntologyMapping sourceKind="relational">
    <Ontology type="property">http://www.owl-ontologies.com
        /unnamed.owl#Inspection_i_dwellingkeyATRIBUT </Ontology>
    <RefOntology type="concept"> http://www.owl-ontologies.com
        /unnamed.owl#Dwelling</RefOntology> <Mapping>
        <Tablename>inspection</Tablename>
        <Projections>
            <Attribute>i_dwellingkey</Attribute>
        </Projections>
    </Mapping>
</OntologyMapping>
```

Figure 5.26: Source Mapping - Mapping of an ontology datatype property - RE4DSS

Complementary GEM System

The previous sections from Solution-Oriented requirements have covered the three inputs we need for the GEM system, namely the Domain Ontology, the Source Mapping and the Structured Requirements. After this step, the GEM is triggered. In short, the process followed by GEM is: Requirement Verification stage, Requirement Completion stage, Multidimensional Tagging, Operator Identification and Conciliation.

For the purpose of our study, GEM is an assisting system in order to generate the ETL and MD designs in a semi-automatic manner; which clearly complements perfectly the output of our approach. In this Case Study, we will not dig into the details from the GEM system and take the system as a black box just emphasizing the inputs and outputs from the systems. Although, more details regarding the GEM may be found in Chapter 4.

5.5 Automatic Verification & Management Activities

In addition to the management, during the Chagas case study we have performed the validation and verification activities. Verification and Management activities are done in an automatic manner by the GEM, whereas the Validation activity is done using prototypes to validate our ontology, and the final ETL and MD Designs (explained in the last section of this chapter). During the execution of the inputs, the GEM system in order to verify the MD design produced in the Solution-Oriented Requirements, it verifies two main issues:

i. Whether the factual data is arranged in an MD space (e.g., if each instance of factual data is identified by a point in each of its analysis dimensions)

ii. Whether the summarization is correct by examining following conditions proposed in (jLS97):

During the verification, if fails occurs, there must be alternative solutions proposed. Next, these tags are validated to see if these are sound and make MD sense. More details may be found in Chapter 4.

Management of the Decision Support System requirements encompasses lineage to keep traceability capturing information about the sources. For this reason, the Chagas project keeps track of three different aspects.

- **Traceability of Data Sources:** Keeping the traceability of the sources is considered as a key aspect from the management perspective in the Chagas project considering that different sources are available regarding infected population. For instance, incongruities were found between the official data from the governmental sources and the data elicited from the hospitals. In this case, it is crucial to keep track of the sources where the data comes from, by reason of making further analysis and compare statistics regarding the infected people.

  Another example is keeping traceability regarding the official and unofficial sources uploaded to the warehouse. On one hand, Official statistics on patients are gathered as a function day-to-day organizational procedures conducted by the WHO or other agencies cooperating with the government. While, on the other

hand, there are also unofficial statistics gathered by researchers for the express purpose of identifying information about the Chagas and estimating incidence. Consequently, tack of the sources, must be kept all the time.

- **Traceability of the mappings to the integration schema (including schema versioning):** As mentioned before, management of the Decision Support System requirements encompass having traceability of the schema. For this reason, module ORE keeps traceability by considering each of the requirement given by the WHO stakeholders separately and incrementally build the MD schema per requirement and finally it unifies to complete the schema satisfying the entire set of requirements. ORE systematically maintains and keep the trace of everything from the MD design which has been integrated so far (e.g. candidate improvements and alternatives). As mentioned in the previous chapter, ORE works iteratively, and the traceability of the metadata keeps growing every time a new requirement is integrated, along with its feedback if provides the main solid base for the final MD Schema. Same thing is done for the ETL design using the module COAL.

5.6 Final MD & ETL Design

At the latest phase of our approach, all the above stages are meant to run once for each input business required. These are then conciliated into a unified MD design, and single ETL Process design (See figure 5.27). The following example is extracted from the Ontology representation in figure 5.21. The example focuses on the fact "DwellingInsecticideApplication"; Whenever inspectors find evidences of insects in the swellings, it is necessary to start fumigating the area to kill the insects. DwellingInsecticideApplication, aims to define the information of the dwellings, for instance, if its fumigated in intradomicile or peridomicile, or both. In addition, it defines the fumigator/sprayer that has covered the area. Temporal and Geographical locations are also needed to explore the data at a given time, for instance, per region in La Rioja Argentina. Finally, from the example we may notice that it is also necessary to to specify the type of insecticides used in each fumigated dwelling.

Notice that an example of the ETL Design is not provided because it was agreed with the WHO stakeholders that the available data in the sources was having an enormous amount of quality issues in the excel sources such as:

- **Accuracy:** Issues with the locality names in La Rioja "Las Talas" instead of "Las Tablas" and dates "2007" vs. "Julio 2007."
- **Completeness:** Many null values in the inspected dwellings.
- **Uniqueness:** Duplicated entries were found in the inspected dwellings in La Rioja.
- **Consistency:** Some sources had different information regarding the same analysis; for instance, the number of infected people was not the same amount from hospitals and governmental sources for the same date.

Given all these issues, it was agreed that the available sources collected before were not going to be used. However, thanks to RE4DSS, the newly created system is built from a common vocabulary as a standard to be used by

the WHO (entomologists, researchers, etc.) in order to collect new data with the format as input (see prototype in Shared Ontology section).

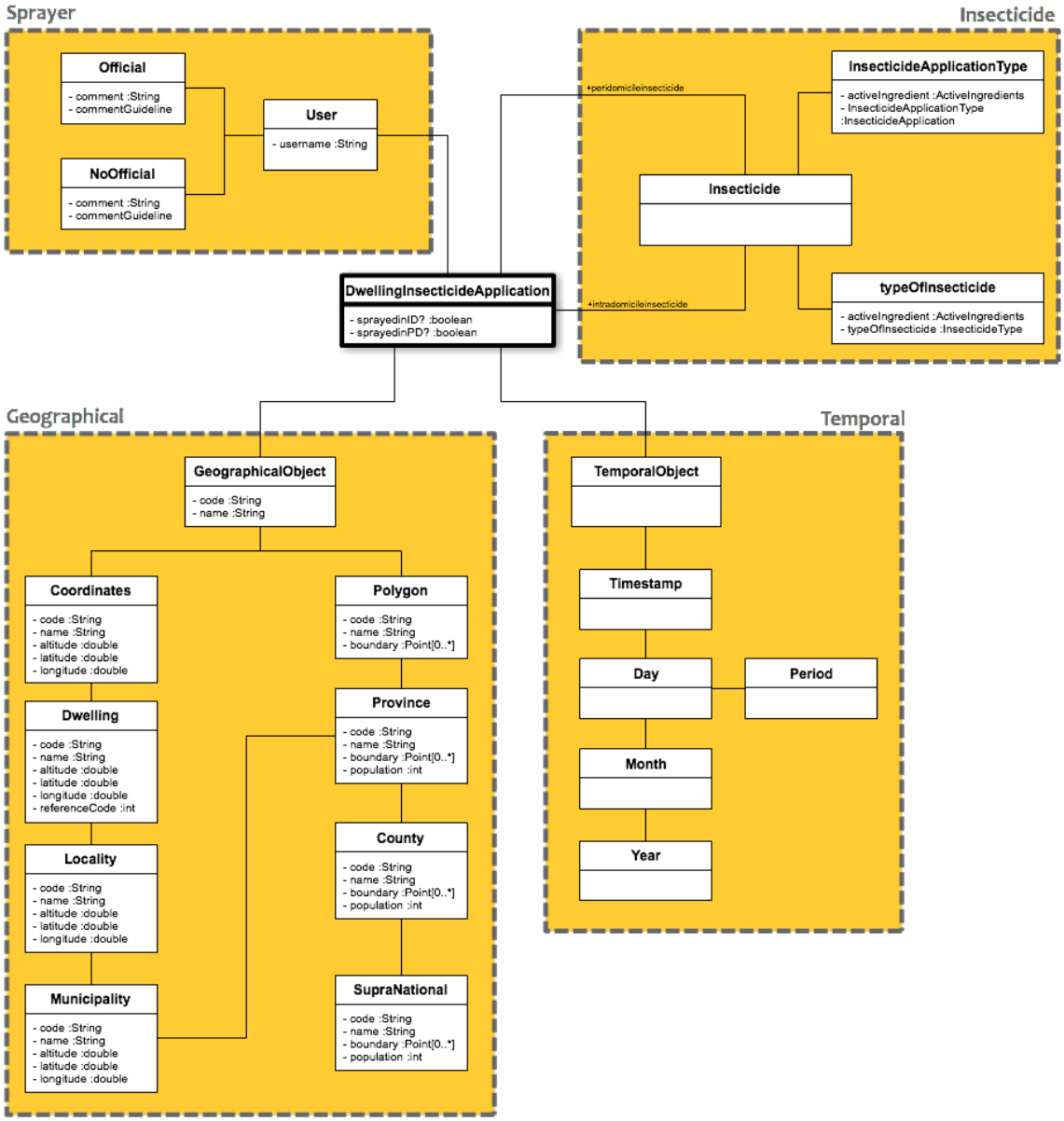


Figure 5.27: Consolidation of Final MD & ETL Design - RE4DSS

5.7 Validation & Exploration Requirements

After finalizing with the MD and ETL Designs, still it was found difficult for WHO stakeholders and Health experts to understand this output presented in the previous example of the schema. For this reason, in order to validate this activity, prototypes/mock-ups from the schema were done in order to validate the design. This facilitated the end-users to provide with feedback, changes, modifications and suggested improvements of the

information they wanted to visualize.

As stated in RE4DSS and the previous chapters *Exploration Requirements* are used at the very end for finally future tuning of the Data Warehouse. Hence, the prototypes were build by using as a reference the gathered identified in the Goal Classification Tree. An example of this is shown in the first prototype (also related to our schema presented in the example) was done in order to validate one of the requirements from the entomologists: "I need to display data about infested dwellings per insecticide used and area." (See figure 5.28)

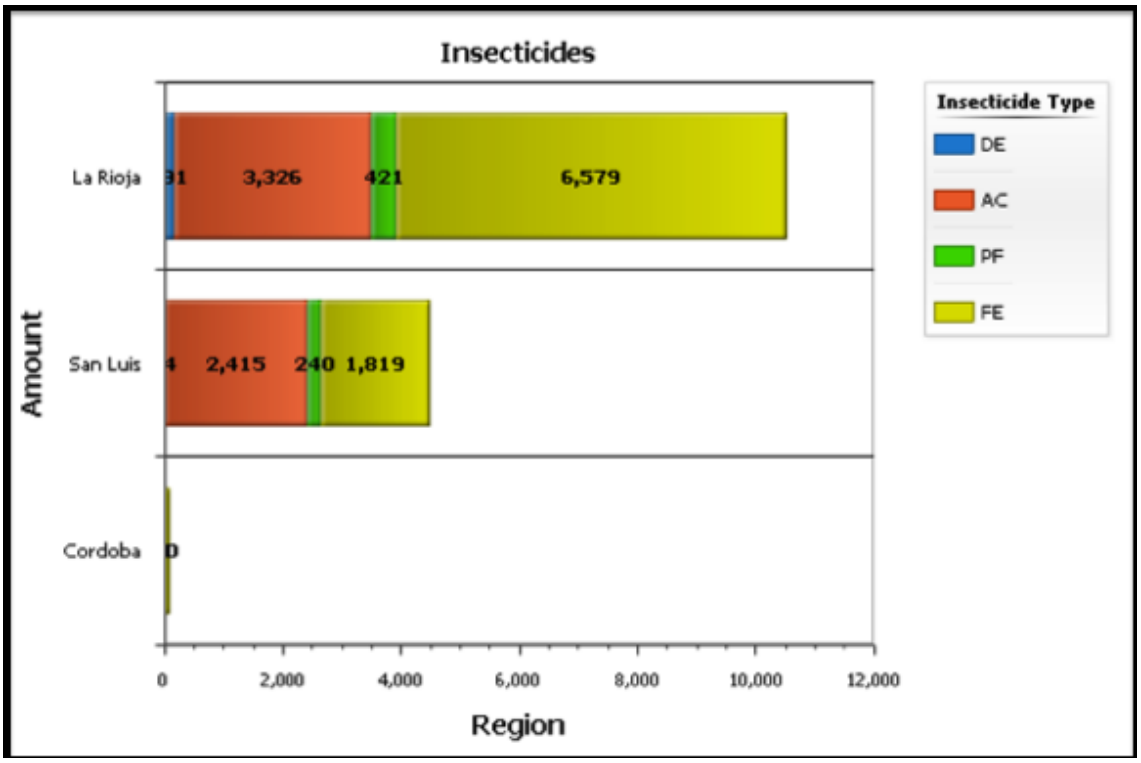


Figure 5.28: Prototype Validation of MD Design - "Data about infested dwellings per insecticide used and area."

In the second example, we validate the following requirement from the entomologist "I want to generate graphics showing the temporal variation of evaluated inspected houses by department every two years for La Rioja (Argentina)" (See figure 5.29).

The final documentation is entitled the "Data Warehouse Specification" which includes the final unified MD design and the ETL design. This final document will be used as input for the DBMS or ETL tool in order to start the implementation.

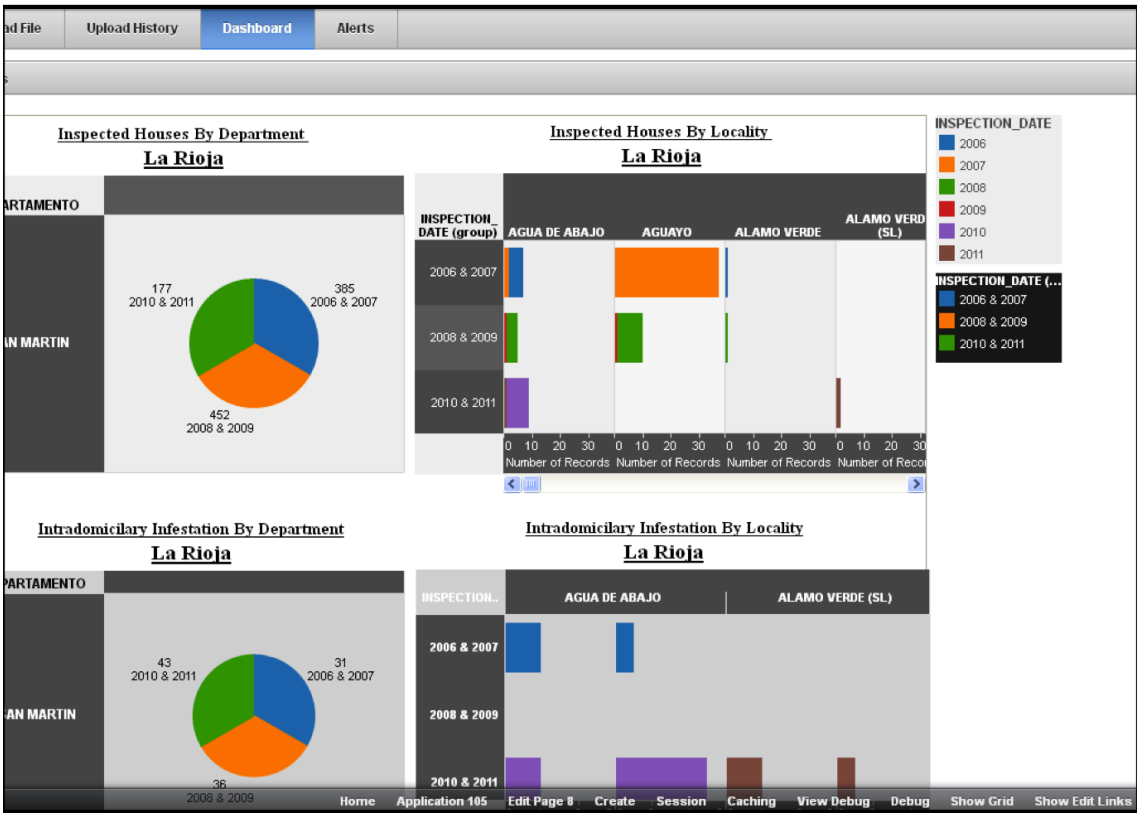


Figure 5.29: Prototype Validation of MD Design - “Graphics showing the temporal variation of evaluated inspected houses by department every two years for La Rioja.”

CHAPTER

6

"There is no real ending. It's just the place where you stop the story."
— Frank Herbert

Conclusions and Further Work

Decision Support Systems have become an integral element of the business processes for numerous companies worldwide. These systems feed employees with knowledge and insights to make the best decision possible according to their objectives. The starting point of every software system is the Requirements Engineering phase, however, one of the greatest failures in Decision Support System projects is setting the wrong requirements by approaching these systems as operational systems; which development is quite different. In consequence, information systems professionals lack of an adequate approach to determine information needs specifically for Decision Support Systems.

Having said this, we conducted a the profound literature classification during our study addressing the complexities of Requirements Engineering for Decision Support Systems. It is demonstrated that existing approaches and methodologies either take for granted the elicitation and negotiation activities to extract the requirements or ignore some specific aspects for Decision Support Systems such as: integration of heterogeneous sources demand complex procedures to control the integration and transformation phases, producing high-quality documentation, the creation of analytical results is expected to be explored across multiple levels, the outcome must always be validated and verified, traceability of must always be kept, and the manual process of translating business needs to the MD and ETL Designs tends to be error-prone and demands several rounds of redesign to satisfy all the stakeholder's requirements. On the basis of the evidence gathered during the literature classification, it seems fair to suggest that leading Information Systems professionals lack an adequate approach to determine information needs specifically for Decision Support Systems.

Therefore, this thesis at hand has consolidated and proposed a block-oriented, systematic Requirements Engineering Approach for Decision Support Systems entitled: *"RE4DSS"* accompanied with the GEM system to translate

these requirements and finally produce the appropriate ETL & MD designs. RE4DSS is integrated with the main structural building blocks from the well recognized framework in the field: “Requirements Engineering Framework" and it aims to make requirements generation phase more manageable while also enabling to deliver the expected decision-support facilities in a systematic manner. A real case study conducted by the Technical University of Catalonia (UPC) in collaboration with the World Health Organization (WHO) demonstrates how RE4DSS approach has been successfully applied during the requirements process to create the ETL and MD Designs of an information surveillance system to control/eliminate the neglected Chagas disease.

From the study in the literature classification and during the creation of RE4DSS, we have identified the key lessons:

- In most cases, failure in Decision Support Systems arises from the poor communication between the developers and the business users (stakeholders). During the case study of the Chagas project, this situation was extremely evident given that software analyst from the UPC were not familiar with the domain of the project (i.e. Trypanosoma cruzi and nifurtimox); but surprisingly, neither was easy to communicate between the entomologists and the doctors. Consequently, no clear understanding was reached between the software analysts and the stakeholders from the WHO (entomologists, researchers, doctors, etc.). Therefore, we propose to create a common vocabulary with homogenized terms in order to avoid linguistic inconsistencies between all stakeholders including developers and business users, enabling them to enroll in an “intelligent conversation” about their true requirements. However, in RE4DSS, the vocabulary is not intended to only to be a document to homogenize terminology. One of our main contributions in RE4DSS is that we have in addition suggested to further use this document to enrich the Shared Ontology in a machine-readable format to open the door to automatize the creation of further processes, such as the creation of MD and ETL Designs. The advantage of this, is that a vocabulary is unlikely to change (non-volatile). So, a Shared Ontology must be able to provide an integrated view of data and describe any data source having a cross-reference structure. The Shared Ontology is finally utilized to define the elements for the corresponding ontology (Domain Ontology), enrich it and by being in a machine-readable format, we can automatize the process by pushing the entry to the GEM to finally create the MD and ETL Designs from all the requirements.
- In our experience, we have perceived that usually business users do not know what they want. Hence, it is sometimes challenging for them to articulate the requirements. The key is to start with “what do you do (and why)” and not ask “what do you want,” this way it is possible to identify the customer needs to bring more benefit to the company. From here, the high-level overview of the Decision Support System is defined and then refined in several sub-goals (i.e. Goal Classification Tree).
- Even if the user has some initial requirements, as the development process takes place and even testing, new needs will arise with new ideas while the user starts to realize how the data warehouse can make reporting capabilities for them. In turn, our approach supports incremental and couples well with agile practitioners.
- Decision Support System projects are considered as long-term projects. And given that new requirements

will arise, changes will take place and evolution must always be maintained. Most of the time, this becomes a challenge for Information System professionals, however, these changes must not be seen as negative given that these additional requirements usually represent the greatest potential value for Decision Support System projects. Therefore, new requirements must be incorporated in a flexible manner. RE4DSS along with GEM provides a management of requirements incorporated to the MD Design by systematically tracing each of the requirements that has been integrated so far. This keeps growing every time a new requirements is integrated, along with its feedback. In addition, lineage is also used to keep traceability capturing information about different aspects such as traceability of Data Sources (i.e. official and unofficial data about infected peopled) and about the mappings to the integration schema per business requirement.

- RE4DSS accompanied with the GEM can verify the if the Solution-Oriented Requirements make MD sense by considering two main aspects: First, the placement of data in a multidimensional space, and second, the correct summarizability of the data.
- Finally, the validation process aims to certify if the input and output activities satisfy the requirements from our stakeholders. To achieve this, it is recommended by practitioners to create prototypes (e.g. prototypes of MD Design and of the Shared Ontology Vocabulary). This validations process is performed jointly with the stakeholders and negotiation can be done to improve before implementation.

In spite of the benefits of RE4DSS, the experience in the joint project with the WHO (Chagas) also revealed that further research in this area may include **traceability**. Up to now, we have limited ourselves to the traceability of the schema. Although, in the future work we will consider crucial to record the traceability of the metadata with different objectives: to reinforce the user assistance in order to provide the user with guidance and to visualize the needed information (e.g. query recommendation, query personalization) and for database tuning by optimizing the part of the system exploited by the user (see “Towards Next Generation BI Systems: The Analytical Metadata Challenge” by (VRPT14)). This may be done through materialized views, adding indexes to the data warehouse using the most queried tables by user action without the administrator intervention and selecting which structures may help us to optimize the system.

Bibliography

[ACNM06] Ben Azvine, Zhan Cui, DD Nauck, and B Majeed. Real time business intelligence for the adaptive enterprise. In *E-Commerce Technology, 2006. The 8th IEEE International Conference on and Enterprise Computing, E-Commerce, and E-Services, The 3rd IEEE International Conference on*, pages 29–29. IEEE, 2006.

[AW05] Aybüke Aurum and Claes Wohlin. *Engineering and managing software requirements*, volume 1. Springer, 2005.

[BDGM08] Paola Britos, Oscar Dieste, and Ramon Garcia Martinez. Requirements elicitation in data mining for business intelligence projects. In David Avison, GeorgeM. Kasper, Barbara Pernici, Isabel Ramos, and Dewald Roode, editors, *Advances in Information Systems Research, Education and Practice*, volume 274 of *IFIP - The International Federation for Information Processing*, pages 139–150. Springer US, 2008.

[BEK04] Bartosz Bebel, Johann Eder, Christian Koncilia, Tadeusz Morzy, and Robert Wrembel. Creation and management of versions in multiversion data warehouse. In *SAC*, pages 717–723, 2004.

[Bib14] The DBLP Computer Science Bibliography. http://dblp.uni-trier.de, 2014. [Online; accessed 19-July-2014].

[BPG04] Paolo Bresciani, Anna Perini, Paolo Giorgini, Fausto Giunchiglia, and John Mylopoulos. Tropos: An agent-oriented software development methodology. *Autonomous Agents and Multi-Agent Systems*, 8(3):203–236, 2004.

[CCS93] E.F. Codd, S.B. Codd, and C.T. Salley. *Providing OLAP (On-line Analytical Processing) to User-analysts: An IT Mandate*. Codd & Associates, 1993.

[CD97] Surajit Chaudhuri and Umeshwar Dayal. An overview of data warehousing and olap technology. *ACM Sigmod record*, 26(1):65–74, 1997.

[Cip97] Vinko Cippico. Comparison of the decision support systems and the transaction support system development methodologies. In *ADBIS*, pages 416–426, 1997.

[Cla95] Tom Clancy. The standish group report. http://www.projectsmart.co.uk/docs/chaos-report.pdf, 1995. [Online; accessed 01-February-2014].

[CMM97] R. Connelly, R. McNeill, and R. Mosimann. *The Multidimensional Manager: 24 Ways to Impact Your Bottom Line in 90 Days*. Cognos Incorporated, 1997.

[Coc01] A. Cockburn. *Writing Effective Use Cases*. Agile Software Development Series. Addison-Wesley, 2001.

[DBL08] *Informal Proceedings of the Second International Workshop on Business Intelligence for the Real-Time Enterprise, BIRTE 2008, in conjunction with VLDB'08, August 24, 2008, Auckland, New Zealand*, 2008.

[EBMJ12] Neil A Ernst, Alexander Borgida, John Mylopoulos, and Ivan J Jureta. Agile requirements evolution via paraconsistent reasoning. In *Advanced Information Systems Engineering*, pages 382–397. Springer, 2012.

[fDCP11] Centers for Disease Control and CDC Prevention. Neglected tropical diseases. http://www.cdc.gov/globalhealth/ntd/, 2011. [Online; accessed 21-July-2014].

[Gar12] Natheer K Garaibeh. Dss development and agile methods: Towards a new framework for software development methodology. *Power*, 7(13):28, 2012.

[GOIM00] Jesús García, María-José Ortín-Ibáñez, Begoña Moros, Joaquín Nicolás, and José Ambrosio Toval Álvarez. Toward use case and conceptual models through business modeling. In *ER*, pages 281–294, 2000.

[GR09] Matteo Golfarelli and Stefano Rizzi. *Data warehouse design: Modern principles and methodologies*. McGraw-Hill, Inc., 2009.

[GRG08] Paolo Giorgini, Stefano Rizzi, and Maddalena Garzetti. Grand: A goal-oriented approach to requirement analysis in data warehouses. *Decision Support Systems*, 45(1):4–21, 2008.

[GRT11] Matteo Golfarelli, Stefano Rizzi, and Elisa Turricchia. Modern software engineering methodologies meet data warehouse design: 4wd. In *DaWaK*, pages 66–79, 2011.

[IBM09] IBM. Gathering and documenting your bi business requirements. IBM, 2009.

[Iee90] E. Iee. IEEE Std 610.12-1990(R2002). *IEEE Standard Glossary of Software Engineering Terminology*, 1990.

[IIS02] William H Inmon, Claudia Imhoff, and Ryan Sousa. *Corporate information factory*. John Wiley & Sons, 2002.

[Inm96] William H. Inmon. *Building the data warehouse*. Wiley computer publishing. Wiley, New York, NY [u.a.], 2. ed edition, 1996.

[Int13] The Standish Group International. Chaos manifesto 2013. http://www.projectsmart.co.uk/docs/chaos-report.pdf, 2013. [Online; accessed 01-February-2014].

[IP01] Burbano Insfran and Pastor. Requirements engineering applied to user interface conceptual modeling. *JIRA 2001*, 2001.

[jLS97] Hans j. Lenz and Arie Shoshani. Summarizability in olap and statistical data bases. pages 132–143. IEEE Computer Society, 1997.

[Jov11] Petar Jovanovic. Integration of multidimensional and etl design. http://upcommons.upc.edu/pfc/bitstream/2099.1/12690/1/Master%20thesis_Petar_Jovanovic.pdf, 2011. [Master in Information Systems; accessed 18-February-2014].

[JRSA12a] Petar Jovanovic, Oscar Romero, Alkis Simitsis, and Alberto Abelló. Integrating etl processes from information requirements. In Alfredo Cuzzocrea and Umeshwar Dayal, editors, *DaWaK*, volume 7448 of *Lecture Notes in Computer Science*, pages 65–80. Springer, 2012.

[JRSA12b] Petar Jovanovic, Oscar Romero, Alkis Simitsis, and Alberto Abelló. Ore: an iterative approach to the design and evolution of multi-dimensional schemas. In *DOLAP*, pages 1–8, 2012.

[Kim98] Ralph Kimball. *The data warehouse lifecycle toolkit: expert methods for designing, developing, and deploying data warehouses*. John Wiley & Sons, 1998.

[LB02] Beum-Seuk Lee and Barrett R. Bryant. Contextual natural language processing and daml for understanding software requirements specifications. In *COLING*, 2002.

[Luh58] Hans Peter Luhn. A business intelligence system. *IBM Journal of Research and Development*, 2(4):314–319, 1958.

[MLT] Jose-Norberto Mazón, Jens Lechtenbörger, and Juan Trujillo. Solving summarizability problems in fact-dimension relationships for multidimensional models.

[MLT09] Jose-Norberto Mazón, Jens Lechtenbörger, and Juan Trujillo. A survey on summarizability issues in multidimensional modeling. *Data Knowl. Eng.*, 68(12):1452–1469, 2009.

[MPT07] Jose-Norberto Mazón, Jesús Pardillo, and Juan Trujillo. A model-driven goal-oriented requirement engineering approach for data warehouses. In *Advances in Conceptual Modeling–Foundations and Applications*, pages 255–264. Springer, 2007.

[MSS] Sandeep Mathur, Girish Sharma, and AK Soni. Requirement elicitation tecniques for datawarehouse review paper.

[MT12] Alejandro Maté and Juan Trujillo. A trace metamodel proposal based on the model driven architecture framework for the traceability of user requirements in data warehouses. *Inf. Syst.*, 37(8):753–766, 2012.

[Org14] World Health Organization. About who. http://www.who.int/about/en/, 2014. [Online; accessed 11-July-2014].

[PC02] Fá Rilston Silva Paim and Jaelson Castro. Enhancing data warehouse design with the nfr framework. In Oscar Pastor and Juan SÃ¡nchez DÃaz, editors, *WER*, pages 40–57, 2002.

[PCF95] N. Pendse, R. Creeth, and Businessintelligence (Firm). *The OLAP Report: Succeeding with On-line Analytical Processing*. Business Intelligence, 1995.

[PdC03] Fá Rilston Silva Paim and Jaelson Freire Brelaz de Castro. Dwarf: An approach for requirements definition and management of data warehouse systems. In *Requirements Engineering Conference, 2003. Proceedings. 11th IEEE International*, pages 75–84. IEEE, 2003.

[Ped09] Torben Bach Pedersen. Multidimensional modeling. In *Encyclopedia of Database Systems*, pages 1777–1784. 2009.

[Poh10] Klaus Pohl. *Requirements Engineering - Fundamentals, Principles, and Techniques*. Springer, 2010.

[Pro12] Tropos Project. Journal of software engineering and applications. http://file.scirp.org/Html/3-$9301547_26213.html, 2012. [Online; accessed 01-April-2014]$.

[Pro14] Tropos Project. Requirements engineering with tropos. http://www.troposproject.org/node/120, 2014. [Online; accessed 01-April-2014].

[RA10] Oscar Romero and Alberto Abelló. A framework for multidimensional design of data warehouses from ontologies. *Data Knowl. Eng.*, 69(11):1138–1157, 2010.

[RA13] Oscar Romero and Alberto Abelló. Open access semantic aware business intelligence. In *eBISS*, pages 121–149, 2013.

[RB12] Ana Respicio and Frada Burstein, editors. *Fusing Decision Support Systems into the Fabric of the Context, 16th IFIP Working Group 8.3 International Conference on Decision Support Systems (DSS), Anavissos, Greece, June 27-30, 2012*, volume 238 of *Frontiers in Artificial Intelligence and Applications*. IOS Press, 2012.

[Rom11] Oscar Romero. Database design course. University Lecture. Universitat Politecnica de Catalunya (UPC-BarcelonaTech), 2011.

[RSA11] Oscar Romero, Alkis Simitsis, and Alberto Abelló. Gem: Requirement-driven generation of etl and multidimensional conceptual designs. In *DaWaK*, pages 80–95, 2011.

[Sch14] Google Scholar. http://scholar.google.es, 2014. [Online; accessed 19-July-2014].

[SDC10] Lorenzo. Savioli, Denis. Daumerie, D. W. T. Crompton, Patricia. Peters, and World Health Organization. *Working to overcome the global impact of neglected tropical diseases : first WHO report on neglected tropical diseases*. World Health Organization Geneva, Switzerland, 2010.

[SS07] Timos K. Sellis and Alkis Simitsis. Etl workflows: From formal specification to optimization. In Yannis E. Ioannidis, Boris Novikov, and Boris Rachev, editors, *ADBIS*, volume 4690 of *Lecture Notes in Computer Science*, pages 1–11. Springer, 2007.

[SS14] Hergovind Singh and Harsh Vardhan Samalia. A business intelligence perspective for churn management. *Procedia - Social and Behavioral Sciences*, 109(0):51 – 56, 2014. 2nd World Conference on Business, Economics and Management.

[Sta14] WHO Staff. Chagas disease (american trypanosomiasis). http://www.who.int/neglected$_d$*iseases*/*diseases*/*chagas*/*en*/, 2014. [*Online*; *accessed* 11 − *July* − 2014].

[Str02] Bernhard Strauch. *Entwicklung einer Methode fur die Informationsbedarfsanalyse im Data Warehousing*. 2002.

[SWW11] Florian Stroh, Robert Winter, and Felix Wortmann. Method support of information requirements analysis for analytical information systems - state of the art, practice requirements, and research agenda. *Business Information Systems Engineering*, 3(1):33–43, 2011.

[Tag03] Luca Vetti Tagliati. *UML e ingegneria del software: dalla teoria alla pratica*. Tecniche Nuove, 2003.

[TCM02] Marco Toranzo, Jaelson Castro, and E. Mello. Uma proposta para melhorar o rastreamento de requisitos. In Oscar Pastor and Juan Sanchez Diaz, editors, *WER*, pages 194–209, 2002.

[The10] The Economist. A special report on managing information: Data, data everywhere. *The Economist*, February 2010.

[Val37] Paul Valéry. Notre destin et les lettres. 1937.

[vL04] Axel van Lamsweerde. Goal-oriented requirements enginering: A roundtrip from research to practice. In *RE*, pages 4–7. IEEE Computer Society, 2004.

[VN14] Jaume Vinas Navas. Building a data warehouse for the global who information and surveillance system to control/eliminate chagas disease captura de requisits per a un sistema d'informacio i vigilancia a la oms per a la malaltia de chagas. https://upcommons.upc.edu/handle/2099.1/20800?locale=es, 2014. [Master in Computing; accessed 21-February-2014].

[VRPT14] Jovan Varga, Oscar Romero, Torben Bach Pedersen, and Christian Thomsen. Towards Next Generation BI Systems: The Analytical Metadata Challenge. In *DaWaK*, 2014. In Press.

[Who10] Who. Chagas disease: control and elimination. Sixty Third World Health Assembly, 2010.

[WHO13] *Sustaining the drive to overcome the global impact of neglected tropical diseases: second WHO report on neglected diseases*. WHO/HTM/NTD/2013.1. World Health Organization, Geneva, advanced proof edition, 2013. Copies of materials can be requested through: NTD Information Resource Centre E-mail: ntddocs@who.int.

[Woo97] Wayne. Woodruff. Requirements management for small organizations, 1997.

[WS03] Robert Winter and Bernhard Strauch. A method for demand-driven information requirements analysis in data warehousing projects. In *HICSS*, page 231, 2003.

			Requirements Artifacts			Core Activities				
Investigated Literature	ID	Description of the proposed practice or RE in DSS matching Pohl's Framework	Goals	Scenarios	Solution-Oriented Requirements	Documentation	Elicitation	Negotiation	Management	Validation
[ACNM06]	1	A common business vocabulary provides an integrated view of data and is rich enough to describe any data source.			**Business Vocabulary**	✓				
[SWW11]	2	An increased demand for the comprehensibility of the documentation exists. Moreover, in order to avoid linguistic inconsistencies the desire is expressed to homogenize used terms (i.e. dimensions)			**Glossary**	✓				
[GR09]	3	Proposes an informal approach that requires glossaries to support designers in the conceptual and design phase.			**Glossary**	✓				
[PdC03]	4	Project glossary: Organizes BI terminology related to problem domains. Enhance understanding among all parties.			**Glossary**	✓				
[BDGM08]	5	Glossary proposal: It is necessary to identify definitions, acronyms and abbreviations for establishing lexis to be shared among all persons related to the DM-BI project.			**Glossary**	✓				
[GOIM+00]		The glossary will have a hypertext (cross-references) structure, in order to maintain the traceability relationships from the business processes			**Glossary**					
[ACNM06]	6	A shared ontology is a common, agreed vocabulary of both domain users and developers. This ontology is as a superset of vocabulary, which is rich enough to describe any enterprise data sources.			**Vocabulary Shared Ontology**	✓				
[Str02]	7	Represents which data is related to which concepts. Represents which concepts are homonyms or synonyms, etc. on an aggregate level.			**MD Vocabulary**	✓				
[GR09]	8	Specification of facts and preliminary workload are crucial for the conceptual design for the data mart: Such as identifying facts, dimensions and measures by queries and interaction with the user.			**Conceptual Model**	✓	*Identify: Measures, Dimensions, Facts*			
[WS03]	9	In order to create results that are reusable in subsequent phases of the data warehouse development project, models for information requirements representation should be aligned with or even be identical with models used for data warehouse design. Therefore project managers prefer conceptual models for multidimensional data.			**Conceptual Model**	✓				
[IP01]	11	Conceptual Schema Generation Approach: Combines Use Case specification with Function Refinement Tree. Using a Function Decomposition Table or CRUD matrix where the top rows are Use Cases and left column classes are from sequence diagram. All this done to finally construct the conceptual schema.		✓	**CRUD Matrix Conceptual Schema**	✓				
[WS03]	10	As one of the results of the information requirements analysis, a document 'Information map' should be created that represents where data are sourced from and the organizational units using which data.			**Information Map**	✓			✓	
[PdC03]	14	Non-Functional Requirements Specification: Complements the use case specifications, describing non-functional requirements not covered by the use case model, as well as design constraints and other restrictive factors (ex. legal issues, tool limits).			**Non-Functional Requirements Specification**	✓				
[PdC03]	15	Multidimensional Requirements Specifications: Describes the multidimensional portrait, including restrictions; factual and dimensional information; granularity, additivity and conformance of multidimensional requirements.			**Multidimensional Requirements Specifications**	✓				
[PdC03]	16	Business Rules Specification: Register business rules that regulate data warehouse and data mart functionalities.			**Business Rules Specification**	✓	*Elicit Business Rules*			
[Kim98]	17	Formal documentation must be to validate findings with users. Findings can be documented including a high-level enterprise data warehouse bus matrix (Business Process vs. Dimensions).			**Bus Matrix**	✓				✓
[Gar12]	18	DSS projects which have an increasingly volatile business environment, firms are asking for lighter weight, faster and more agile software development processes that can accommodate the inevitable ongoing changes to requirements							*Agile*	
[RB12]	19	Practitioners agree that agile style development is most appropriate for BI systems and had been the main approach used for a long-time.							*Agile*	
[EBMJ12]	20	Proposes a systematic approach to agile requirements evolution where it is easy to change requirements and automatically evaluate the consequences of these changes.							*Agile*	
[SWW11]	21	DSS are not meant to be complete systems but they are expected to be under modification and evolution. Accordingly, it is useful to introduce the process perspective (in addition to the project view), based on sub-activities of the project-specific management, which manages information requirements continuously and across projects.							*Process Perspective*	
[Kim98]	22	Differentiate between project and program requirements. While the mechanics are							*Program*	

Figure 6.1: Literature Analysis of DSS Requirements Engineering

		similar, there are differences in terms of the participants, breadth and depth of details.								*Requirements*	
[Kim98]	38	For the DW/BI requirements analysis there are two passes: gather high-level program requirements followed by a more detailed project-specific requirements.					✓			*Detailed project-specific requirements*	
[WS03]	23	Since limited resources allow only selected unsatisfied information requirements to be covered by the DW system, and evaluation criteria have to be developed which then allow for assigning priorities.				*Evaluation Criteria*				*Prioritization*	
[GR09]	24	Collaboration of designers and end users aims at specifying goals and limitations of the data warehouse system and assessing priorities.	*Collaboration of designers and end users*		✓		*Elicit goals and limitations*			*Prioritization*	
[SWW11]	25	Prioritization of information requirements is an essential feature								*Prioritization*	
[Kim98]	26	Two-by-two prioritization grid for the requirements. X-axis as feasibility and Y-axis as potential business impact. Begin with requirements from the upper-left quadrant.	✓	✓	✓	✓				**2-by-2 Prioritization Grid**	
[IBM09]	27	BI Specification Practices:			✓	✓				**-Prioritization Matrix -Risk Mgmnt Plan -Change Mngmt Plan**	
[SWW11]	28	For the traceability of information requirements, a change of history is maintained.				✓				*Traceability*	
[IBM09]	29	Keys to requirements traceability: Unique Numbering, Cross Referencing, Referred To During Project, Key To Testing, Simple to Understand, Easy to Search For.				✓				*Traceability*	
[PdC03]	30	Traceability Matrices to show requirements dependencies: enable cross-reference analysis between requirements (ex. functionalities versus facts; facts versus dimensional attributes)			✓	✓				**Traceability Matrix with Cross-Reference**	
[PdC03]	31	In data warehouse environments, traceability and change management must be carried out in both requirements and database architecture in order to clear up what impact the underlying change will have in the multidimensional schema.			✓					*Traceability and Change Management*	
[PdC03]	32	Changes in requirements affect the database model in both Data Mart and Data Warehouse visions. Most relational databases offer tools such as Microsoft SQL Server and Oracle 9i Developer Suite to search for affected attributes in the database, and discover how many (and at what extent) database elements are influenced by the change.								*Tools for Change Management*	
[PdC03]	34	Surrounding the requirements definition, a backbone phase named Requirements Management Control is placed to perform permanent quality assessment of requirements evolution.				✓				*Requirements Management Control*	
[GR09]	39	Requirements will play a vital role ensuring that the integration with previous information systems is as smooth as possible.								*Integration with other systems*	✓
[PdC03]	35	The initially elicited (raw) Data Mart requirements traverse a sequence of iterations, along which requirements are analyzed, negotiated within participants, registered and conformed to a broader data warehouse specification.			✓	**Data Warehouse Specification**	✓	✓		*Requirements Iteration*	
[PdC03]		Data warehouse specification, we emphasizes on the following: (a) avoid redundancy and ambiguity between requirements that oversees the entire data warehouse; (b) allow common dimensional aspects (c) in conjunction with a use case approach, promote reusability of agreed knowledge in the project; (d) improve user interface and data content consistency; (e) enable OLAP operations involving multiple fact tables, etc.			✓	**Data Warehouse Specification**		*Overlapping Requirements*			
[PdC03]	36	Requirements Conformance: specific phase where data warehouse must be prepared to cope with continuously evolving requirements. A dimension is said to be "conformed" when it means the same to every fact table. Similarly a fact conforms if the same terminology is used across data marts to represent its content. The principle is done by extending this concept to an utmost level of abstraction on which all-common system requirements are conformed			✓	**Data Warehouse Specification**					Requirements Conformance
[PdC03]	37	Requirements Conformance: in conjunction with a use case approach, promote reusability of agreed knowledge in the project, thus enhancing quality;		**Use Cases**		✓		*Reusability of the agreed knowledge*			Requirements Conformance
[MLT08]	40	A normalization process is carried out in order to obtain a MD model that ensures summarizability while accurately capturing the expressiveness of the demanded real-world situation. The output of such a process is a conceptual model constrained to those elements and relationships that do not violate summarizability.			✓	**Conceptual Model**					**Normalization Process**
[SWW11]	41	The increased integration of prototypes may facilitate the specification and, in particular, the validation of information requirements for analytical information systems.				✓					Prototype
[SWW11]	42	Validation of the documented information need is carried out through a prototype implementation (i.e. reports).				✓					Prototypes
[PdC03]	43	The validation phase corrects potential pitfalls. Review sessions, together with prototyping prove to be an effective strategy to detect and remove defects in the target application, before they become part of the delivered data mart package.				✓					Review Sessions + Prototyping
[PdC03]	44	Prototyping to simulate OLAP Behavior. Stakeholders may consolidate how their ideas may be allocated to the multidimensional solution.				✓					Prototypes

Figure 6.2: Literature Analysis of DSS Requirements Engineering

[PdC03]	45	An OLAP interface prototype helps users to recognize architectural aspects derived from defined requirements and validate the solution.				✓				OLAP interface Prototypes
[SWW11]	46	Validation of the documented information requirements is based on interviews of IT and business staff.				✓				Interviews
[PdC03]	47	Rilston, Paim and Castro defined a checklist for data warehouse requirements: (See Table below document)				✓				Requirement's Checklist
[PdC03]	48	Revision Report: A simple report to hold the actions agreed after a requirements validation session.				✓		✓		**Revision Report**
[PdC03]	50	Guarantee the correctness of aggregation results for combination of facts and dimensions, also known as 'Summarizability'				✓				*Guarantee Summaizability*
[PdC03]	51	Data warehouse applications encompass evolution that can impact the overall validity of derived data.				✓			✓	*Validate Derived Data*
[PdC03]		Add external reviewers to validation process as a means of bringing an unbiased perspective about the data warehouse solution. This practice improves defect detection efforts and suggested new OLAP analysis over data.								*Add external reviewers to validation process*
[PdC03]		When problems are located, the validation team immediately attaches a list of actions in response to each problem. The development process returns to the specification phase.				*List of actions*				*When problems detected return*
[IBM09]	52	BI Validation Techniques:				✓				-Quality Review -Peer Review -Customer Review -IT Review -Phase Gate -Project Sponsor -Requirement Presentation
[PdC03]	53	There is a lack of higher-level requirements vision in user needs.	*High-level requirement vision*			✓				
[PdC03]	12	Data Warehouse Vision: Describes data warehouse requirements, descriptions of motivation, objectives and problem issues	**Data Warehouse Vision**			✓				
[PdC03]	13	Data Mart Vision: Collects high-level data mart user needs, as well as features and actors to support such needs.	**Data Mart Vision**			✓				
[GR09]	54	Proposes an approach based on Tropos (based on i*) formalism for user requirement analysis.	**Tropos**			✓				
[MSS]	55	Tropos methodology emphasis on early requirement analysis based on the goals of stakeholders, which captures the details of "why" a DWH need to be developed.	**Tropos**			✓				
[PC02]	56	DW-NFR Framework uses Softgoal Interdependency Graph that represents the influence or interdependency of one softgoal on another. A softgoal is said to be "satisfied" when there is sufficient positive evidence and little negative evidence against it.	**Softgoal Inter-dependency Graph**			✓				
[IP01]	90	Conceptual Schema Generation Approach: Combines Use Case specification with Function Refinement Tree. Using a Function Decomposition Table or CRUD matrix where the top rows are Use Cases and left column classes are from sequence diagram. All this done to finally construct the conceptual schema.	**Function Refinement Tree**	**Use Cases**		✓				
[MPT07]	91	To ease the task of eliciting goals for DWs, three kind of goals are classified: 1-Strategic goals (Highest abstraction level, i.e. "increase sales"), 2-Decision goals (Medium abstraction level helping reach strategic goals, i.e. "open new stores"), and 3-Information goals (Lowest abstraction level helping reach decision goals on the 'how', i.e. "analyze customer purchases")	**Goals' Classification**			✓				
[PdC03]	92	The framework Non-Functional Requirements (NFR) allows for the definition and reuse of design catalogues which organize knowledge and methods about particular NFR.	**Non-Functional Requirements Framework (NFR)**			✓				
[BDGM08]	57	Identify the objective of the DM-BI project and its motivation to characterize what customer needs.	*Identify Customer Needs*							
[RB12]	58	Identify the business requirements that keep changing which refer to the 'needs' rather than their 'wants'.	*Identify Customer Needs*							
[PdC03]		"Requirements Management Plan" gathers the "User Needs" and the "Business Application Domain" which are later sent to the "Requirement Specification".	*Customer Needs + Business Application Domain*			**Requirements Management Plan**				
[PdC03]	33	A "Requirements Management Plan" with early general guidelines is defined in terms of business rules, procedures and processes: • Multidimensional Requirements Focus: Granularity in data marts, constraints for multidimensional analysis. • Source Integration Premises: Defining rules for data exchange between systems, periodicity, data loading priorities and responsibilities	*Granularity in data marts, constraints for MD analysis, and rules for data exchange and load.*			**Requirements Management Plan**				
[PdC03]		"Requirements Management Planning": phase wrapped clients into the novel experience of discussing the data warehouse inner aspects, objectives and construction guidelines together with the technical team.	*Requirements Management*							

Figure 6.3: Literature Analysis of DSS Requirements Engineering

			Planning Phase								
[Kim98]	59	One of the two basic techniques used to conduct user requirement analysis: facilitated sessions.				✓	Facilitated Sessions				
[GR09]	60	Facilitated sessions involve large heterogeneous groups good for brainstorming.				✓	Facilitated Sessions + Brainstorming				
[Kim98]	61	One of the two basic techniques used to conduct user requirement analysis: interviews				✓	Interviews				
[GR09]	66	The preliminary activities involved in interviews are: Pre-interview research, interviewee selection, question development, interview scheduling, and preparation. Types of questions: Open-ended, closed and evidential.				✓	Interviews				
[Kim98]	62	During the interview ask: "what do you do (and why)"; don't ask "what do you want" in the data warehouse.	*Identify Customer Needs*			✓	Specialized Inteviews				
[PdC03]	63	Specialized interviews with straight questions tailored to data warehouse issues like granularity and multidimensionality. (Using Kimball's approach)				✓	Specialized Inteviews				
[Kim98]	64	Begin by talking to the business users about their jobs, their objectives, and their challenges. Then IT interviews.				✓	Specialized Inteviews				
[Kim98]	65	Do get face-to-face with business representatives or voice-to-voice. Don't rely on non-interactive surveys or questionnaires.				✓	Face-to-face Interviews				
[Kim98]	67	Proposes techniques to deal interviews with different users: Abused, Overbooked, Comatose, Overzealous, Know-it-all, Clueless, and Non-existent user.				✓	Interviews by User				
[Kim98]	68	Data audit interviews help to understand whether the data is complete, reliable and is there to support business needs.				✓	Data Audit Interviews				✓
[GR09]	69	Interviews are conducted either with single users or small, homogeneous groups, offering good results in detailed list of specifications.				✓	Interviews				
[Kim98]	70	During the interview pull out a list of data elements to determine what's needed.				✓	Elicitation Checklist				
[PdC03]	71	Making data warehouses requirements workshops to: encourage consensus over application's multidimensional scope; promote fast agreement about the course of actions for data mart delivering; and capture major functionalities and operational constraints.				✓	DW Requirments Workshop	*Promote fast agreement about the course of actions for system's delivery*			
[IBM09]	72	BI Elicitation Techniques:				✓	-Brainstorm -Analysis Interviews -Group Interface -Observations -Prototyping Reverse -Engineering Surveys -Workshops	✓			
[RB12]	73	Building prototypes helps users understand their requirements and the system's capabilities	✓	✓	✓	✓	Prototypes				
[GR09]	74	The main 'sources' from which to draw requirements are the future data mart end-users. Also called business users.	✓				*Elicitation Source: Business Users*				
[IBM09]	75	IS/IT start with Technical Requirements, such as: Information, tools and Software. BI start with Business Requirements: Alignment and Governance: People, Processes and Culture. Fully engage business stakeholders	✓	✓			*Start w/Business Requirements*				
[SWW11]	76	Elicitation process should not only address employees with operational tasks, but also include those with middle management responsibilities and decision-making competence, such as department or team leaders.	✓	✓			*Include middle management & decision-making competence*				
[GR09]	77	As far as the technical know-how is concerned, information system administrators and/or data processing staff will be a point of reference for designers.		✓	✓		*Elicitation Source: Data Processing Staff*				
[SWW11]	78	People responsible for the source system are asked about technical constraints in meeting the information needs.			✓		*Elicit Technical Constraints*				
[BDGM08]	79	In order to specify each DM-BI project requirement context, it is necessary to identify requirement limits: • Related to data (to access sources of information, and data quality) • Related to human and technical resources (size of the data sources, hardware and software limitations, human resources) • Related to the project security	✓		✓	✓	*Data, technical resources and security limits*				
[PdC03]	80	Involves developers using pre-existing operational documentation to define data extraction and integration procedures.			✓	*Use pre-existing operational documentation*	*Data Extraction & Integration Procedures*				
[PdC03]	81	In a data warehouse environment, data is collected from several different sources, inside and outside the enterprise environment. Therefore specifying procedures and requirements for data collection must be done.			✓	✓	*Data Collection Procedures*				
[SWW11]	82	In the elicitation of information needs, future target requirements are taken into account.	✓				*Elicit Future Target Requirements*				
[PdC03]	83	Analyzing user requirements implies identification of facts by perceiving the metrics behind user demands. And determine the extent to which a metric can be operated (summarized, counted, drawn an average) along dimensional hierarchies.		*Determine the extent*		✓	*Identify Facts, metric operations, and*				

Figure 6.4: Literature Analysis of DSS Requirements Engineering

				metrics can be operated			*dimension hierarchies.*			
[SWW11]	84	Mutually dependent and overlapping information needs are identified during the negotiation.						*Identify Overlapping Requirements*		
[PdC03]	85	Project Objectives: clear objectives and project restrictions must be agreed among all participants (i.e, aggregated information).	*Project Objectives*					*Agreement among all stakeholders*		
[GR09]	86	After investigating the available information resources in the data source analysis and reconciliation phase, it is now time to approach the end users again to get more accurate information.	*Investigate available sources*					*Negotiation Stategy*		
[PdC03]	87	Use Cases can be used to specify scenarios textual or graphic (i.e. UML). Standard Use Case model in UML can be used for the main data warehouse functionalities, namely extracting, transforming and accessing data. "Querying" and "Loading" processes can be designed as single use cases using parameterized entries.		**Use Cases**		✓				
[PdC03]	88	Use Cases Specification: Detail the procedures required to implement all functionalities describing the possible sequences. While it also enables the reuse of common behavior shared among different Data Mart scenarios.		**Use Cases**		**Use Cases Specification**			*Reuse of common behavior scenarios*	
[GOIM+00]		A business use case is defined for each business process and a business use case diagram is used to show the context and the boundary of the enterprise. While System Use Cases document the system's functions.		**Business / System Use Case**		✓				
[GOIM+00]		Scenarios are created to show the behavioral aspect of the role collaboration. In UML sequence diagrams, the objects denote the instances of the roles that participate in the interaction.		**Sequence Diagram**		✓				
[GOIM+00]		Process diagrams show the workflow performed to obtain some goal of the organization, indicating the roles that are in charge of each activity, and the data required and produced by each activity. These can be easily adapted to UML, by using activity diagrams with swim lanes.		**Process Diagram**		✓				
[GOIM+00]		Workflows are represented by means of activity diagrams, use cases. These are elicited and structured starting from the activities of each process, while the concepts of the conceptual model are obtained from the data that flow between activities.		**Workflow Models**		✓				
[IBM09]	89	BI Analysis Practices:		**-Use Case User Stories -Process Models -Domain Models - Workflow Models**		✓				
[PdC03]	93	Data Warehouse-Extended NFR Framework or DW-ENF applies to selecting indexing strategies, disk space optimized methods, loading schema and other critical parts of a data warehouse design. The DW-ENF non-functional requirements list also works as a checklist to guide the specification of system quality requirements.			**DW - ENF Framework**	*Checklist for Quality Requirements*			✓	
[PdC03]	94	Data warehouses have the capability of turning data into information. Such capability is supported by OLAP technology, which provides the final users with configurable views of data from different angles and aggregation levels.			*Configurable Views*	✓				
[PdC03]	49	There is a lack of multidimensional restrictions and quality constraints.			*MD restrictions and quality constraints*	✓				

Figure 6.5: Literature Analysis of DSS Requirements Engineering

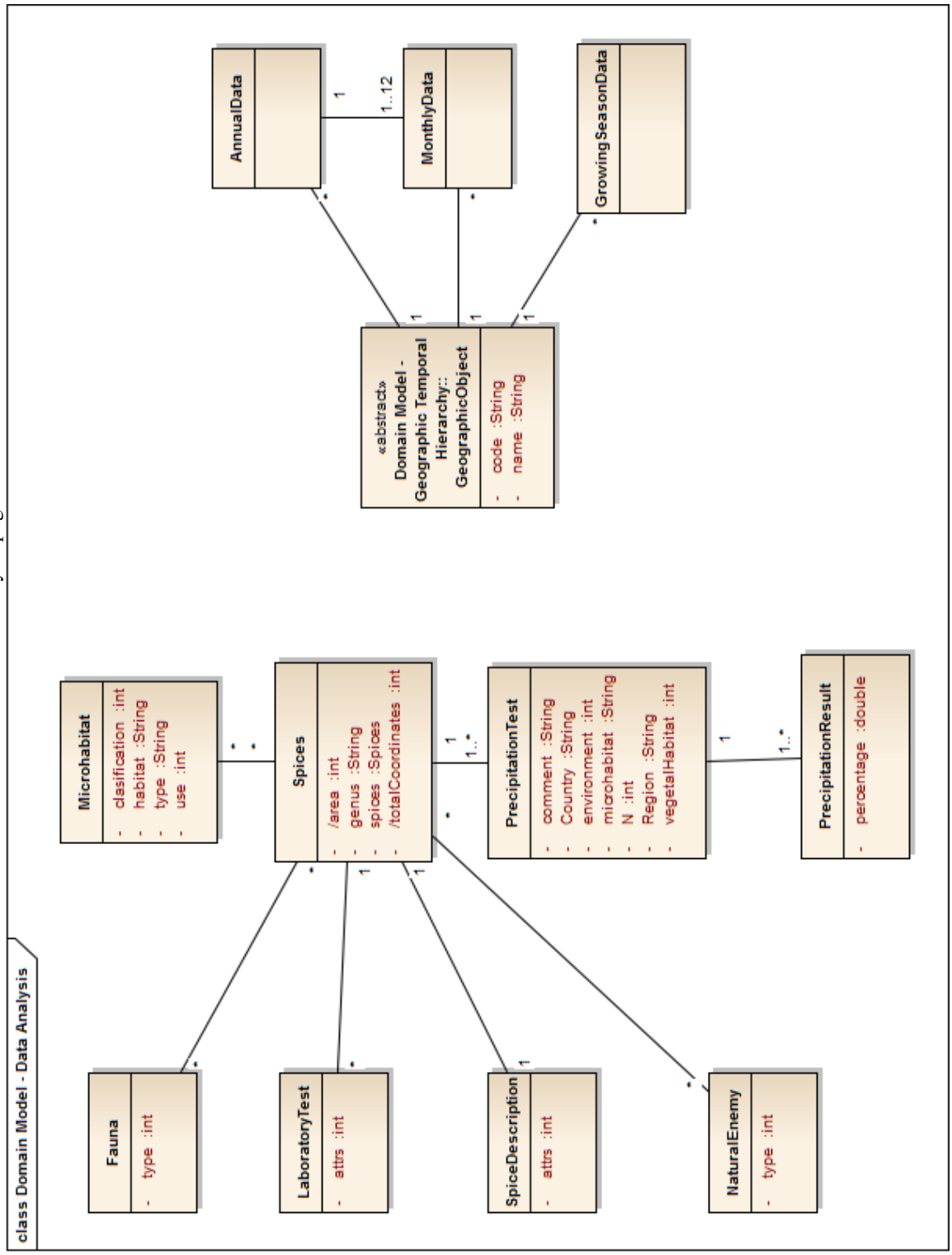


Figure 6.6: Vocabulary Representation - Domain Model/Data Analysis. Reprinted from (VN14)

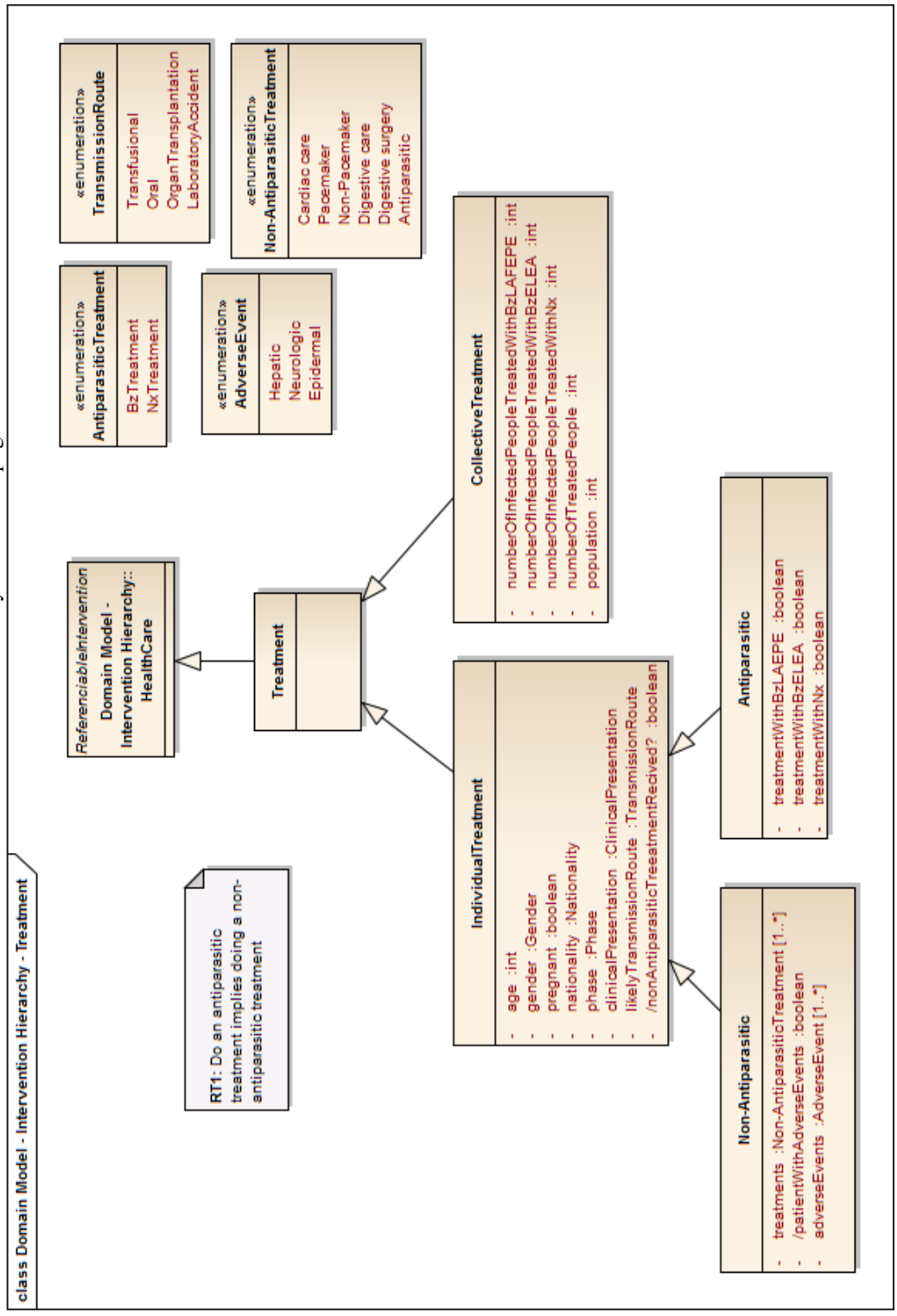


Figure 6.7: Vocabulary Representation - Domain Model - Intervention Hierarchy/Treatment. Reprinted from (VN14)

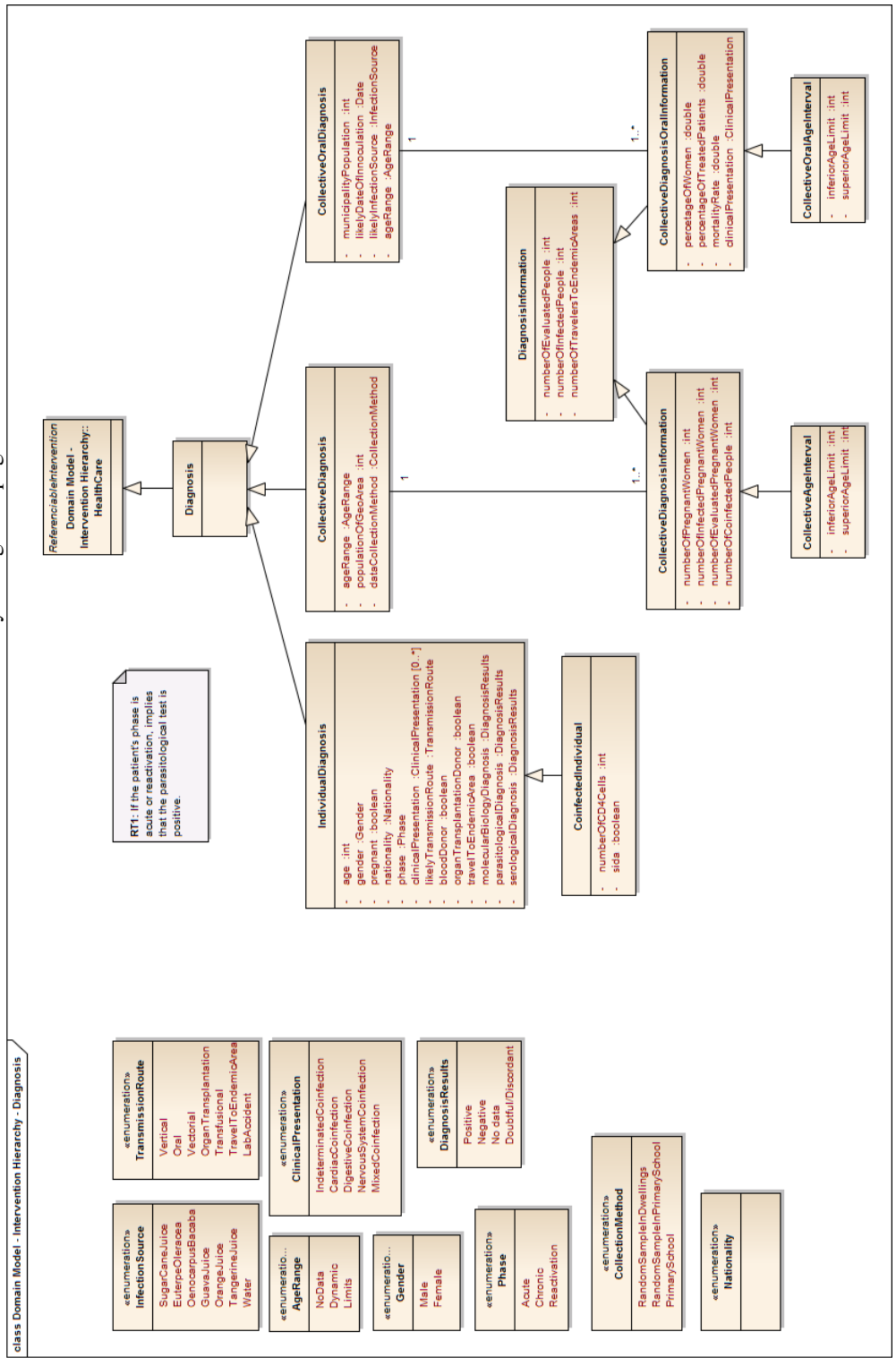


Figure 6.8: Vocabulary Representation - Domain Model - Intervention Hierarchy/Diagnosis. Reprinted from (VN14)

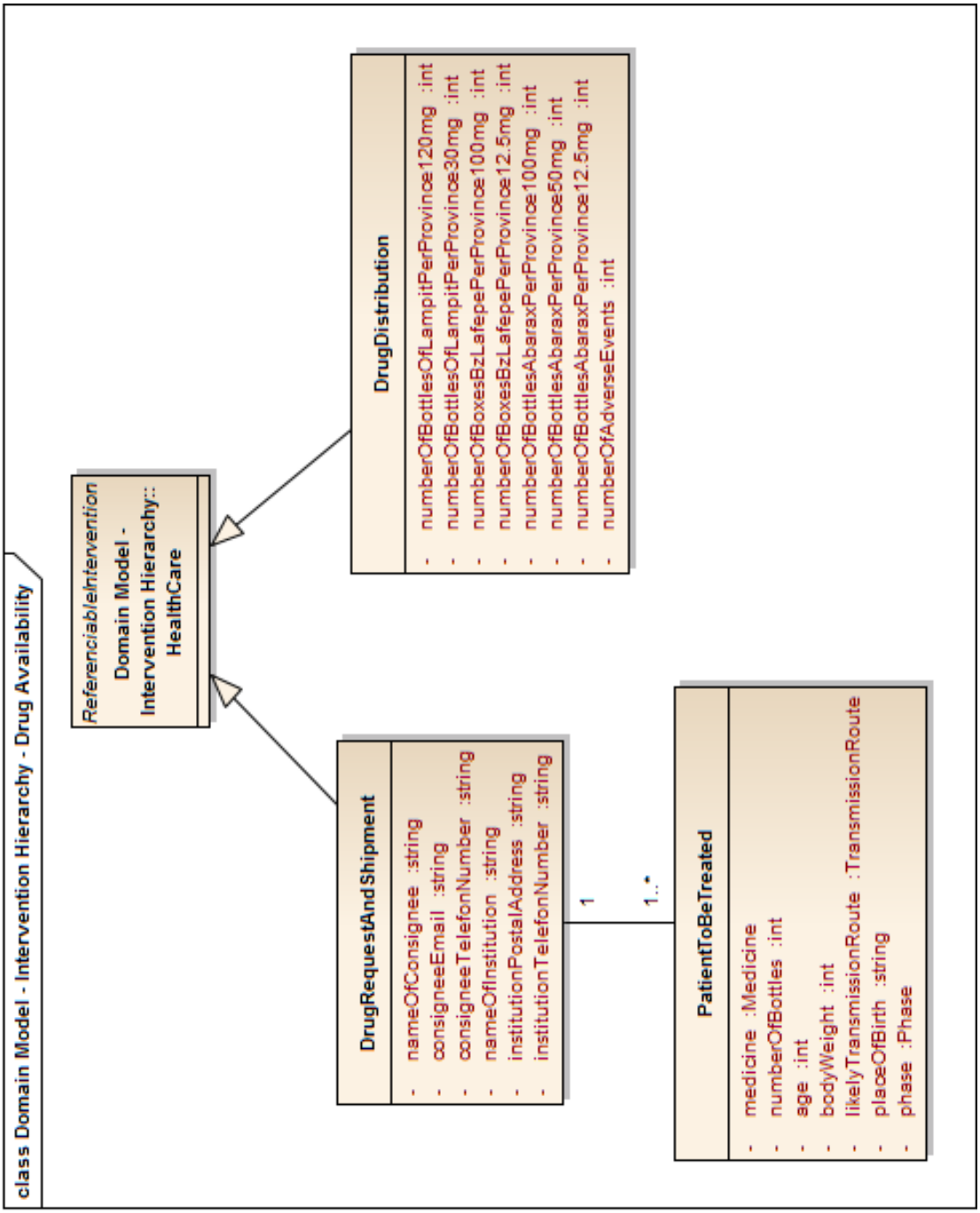


Figure 6.9: Vocabulary Representation - Domain Model - Intervention Hierarchy/Drug Availability. Reprinted from (VN14)

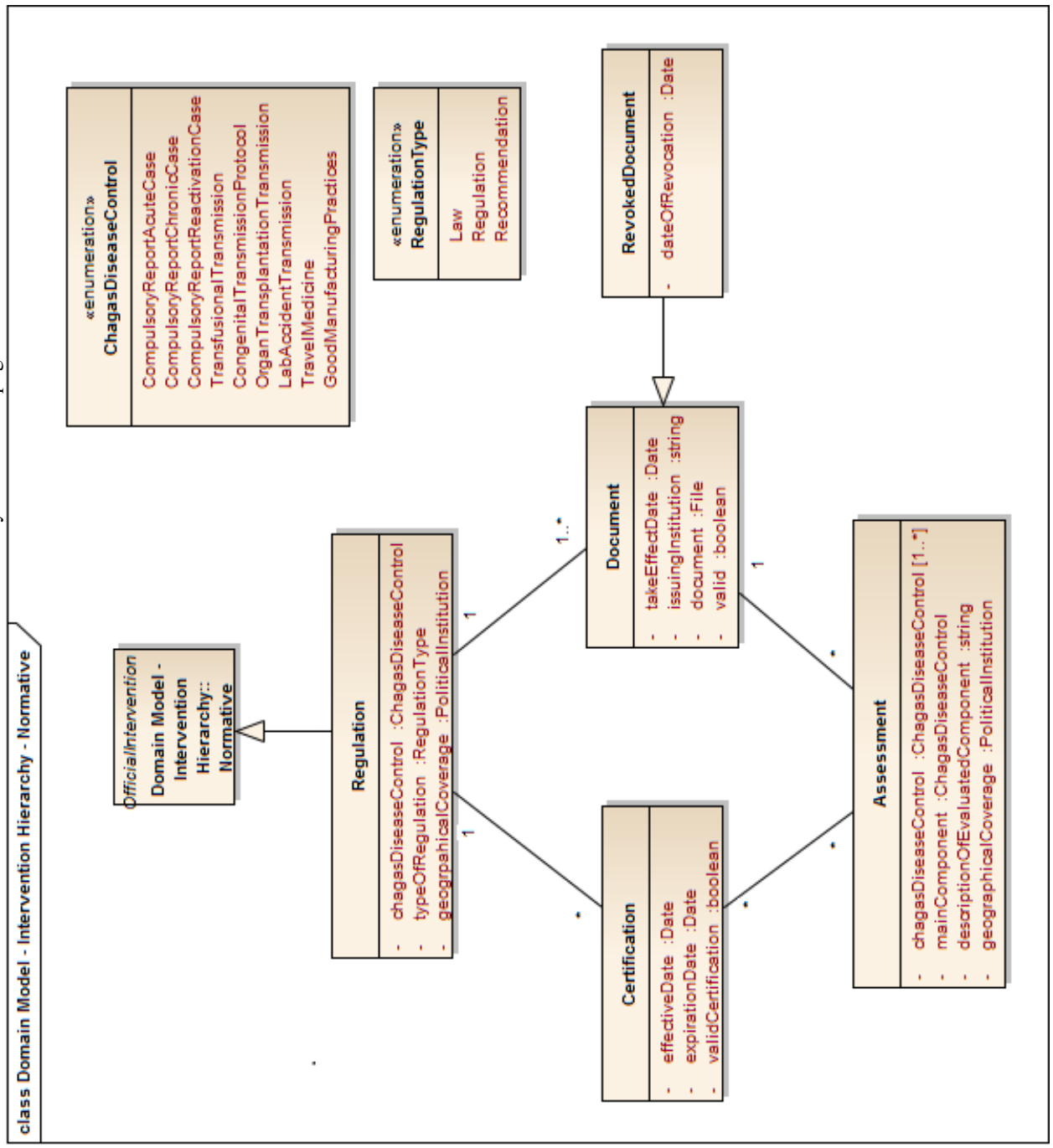


Figure 6.10: Vocabulary Representation - Domain Model - Intervention Hierarchy/Normative. Reprinted from (VN14)

Model - Intervention Hierarchy - Systemic.png

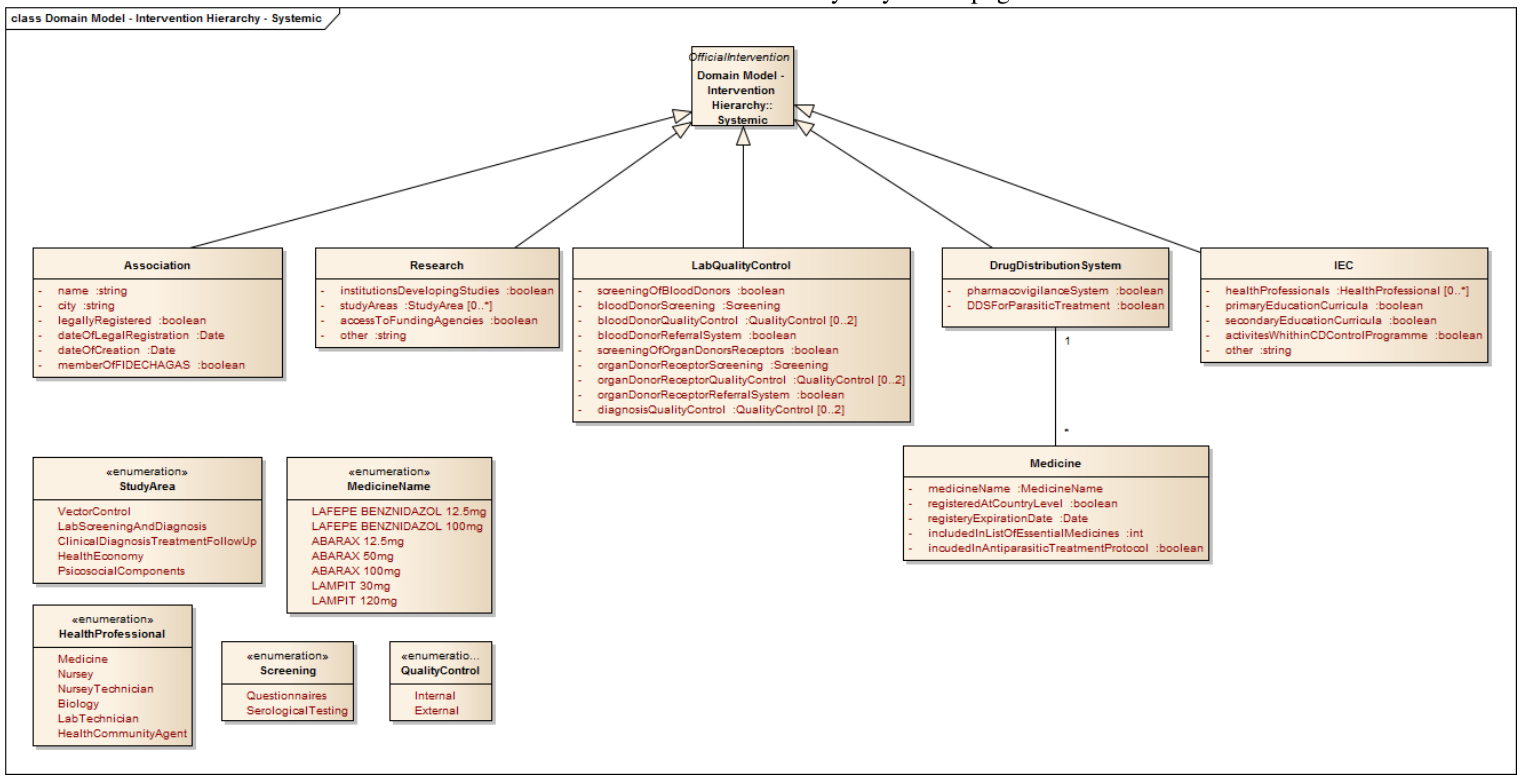


Figure 6.11: Vocabulary Representation - Domain Model - Intervention Hierarchy/Systematic. Reprinted from (VN14)