SysPRE
Systematized process for requirements engineering in knowledge discovery projects

MASTER DISSERTATION

Ana Beatriz da Palma Rodrigues Neto
MASTER IN INFORMATICS ENGINEERING

UNIVERSIDADE da MADEIRA
A Nossa Universidade
www.uma.pt

October | 2014

SysPRE

Systematized process for requirements engineering in knowledge discovery projects

MASTER DISSERTATION

Ana Beatriz da Palma Rodrigues Neto
MASTER IN INFORMATICS ENGINEERING

SUPERVISOR
David Sardinha Andrade de Aveiro

Acknowledgments

I would like to thank to my advisor David Aveiro who inspired my choice of thesis area when he taught the courses “Organizational Engineering” and “Decision Support Systems” and that supported and encouraged me during the whole process.

I would also like to thank all my other teachers in the Master in Computer Engineering, namely Prof. Evan Karapanos, Prof. Larry Constantine, Prof. Morgado Dias, Prof. Olga Lyra, Prof. Eduardo Teles, Prof. Lina Brito, Prof. Dan Boyarski, Prof. Robert Spencer and Prof. Pedro Campos for helping me grow as a Software Engineer.

A special thanks goes to Duarte Gouveia, my husband and fellow Master student for taking this journey with me and for his endless support and love.

Resumo

O domínio de *"Knowledge Discovery"* (KD) e *"Data Mining"* (DM) é cada vez mais importante numa altura em que cada vez mais dados são produzidos e o conhecimento é um dos bens mais preciosos.

Tendo explorado a teoria subjacente, os resultados da investigação em curso no mundo académico e as práticas da indústria no domínio do KD e DM, descobrimos que este é um domínio que ainda carece de alguma sistematização.

Por outro lado, esta sistematização existe em maior grau no domínio da Engenharia de Software e da Engenharia de Requisitos, provavelmente devido a estas serem áreas mais maduras.

Acreditamos que é possível facilitar a participação das partes interessadas a nível da engenharia de requisitos para projetos de KD sistematizando o processo de engenharia de requisitos para este tipo de projetos.

Com isso em mente e com base em todas as informações encontradas no levantamento do estado-da-arte, propomos o SysPRE (do inglês *Systematized Process for Requirements Engineering)* que é destinado especificamente para projetos de KD.

Começamos por propor uma descrição genérica para o processo de KD, com grande enfâse na engenharia de requisitos. Esta descrição é então usada como base para a aplicação da metodologia DEMO (do inglês *Design and Engineering Methodology for Organizations*), em que especificamos uma ontologia formal para este processo. A especificação ontológica do SysPRE obtida é uma base que pode ser usada, não só para que as empresas se tornem conscientes do seu próprio processo de KD e do processo de engenharia de requisitos nos projetos de KD, mas também para que possam, então, ser capazes de melhorar esses processos, nomeadamente em termos de taxa de sucesso.

Palavras-chave

Descoberta de Conhecimento, Mineração de Dados, Engenharia de Requisitos, DEMO

Abstract

The domain of Knowledge Discovery (KD) and Data Mining (DM) is of growing importance in a time where more and more data is produced and knowledge is one of the most precious assets.

Having explored both the existing underlying theory, the results of the ongoing research in academia and the industry practices in the domain of KD and DM, we have found that this is a domain that still lacks some systematization.

We also found that this systematization exists to a greater degree in the Software Engineering and Requirements Engineering domains, probably due to being more mature areas.

We believe that it is possible to improve and facilitate the participation of enterprise stakeholders in the requirements engineering for KD projects by systematizing requirements engineering process for such projects. This will, in turn, result in more projects that end successfully, that is, with satisfied stakeholders, including in terms of time and budget constraints.

With this in mind and based on all information found in the state-of-the art, we propose SysPRE - Systematized Process for Requirements Engineering in KD projects.

We begin by proposing an encompassing generic description of the KD process, where the main focus is on the Requirements Engineering activities. This description is then used as a base for the application of the Design and Engineering Methodology for Organizations (DEMO) so that we can specify a formal ontology for this process. The resulting SysPRE ontology can serve as a base that can be used not only to make enterprises become aware of their own KD process and requirements engineering process in the KD projects, but also to improve such processes in reality, namely in terms of success rate.

Keywords

Knowledge Discovery, Data mining, Requirements Engineering, DEMO

List of Figures

List of Acronyms

ACM - Association for Computing Machinery

ASD - Adaptive Software Development

ATD - Actor Transaction Diagram

BDUF - Big Design Up Front

CAP - Coordination-Actors-Production

CMM - Capability Maturity Model

CRISP-DM - CRoss-Industry Standard Process for Data Mining

DEMO – Design and Engineering Methodology for Organizations

DM - Data Mining

DMIE - Data Mining for Industrial Engineering

DMKD - Data Mining and Knowledge Discovery

FR - Functional Requirements

IT - Information Technology

KDCF - Knowledge and Discovery and Communication Framework

KD - Knowledge Discovery

KDD - Knowledge Discovery in Databases

KDLC - Knowledge Discovery Life Cycle

KDP - Knowledge Discovery Process

MLP-ANN - MultiLayer Perceptron - Artificial Neural Networks

NFR - Non-Functional Requirements

ODKD - Ontology Driven Knowledge Discovery

OFD - Object Fact Diagram

OpenUP - Open Unified Process

ORM - Object-Role Modeling

PCA - Principal Component Analysis

PIF - Performa-Informa-Forma

PSD – Process Structure Diagram

R-CMM - Requirements Capability Maturity Model

RAD - Rapid Application Development

RAMSYS - RApid collaborative data Mining SYStem

RE - Requirements Engineering

RUP - Rational Unified Process

SEMMA - Sample, Explore, Modify, Model, Assess

SIGKDD - Special Interest Group on Knowledge Discovery and Data Mining

SOM - Self-Organizing Map

SW_CMM - Software Capability Maturity Model

SysPRE - Systematized Process for Requirements Engineering (in KD projects)

TB - Technique Box(es)

TRT - Transaction Result Table

WOSL - World Ontology Specification Language

XP - Extreme Programming

List of Tables

List of Appendixes

Index

1 Introduction

"The greatest obstacle to discovery is not ignorance - it is the illusion of knowledge."

Daniel J. Boorstin

1.1 Motivation and research question

Software development has been around for several decades now and discussion on its failures and successes has been strong.

It all started with the Standish Group's Chaos Report of 1994 [1] that stated that projects that did not meet customer satisfaction and/or went over time or budget in a significant way corresponded to 53 percent. It was a bit shocking to see a figure that amounted for over half of the projects and a discussion about a software crisis was started.

This report, however, was since then criticized for lack of peer review, for not having a complete description of the study design or of the project selecting criteria, and for defining successful and failed projects in a way that may bias the study [2], [3], [4], [5].

Over 20 years later, the debate is still on, but there seems to be an agreement on the failure rate of software development projects having dropped [5], [6], [7]. Although the values do not coincide, they show a decrease tendency that we believe is significant if you take into account that projects are increasingly complex.

We also believe that one of the areas of software development that has helped this increased success in software projects is Requirements Engineering (RE), following previous research such as [8] and [9]. Furthermore, according to [6], one of the three main reasons for the positive development is that the communication of requirements has much improved. Wateridge makes an even stronger statement that "Meets user requirements" is the most important success criteria for both users (96%) and project managers (81%) [10].

Knowledge discovery and data mining are much more recent areas than software development and we believe that they are less mature fields. For instance, if we consider process model development to be a sign of maturity, we can see that the first process model for this area dates back to 1996 [11], while in software development, the well-known Waterfall model goes back to 1970 [12].

Nonetheless, it is indisputable that knowledge discovery and data mining are of growing importance in a time where more and more data is produced.

Data production numbers are, in fact, staggering:

- 144.000 hours of video are uploaded to Youtube per day [13] (which amounts to an average of 100 per hour)
- 182.900.000.000 emails are sent per day [14] (which amounts to an average of 35 emails per person over 15)
- 1.000.000.000 pieces of content are shared on Facebook per day [15] (which amounts to an average of 11500 per second)

This results in massive amounts of data. Facebook has one of the largest data warehouses in the world, storing more than 300 petabytes [16].

With such a large production of data and in a time when knowledge is one of the most precious assets, it is no wonder that knowledge discovery and data mining are of increasing importance.

We believe that the road for knowledge discovery and data mining projects is to increase systematization, as the area becomes more main stream.

This seems important because the trends in this area, currently, are to have larger projects (with larger amounts of data involved) and, at the same time, to have the people involved in those same projects with lower technical skills and very little time to experiment with different approaches [17].

Within the knowledge discovery projects, we believe the area of requirements engineering is the one that one can reap more benefits thanks to a higher level of systematization.

Firstly, because requirements engineering is particularly neglected in this type of projects. Some authors even argue that this type of projects should be based on the available data and not on stakeholders' requirements [18].

Secondly, because, being a less mature field, less systematization efforts have been made so far and we believe that when they occur, the participation of enterprise stakeholders will be improved and facilitated and the area will follow software engineering in general, that has, as we have seen above improved in terms of customer satisfaction and time and budget compliance.

For these reasons, the research question that we investigated during this Master thesis was **"How can systematization be brought into Knowledge Discovery projects, in general, and into their Requirement Engineering phase, in particular, aiming at improvements in their success rate?"**.

We started by analysing the Knowledge Discovery process through a systematic review of the state-of-the-art in academia and industry regarding knowledge discovery and data mining process models. To conclude this review we compared the main process models found.

We then analysed the Requirements engineering area in a similar way.

Focusing on requirements engineering for KD projects was the next step. We found that requirements engineering for KD is different. That is why we next claim that a requirements engineering for KD process model is needed and propose SysPRE, a Systematized Process for Requirements Engineering designed specifically for KD projects.

The proposed model, SysPRE, began from an initial textual description which was then formally specified as a DEMO ontology. This formal specification was instantiated in two case studies so that trivial and non-trivial errors could be identified and the necessary adjustments made.

We believe SysPRE synthesises the knowledge obtained for the state-of-the-art reviews in a way that can be helpful for enterprises and other organizations with KD projects both for novice and expert users. We hope this systematization can bring improvements to the success rate of such projects.

1.2 Thesis Structure

The remaining of this Master thesis is organized as follows.

In chapter 2 the Knowledge Discovery process is analysed by doing a systematic review of the state-of-the-art in academia and industry regarding knowledge discovery and data mining process models and comparing them.

In chapter 3 the Requirements engineering is analysed in a similar way: an historical overview of the area is presented and a systematic review of the state-of-the-art of the requirements engineering process models is shown.

In chapter 4, the cross between the areas presented in the two previous chapters is analysed, that is requirements engineering for KD projects. We begin by discussing why

requirements engineering for KD is different and argue for the need of a requirements engineering for KD process model.

In chapter 5 we propose SysPRE, a Systematized Process for Requirements Engineering designed specifically for KD projects. We begin by a textual description and then as a DEMO ontology specification.

In chapter 6 we present the case studies used for validation, followed by a discussion regarding the proposed model advantages over the ones presented in the state-of-the art review in chapter 7.

We conclude the thesis by presenting the contributions, limitations and future work in chapter 8 and the references in chapter 9.

Finally, the appendices are attached in chapter 10. In Appendix I we present the requirements elicitation techniques found in our literature review. In Appendix II we present a schematic representation of the historic evolution of Intelligent Digital Assistants whose goal is to automate part of the KD process. In Appendix III, a brief summary of DEMO is presented. In Appendix IV, the final case study instantiation diagram is shown.

2 Knowledge discovery

"There is no sense in being precise about something, when you don't even know what you are talking about."

John Von Neumann

2.1 Definitions

Historically, the term Data Mining (DM) was originally more popular with database researchers and statisticians, while the term Knowledge Discovery in Databases (KDD) was popular in the Artificial Intelligence and Machine Learning research community [11] [19]. Eventually, the term Data Mining became prevalent, especially with business applications and business users. This made the term Data Mining more popular with the general public. [19]

Today, both terms are used interchangeably by the Information Technology (IT) community and the general public. Using both terms together has also become popular. For instance, one of the most popular conferences in this area is the "International Conference on Knowledge discovery and data mining" promoted by the Association for Computing Machinery's Special Interest Group on Knowledge Discovery and Data Mining (ACM SIGKDD).

For most researchers, however, data mining is one of the steps in the knowledge discovery process, as we will see in detail in section 2.3. We will use the definition of [20]: "KDD refers to the overall process of discovering useful knowledge from data, and data mining refers to a particular step in this process. Data mining is the application of specific algorithms for extracting patterns from data.".

As for KDD, the most popular definition says that it is "the nontrivial process of identifying valid, novel, potentially useful, and ultimately understandable patterns in data" [21]. These patterns can be used for explaining past events or predicting future events or trends.

Other authors emphasize that KDD "is a creative process which requires a number of different skills and knowledge" [17]. Other authors still note that KDD can be seen, not only as a process, but also as an end-product and, in fact, as a very coveted one: "Knowledge Discovery is the most desirable end-product of computing." [22].

2.2 Need for a KD process model

The need for a process model stems from the fact that data mining is non-trivial. In 2006, Bernstein et al. refered that "there are many possible choices for each stage, and only some combinations are valid. Because of the large space and nontrivial interactions, both novices and data mining specialists need assistance" [23].

Still the need for a process model goes back to 1989, when it was first discussed during the IJCAI workshop on Knowledge Discovery in Databases (KDD) [24]. This was the original workshop that started the series of KDD workshops that, from 1995 onwards, grew into KDD conferences. Still, only in 1996 the first model was formally proposed (as we will discuss in section 2.3.1).

It is important to note that we will consider a process model as the set of framework activities and tasks necessary to get the job done, including inputs and outputs in every task [25]. We will also refer to methodologies, considering that a methodology is a process model instance, in which not only tasks, inputs and outputs are specified but also the way in which the tasks must be carried out. To sum up, process models define what to do; methodologies define how to do it.

The need for a process model has resulted in the proposal of a multitude of models, as we will see in section 2.3. However, as noted in [26] [17], there is the need for a standardized process model because:

1. It can help ensure that the end product is useful for the end user. Although this is a basic need, it is often neglected and it has even made some researchers insist on the term Actionable Knowledge Discovery (AKD) [27][28].
2. It can be presented to decision-makers who may find difficult to understand the need, value and mechanics behind KDD.
3. It can provide a solid justification to the significant project management effort needed for knowledge discovery projects.
4. It can help in the dissemination of knowledge and experience within the organization.
5. As KDD becomes a more mature field it is an expected evolution.
6. It can help make projects faster, cheaper, more reliable and more manageable.

2.3 Existing KD process models

2.3.1 Original KDD model

The first model for data mining was proposed in 1996 [11]. The KDD model consists of nine iterative steps:

1. Learning the application domain, which includes not only understanding the application domain and any relevant prior knowledge but also identifying the goal of the process
2. Creating a target dataset
3. Data cleaning and pre-processing
4. Data reduction and projection
5. Function of data mining selection (e.g., summarization, classification, regression, clustering)
6. Data mining algorithm(s) selection and specification of any relevant parameters
7. Data mining, which means the actual search for patterns
8. Interpretation of the results
9. Using discovered knowledge, which could be done in many ways, such as incorporating the knowledge into another system or simply generating a report of the findings.

2.3.2 Other models 1996 to 1999

From the original KDD model, other models and methodologies developed over the years:

- The model by Ganesh et al. [29] was proposed in 1996 almost as a side-thought in a paper that is mostly concerned with making data mining more visual. Six steps were considered: (1) scrub, verify and summarize data, (2) selection of the training data sample, (3) model derivation, (4) verify and evaluate, (5) selection of most interesting models and finally (6) model usage, population shift monitoring and incremental learning.

- The model by Adriaans and Zantinge [30], proposed in 1996, has six steps: data selection, cleaning, enrichment, coding, data mining, and reporting.

- The model by Brachman and Anand [31] proposed in 1996, emphasized the interactive nature of the process, taking the viewpoint of the data miner and of the decisions that he had to take. For that reason, is often referred to as a human-centered model. It has six steps: task discovery, data discovery, data cleaning, model development, data analysis and output generation.

- The model by Berry and Linoff [32], proposed in 1997, has four steps: identifying the problem, analyzing the problem, taking action, and measuring the outcome.

- The model by Collier et al. [33], proposed in 1998, extends the original KDD model with the goals of framing the questions in the right way (so that one does not "blindly set the algorithms loose on the data" [33]) and of getting actionable results (so that they can be used for business decisions). The authors also emphasized the need for iteration. This resulted in eight steps: define the objectives, select the relevant business data, data quality analysis, clean and transform data, data mining, acquire knowledge, evaluate results, deploy results or reiterate.

- The model by Cabena et al. [34], proposed in 1998, has five steps: business objectives determination, data preparation, data mining, domain knowledge elicitation, and assimilation of knowledge.

- The Knowledge Discovery Life Cycle (KDLC) model by Lee and Kerschberg [35], proposed also in 1998, again extends the original KDD model. This model has six steps: plan for learning, generate and test hypothesis, discover knowledge, determine knowledge relevancy, evolve knowledge/data and critique by a panel of experts.

- The model by Feelders et al. [36], proposed in 1998, has six steps: problem formulation, identification of background knowledge, selection of data, pre-processing the data, analysis and interpretation and, finally, use of results. The need for ease of interpretation of the model is stressed.

- The model by Buchner et al. [37], proposed in 1999 for use with internet data after the previous work by the authors [38], has eight steps: human resource identification, problem specification, data prospecting, domain knowledge elicitation, methodology identification, data preprocessing, pattern discovery, and knowledge post-processing.

2.3.3 CRISP-DM

CRISP-DM was created in 1997 by a group of organizations involved in data mining (NCR, SPSS, Daimler-Chrysler and OHRA) [41] [39]. The first version was published in August 2000 [39]. Between 2006 and 2008 there were efforts to launch a second version of CRISP-DM, which was referred to as CRISP-DM 2.0, but no result was ever published.

In 2007, an industry poll [40] revealed that the most common used methodologies were CRISP-DM (CRoss Industry Standard Process for Data Mining) and SEMMA (Sample, Explore, Modify, Model, Assess). We will discusse CRISP_DM in the current section and SEMMA in section 2.3.4. The original KDD process described in section 2.3.1 was still the fourth most used.

To the question "What main methodology are you using for data mining?" the responses were the ones shown in Table 1.

CRISP-DM	42%
My own	19%
SEMMA	13%
KDD Process	7%
My organizations'	5%
Domain-specific methodology	5%
Other methodology, not domain-specific	4%
None	5%

Table 1 - Methodology used for data mining [40]

Although CRISP-DM was the *de facto* standard in 2007 [40] and probably still is the most popular methodology to date, it is important to note that the website that supported the project (http://www.crisp-dm.org/) has been offline since, at least, 2011.

Being industry-oriented, it is no surprise that "The CRISP-DM process model aims to make large data mining projects less costly, more reliable, more repeatable, more manageable, and faster." [39].

The CRISP-DM model life cycle consists of six iterative steps:

1. **Business understanding**

   Initial phase that focuses on understanding the project objectives and requirements from a business perspective, defining a data mining problem, and a preliminary project plan

2. **Data understanding**

   Closely linked to the Business Understanding step, starts with the initial data collection and proceeds with activities to get familiar with the data, identify data quality problems and discover first insights into the data

3. **Data preparation**

   Includes all transformations needed on the initial data to create the final dataset that will be fed into the modeling tool

4. **Modeling**

   Modeling techniques are selected and applied, and their parameters are adjusted

5. **Evaluation**

   The model or models built previously are evaluated, and the steps executed to construct them are reviewed, to be certain it properly achieves the business objectives

6. **Deploying**

   This can vary from the simple generation a one-time report to the more complex option of creating a repeatable data mining process for end-users, depending on the initial requirements

CRISP-DM is very detailed and task-oriented, with the goal of helping in:

1. Planning
2. Documentation
3. Communication
    a. Within the project team
    b. Outside the project team (both to clients or sponsors)
4. Justification of how much effort was spent in each task
5. Dissemination of knowledge
6. Making sure nothing was forgotten or overlooked (through to intensive use of checklists)

Some authors argue one of the strong point is the dissemination of knowledge [43]. One of the features that favors this is that CRISP-DM explicitly includes revising the project in the end so that the acquired experience can be documented and later used. The fact is that CRISP-DM is very detailed, with a total of 288 activities proposed as part of the full process.

2.3.4 SEMMA

Unlike CRISP-DM, SEMMA was created to be used in a specific application, SAS Enterprise Miner [42].

The acronym SEMMA stands for sample, explore, modify, model, assess, which are basically the five iterative steps proposed:

1. Sample, which consists of extracting sample data (optional step)
2. Explore, which means the exploring the data or the sample data in order to be able to simplify the model
3. Modify, which can include any cleaning, preprocessing, reductions or projections deemed necessary
4. Model, which is the actual search for patterns
5. Assess, which is the evaluation and interpretation of the results

In fact, despite being referred to as a methodology by both industry practitioners [40] and academia members [44], its creators now refuse the term: "SEMMA is not a data mining methodology but rather a logical organisation of the functional tool set of SAS Enterprise Miner for carrying out the core tasks of data mining" [42]. The truth is that SEMMA is tied to the SAS Enterprise Miner tool and therefore overlooks any steps that are not related to the tool, namely any business understanding tasks.

2.3.5 Catalyst

In 2003, the Catalyst methodology was proposed [45]. This methodology has two parts: business modelling and data mining. For each part, a detailed step-by-step methodology is proposed. Originally it was proposed both in printed form and online, and both formats followed a hyperlink structure. However, the website for it (http://www.modelandmine.com) ceased to be available in 2009.

Considering both parts of the methodology as a whole, we can say that it has six steps:

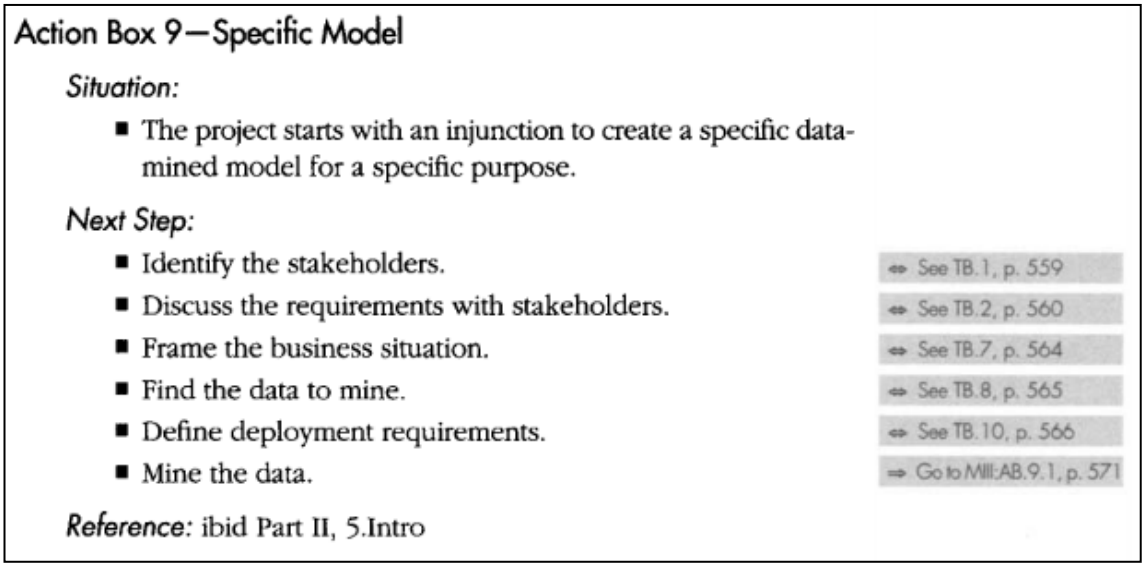
1. Business modelling
2. Data preparation
3. Tool selection
4. Mining
5. Refining
6. Deploying

What makes this methodology interesting is the level of detail that is includes in each step. It is very focused on what needs to be done and how it can be done. This is organized in what the author calls “boxes”. There are four types of “boxes”: Action Boxes, Discovery Boxes, Technique Boxes, and Example Boxes.

For instance, for the starting point (that is, the different business circumstances that prompt the modelling) it considers five different possible options:

1. Data - "Explore this dataset and find interesting and useful relationships."
2. Opportunity - "Here is a business opportunity we need to address - see what data mining can do to help."
3. Prospecting -"What can data mining do for us?"
4. Defined - "Use data mining to build a specific model"
5. Strategic - "In this strategic situation, can data mining help us to find out what's happening and what our options are?"

If you happen to be in situation described above as “Defined”, then you are directed to the Action Box 9, shown in Figure 1.


Action Box 9—Specific Model

Situation:

- The project starts with an injunction to create a specific data-mined model for a specific purpose.

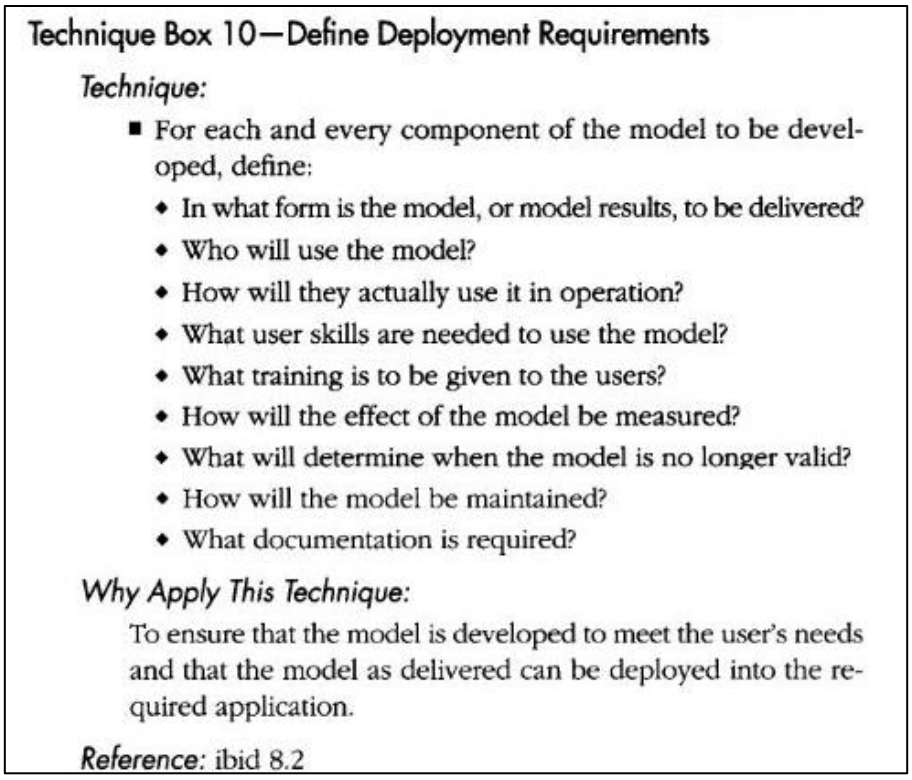
Next Step:

- Identify the stakeholders. ⇔ See TB.1, p. 559
- Discuss the requirements with stakeholders. ⇔ See TB.2, p. 560
- Frame the business situation. ⇔ See TB.7, p. 564
- Find the data to mine. ⇔ See TB.8, p. 565
- Define deployment requirements. ⇔ See TB.10, p. 566
- Mine the data. ⇒ Go to MII:AB.9.1, p. 571

Reference: ibid Part II, 5.Intro

Figure 1 - Example of a Catalyst process model Action Box [45]

You are guided through a list of steps in each Action Box. In the example of Figure 1, these steps are all Technique Boxes (TB). An example of these TB is the one shown in Figure 2.



Technique Box 10—Define Deployment Requirements

Technique:

- For each and every component of the model to be developed, define:
  - In what form is the model, or model results, to be delivered?
  - Who will use the model?
  - How will they actually use it in operation?
  - What user skills are needed to use the model?
  - What training is to be given to the users?
  - How will the effect of the model be measured?
  - What will determine when the model is no longer valid?
  - How will the model be maintained?
  - What documentation is required?

Why Apply This Technique:

To ensure that the model is developed to meet the user's needs and that the model as delivered can be deployed into the required application.

Reference: ibid 8.2

Figure 2 - Example of a Catalyst process model Technique Box [45]

2.3.6 Other models from 2000 onwards

After CRISP-DM was proposed, several variations were proposed over the years [22] [46]. Other models were also proposed:

- The model by Kurgan et al. [47], proposed in 2001, has six steps: understanding the problem domain, understanding the data, preparing the data, data mining, evaluation of the discovered knowledge, and using the discovered knowledge. This model was proposed with the goal of adapting the CRISP-DM model, discussed in section 2.3.3, to the needs of academic research community.

- The model by Han and Kamber [48], proposed in 2001, has seven steps: learning the application domain, creating a target data set, data cleaning and preprocessing, data reduction and transformation, choosing functions of DM, choosing the mining algorithm(s), data mining, pattern evaluation and knowledge presentation, and use of discovered knowledge.

- The model by Edelstein [49], proposed in 2001, has five steps: identifying the problem, preparing the data, building the model, using the model, and monitoring the model.

- The Rapid Collaborative Data Mining System (RAMSYS) model [50], proposed in 2001, is an extension of CRISP-DM, discussed in section 2.3.3, and has five steps: business understanding, data understanding, data preparation, modeling and evaluation. The first paper that proposed this model [50] focused primarily on the data preparation and modeling steps, with a later paper [51] discussing the evaluation step.

- The Data Mining for Industrial Engineering (DMIE) model proposed in a master thesis by Solarte in 2002 [52] was also based in CRISP-DM and also proposes five steps, but they are quite different: analyze the organization, structure the work, develop the data model, implement the model and establish on-going support. The last step is the main difference, as it includes support and maintenance which are not considered in other models, and which include data backups, data maintenance, software updates and data mining model updates when necessary.

- The model by Klosgen and Zytkow [53], proposed in 2002, has seven steps: definition and analysis of business problems, understanding and preparation of data, setup of the search for knowledge, search for knowledge, knowledge refinement, application of knowledge in solving the business problems, and deployment and practical evaluation of the solutions.

- The model by Haglin et al.[54], proposed in 2005, has seven steps: goal identification, target data creation, data preprocessing, data transformation, data mining, evaluation and interpretation, and take action steps. This model was conceived to be used by scientists, as the authors argue that "KDD is really only a very useful tool to aid scientists. It is up to the scientist to formulate goals, present data appropriate to those goals, and draw scientific conclusions from the patterns identified by the data mining step".

- The Data Mining and Knowledge Discovery (DMKD) model by Cios and Kurgan [55], proposed in 2005, is based on CRISP-DM and considers six steps as well: understanding the problem domain, understanding the data, preparation of the data, data mining, evaluation of the discovered knowledge and using the discovered knowledge. The main difference between the DMKD model and CRISP-DM is the introduction of feedback loops, shown in Figure 3. For instance, while in the understanding the data step, you might get an insight that makes you want to go back to the previous step of understanding the problem domain. The notion that this is an iterative and interactive process is emphasized.

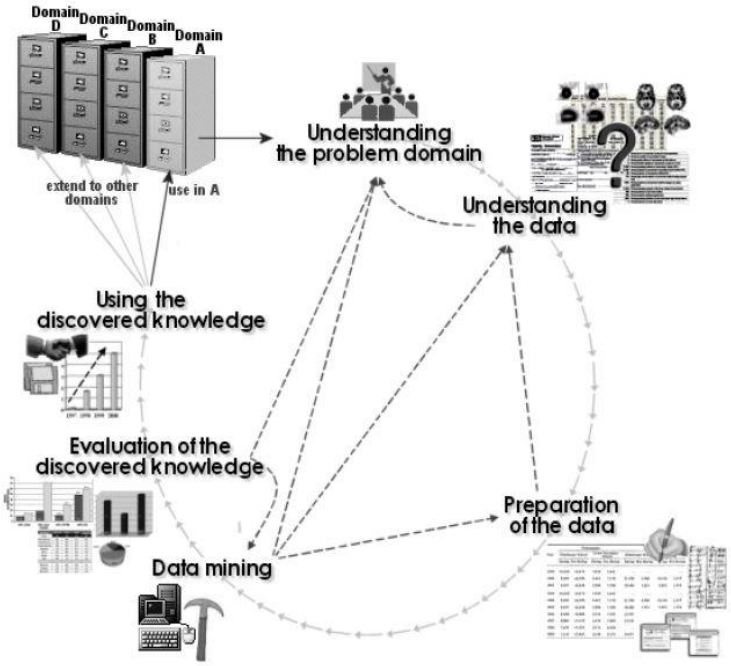


Figure 3 - DMKD model [55]

- The model by Marbán et al.[56] [57], proposed in 2007, also enhances CRISP-DM by considering other dimensions of a data mining project and using standards for the areas that are not the core KDD process. The standards used are two software engineering standard process models: IEEE 1074 [58] and ISO 12207 [59]. Unfortunately, the proposed model is not complete, as the authors themselves say "The model is not complete, as this paper merely states the need for the processes and especially the activities set out in IEEE Std 1074 or ISO 12207 but missing in CRISP-DM. The adaptation and detailed specification of these processes is outside the scope of this paper." [57]

- The Ontology Driven Knowledge Discovery (ODKD) process model was proposed in 2007 by Gottgtroy [60]. It was innovative in trying to combine the prior knowledge (represented in an ontology) and the process of knowledge discovery. The KD part is based on CRISP-DM, but still the integration of the ontologies, resulted in the final model being quite different, with five main steps: ontology preparation (which comprises domain understanding, data understanding and ontology building), ontology analysis, instance preparation (which comprises instance cleaning, instance selection and instance export), modeling and, finally, evaluation (which comprises knowledge extraction, knowledge assessment and ontology learning).

- The Knowledge Discovery and Communication Framework (KDCF) model, proposed by Rennolls and AL-Shawabkeh [61] in 2008 has five steps: data collection and processing, data understanding, data mining/modeling, knowledge understanding and business understanding.

- The Data Mining Life cycle (DMLC) model, initially proposed by Hofmann in 2003 and its enhanced version proposed by Hofmann and Tierney [62] in 2009 has nine steps. These steps (which the authors call processes) are grouped in three stages: hypotheses/objectives preparation stage (composed by the steps business understanding, data understanding and hypotheses/objectives definition), data preparation stage (with the select/sample, pre-process and transformation steps), and a final stage (composed by data mining, evaluation and deployment). The authors also introduce quality assurance into the life cycle, through a control cycle that is a variation of the Plan Do Check Act (PDCA) cycle, also known as Deming cycle or Shewhart cycle [63].

- The ASD-DM model, proposed in 2008 by Alnoukari et al. [64] is based on the agile methodology Adaptive Software Development (ASD). ASD was originally used for software development and replaced the classic Plan-Design-Build lifecycle, with a new Speculate-Collaborate-Learn lifecycle. The ASD-DM uses also these three steps: the speculation step (which now includes business and data understanding, as well as data preparation), the collaboration step (the modeling itself) and finally the learning step (which includes implementation, testing and evaluation). An evolution of this model, published in 2012 by one of its original authors (Alnoukari), includes objectives and hypotheses setting in the speculation phase [65].

- The IKDDM model proposed in 2012 by Osei-Bryson [66] is especially focused on the business understanding phase. Figure 4 shows the process model as a whole, and Figure 5 shows the business understanding phase in detail.

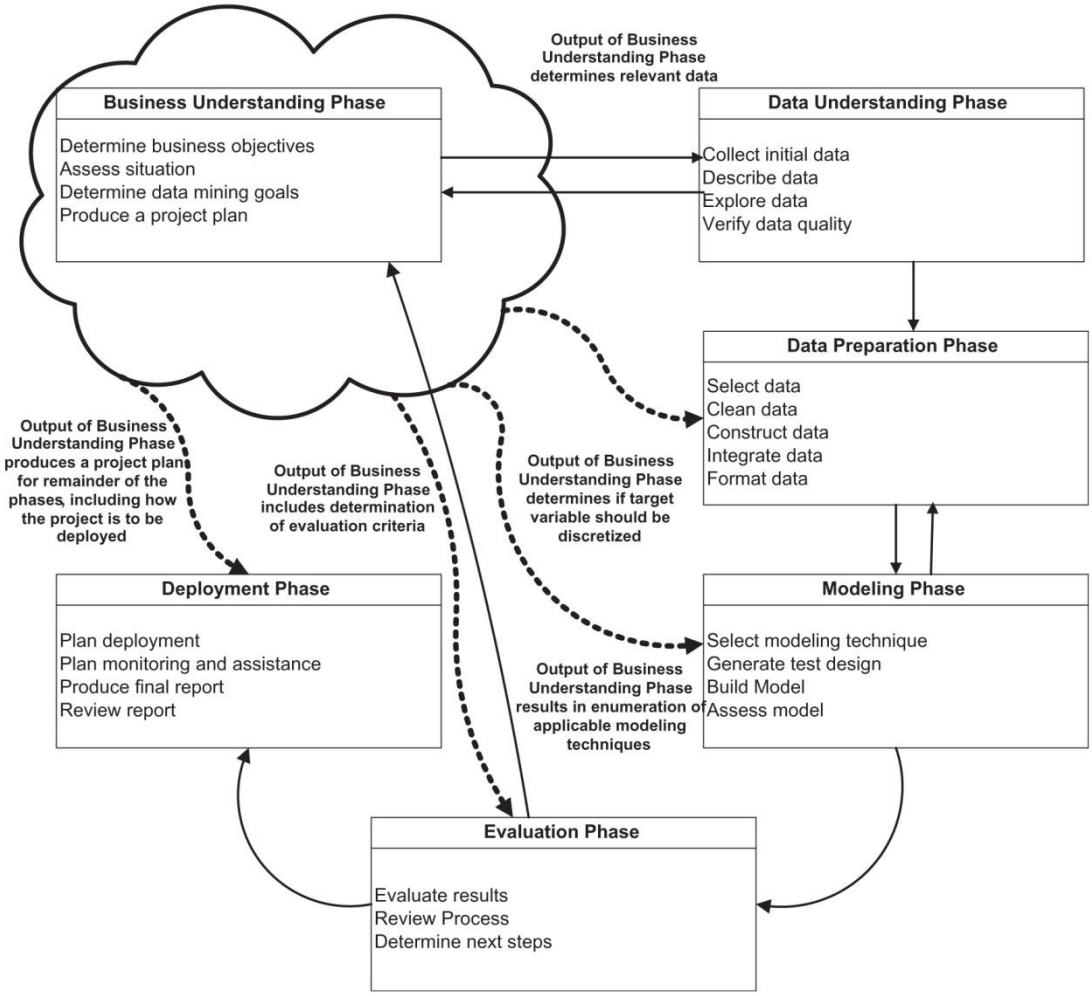


Figure 4 - IKDDM process model [66]

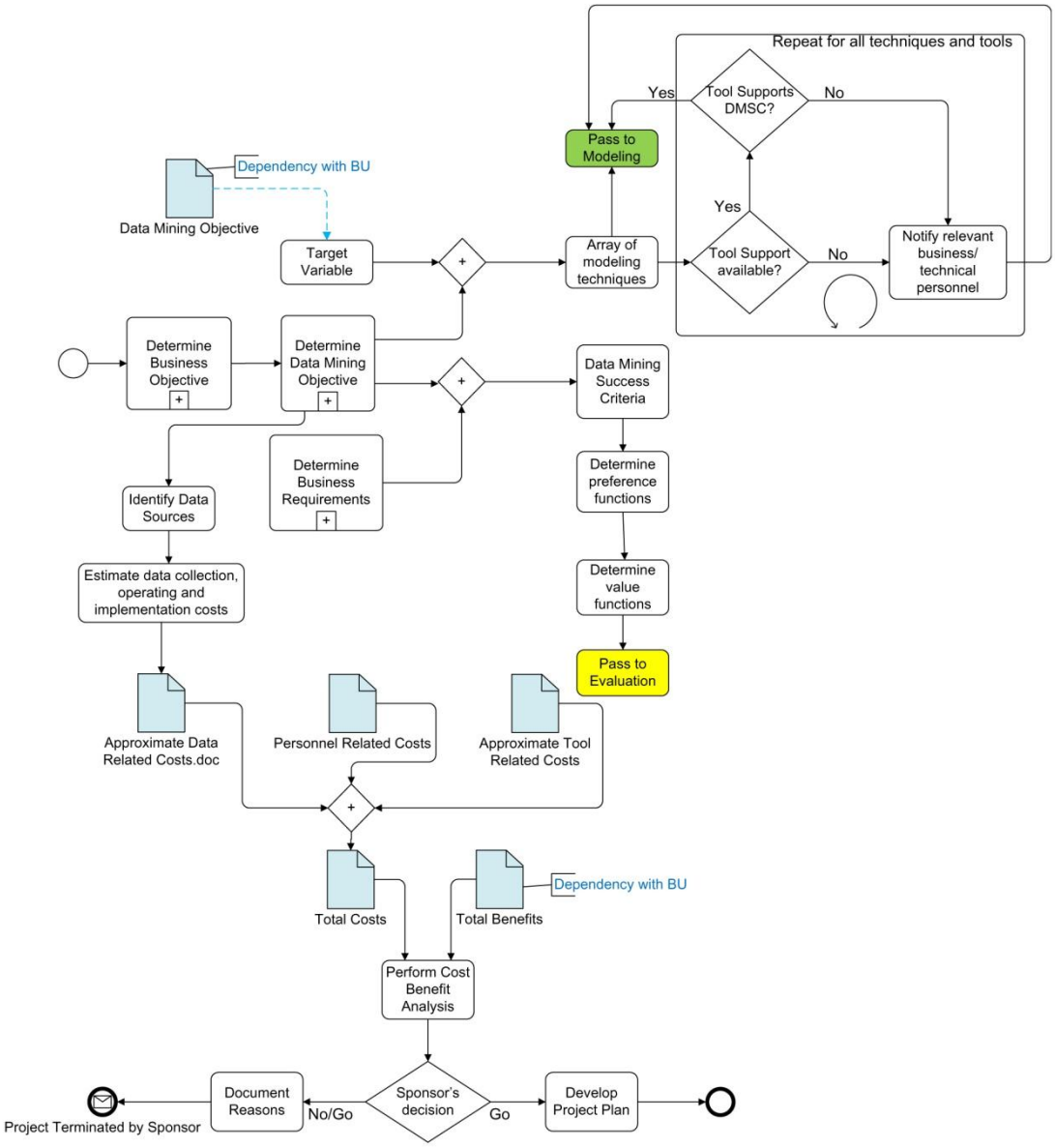


Figure 5 - IKDDM process model, the business understanding phase [66]

There are also some generic models that have been applied, although they are not specific to this domain as the ones above. For instance, [46] mentions the use of the six-sigma paradigm (described in [67]) for quality and excellence in management in the context of data mining, but an adaptation to this particular domain is not made.

In parallel with the evolution described above, there has been a trend for developing Intelligent Discovery Assistants to automate or support the user in an automated way along the KD process. A diagram showing an evolution of the assistants over the years and a possible classification of them is shown in Appendix II.

2.4 Comparing the main process models

In a 2012 survey, it was argued that "The common factor of all data-driven discovery process is that knowledge is the final outcome of this process." [65]

We believe that there is more in common than just the final outcome. That is why we created a diagram that compares the main process models: the original KDD model, CRISP-DM, SEMMA and Catalyst (Figure 6).

We chose the first three models because they are the most used, according to the survey results of Table 1 (page 21). The last model was chosen because we felt it had a different approach from the first three and was quite complete. It also had references in literature referring it as one of the most used [68].

For the original KDD model, the description of section 2.3.1 was used. For CRISP-DM, SEMMA and Catalyst the descriptions of section 2.3.3, 2.3.4 and 2.3.5 was used, respectively.

In Figure 6 we divided the process in 5 stages, each with a different color. The first stage encompasses any learning about the domain, the business or the data. The second stage (in blue) corresponds to choosing and preparing the data. The third stage corresponds to the choice of tool, function and algorithm. The fourth stage (in pink) is the actual data mining and the last one (in purple) refers to what you do with the discovered knowledge; evaluation, refining and deployment.

From the figure it is easy to notice that we always have a sequence of steps that fall into these 5 stages, but that each different model seems to focus on different aspects. For instance, SEMMA totally ignores the first stage and CRISP-DM ignores the third stage. The level of detail also varies a lot.

Figure 6 - Comparison between KDD, CRISP-DM, SEMMA and Catalyst models

3 Requirements engineering

"A lot of times, people don't know what they want until you show it to them."

Steve Jobs

3.1 Definitions

The IEEE Standard Glossary of Software Engineering Technology [69] defines a software requirement as:

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 a 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.

In short, a software requirement is something that we expect the software to meet.

It is important to note that requirements are all about **what** is needed not about **how** it should be built or developed. In other words, "Requirements are necessary attributes defined for an item prior to efforts to develop a design for the item." [70]. This not to say that requirements, design and implementation do not intertwine (as has been noted from the early days, namely in the 1982 paper by Swartout and Balzer [71]).

Requirements of many kinds exist. The simplest categorization is to divide them into functional requirements (FR) and non-functional requirements (NFR). One possible way of elaborating this is shown in [6], in which requirements are divided into:

- Functional requirements

  These concern a result of behaviour that shall be provided by a function of the system.

- Quality requirements

  These pertain to a quality concern that is not covered by functional requirements.

- Constraints

  These cannot be influenced by the team members and limit the solution space beyond what is necessary for meeting the given functional requirements and quality requirements.

A more detailed categorization is made by the FURPS+ model [72] This model uses the acronym FURPS to describe the major categories of requirements:

- **F**unctionality
  These include desired features, capabilities or security.
- **U**sability
  These include human factors, user interface, online help as well as user documentation and training materials.
- **R**eliability
  These include acceptable frequency of failure and any specific needs regarding recoverability, predictability or accuracy.
- **P**erformance
  For a given feature, a performance requirement might specify parameters for speed, efficiency, availability, accuracy, throughput, response time, recovery time or resource usage.
- **S**upportability
  These include any aspects regarding installation, configuration, testing, maintenance (including future extensions and adaptations), localization for different countries or regions and compatibility.
- **+**
  The + in FURPS+ serves as a reminder that there might be other types of requirements that are not included in the previous categories. These might include requirements that are very specific to the project, such as physical requirements.

Finally, if we go back to the IEEE Standard Glossary of Software Engineering Technology [69] there are six types of requirements:

1. Design requirements
   These concern the design of a system (or system component).
2. Functional requirements
   These specify the function that a system (or system component) must perform.

3. Implementation requirements

   These specifically concern the coding or construction of a system (or system component).

4. Interface requirements

   These specify the external elements with which the system or its component must interact, indicating any relevant constraints on formats, timing, or others.

5. Performance requirements

   These imposes constraints or conditions on functional requirements, namely on speed, accuracy, or resource usage.

6. Physical requirements

   These concern the physical characteristic that a system (or system component) must possess, such as material, shape, size, weight.

Having defined what a requirement is, the question that follows naturally is "Where do requirements originate?", or, in other words, "Who says that we need to meet this requirement?". The answer is the stakeholders.

A stakeholder is defined [72] as "an individual who is materially affected by the outcome of the system or the project(s) producing the system". Stakeholder identification is an important part of requirements elicitation, as missing to identify a stakeholder might cause some requirements to be completely ignored.

Finally, it is important to define Requirements Engineering (RE). Sommerville and Sawyer [73] define it as "the process of discovering, documenting and managing the requirements for a computer-based system. The goal of requirements engineering is to produce a set of system requirements which, as far as possible, is complete, consistent, relevant and reflects what the customer actually wants."

Other authors do not limit RE to software or computer-based systems, using a broader definition for RE [74] "Requirements engineering is the branch of engineering concerned with the real-world goals for, functions of, and constraints on systems. It is also concerned with the relationship of these factors to precise specifications of system behavior and to their evolution over time and across families of related systems.".

Other authors point out that [75] "RE is a multi-disciplinary activity, deploying a variety of techniques and tools at different stages of development and for different kinds of application domains.".

It is important to note that RE is more than requirements elicitation, that is, eliciting requirements from the stakeholders. And eliciting requirements is more than talking with the

stakeholders. That corresponds to one of the elicitation techniques (open-ended interviews), but there are many others [76] (the list of elicitation techniques is shown in Appendix I).

According to the classical 1998 textbook "Requirements Engineering: Processes and Techniques" [77] the activities that compose RE are:

- Elicitation
- Analysis and Negotiation
- Documentation
- Validation
- Management

Other authors [78] consider different activities:

- Elicitation
- Documentation and definition
- Specification
- Prototyping
- Analysis
- Review and validation
- Agreement and acceptance

The authors stress that "Not all of these activities involving requirements are needed to the same degree for all software projects.". This is evident when we look at the historical evolution of RE in section 3.2. We analyse in more detail what the RE process is like in section 3.2.6 (page 43).

3.2 Historical overview

There certainly has been a major shift in the way requirements engineering is seen within software development projects in general. As Tom DeMarco put it in 2013 article "When I began my career, technologists (programmers) performed a requirements analysis prior to undertaking the real work of the project: coding and debugging. Today, requirements analysis is performed by a mixed team of systems and business people, and this is the real work of the project."[1][79]. This shift can also be observed through an historical overview of

[1] Tom DeMarco started his carrer in 1963.

how software development, in general, and requirements engineering, in particular, evolved over time.

3.2.1 Waterfall

Winston Royce [12] is often credited as the creator of the waterfall model in 1970 (for instance in [80], [81] and [82]). The model referred by Royce is depicted in Figure 7, adapted from [12].

The waterfall model is a simple model, in which progress is seen as flowing from one phase of software development to the next in a sequential way.

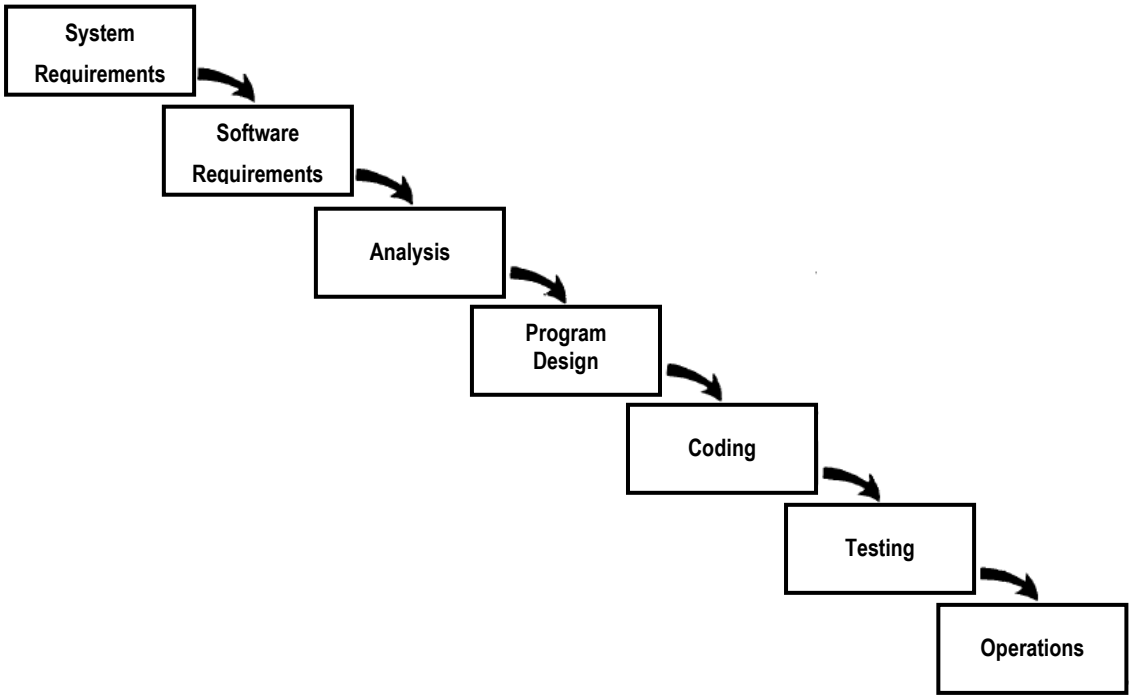


Figure 7 - Waterfall model, adapted from [12]

It is ironic that Royce, in fact, described this model as a model that he did NOT recommend for large-scale software development. He refers to the Waterfall model as "the simpler model" (the word waterfall is not used) and he then proposes an enhanced model that includes building a prototype in an early stage and had an emphasis in feedback and documentation as seen in Figure 8.

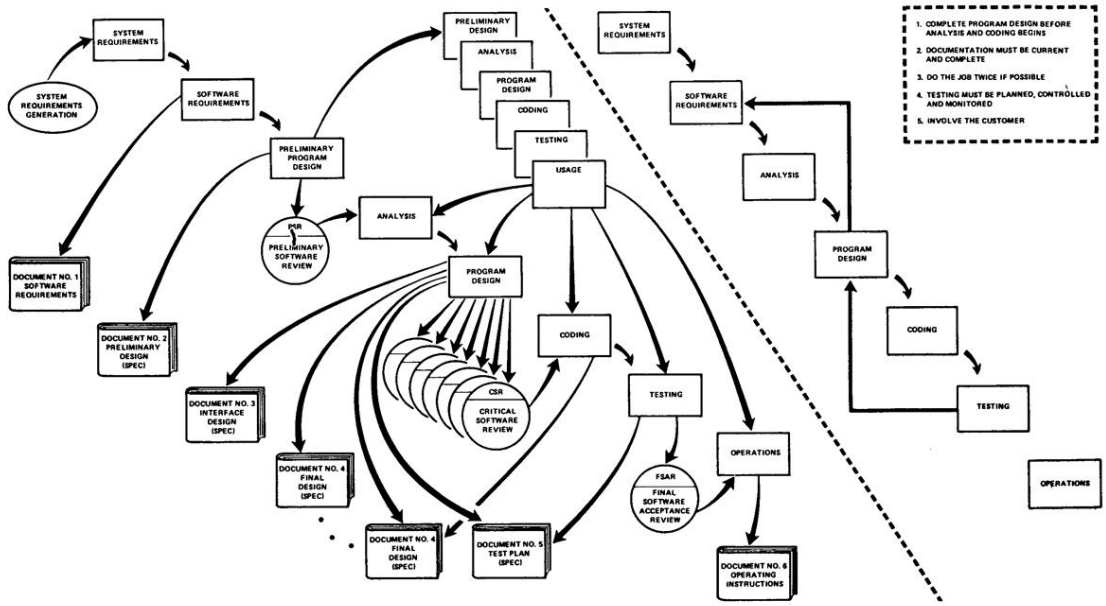


Figure 8 - Enhanced model for the development of large software systems [12]

Quoting [12], "If the relatively simpler process (…) would work successfully, then of course the additional money is not well spent. In my experience, however, the simpler model has never worked on large software development efforts".

Focusing on the requirements aspects of this software development model, we see that the deceptively simple "System requirements" and "Software requirements" implies that there is a set of requirements that can be reasonably determined in the beginning of the project (that is, "up front"). The requirements are fixed and then the project development proceeds.

This "fixed requirements" assumption has been found to be a root cause of project failure. For example, in a study done in the United Kingdom and published in 2000 [83] in which 1,027 Information Technology (IT) projects were followed, only 130 were found successful in the end and over 70% of the participants referred failure occurring during the requirements definition. The study author concluded that "the approach of full requirements definition, followed by a long gap before those requirements are delivered, is no longer appropriate. The high ranking of changing business requirements suggests that any assumption that there will be little significant change to requirements once they have been documented is fundamentally flawed.".

3.2.2 Spiral

The spiral model was proposed in 1986 by Barry Boehm [84] (Figure 9). Requirements still have a strong early placeholder, as the model begins by determining the objectives,

alternatives and constrains. However, the fact that the spiral is possibly passed multiple times, means that this is the first model that is iterative and incremental in terms of the requirements.

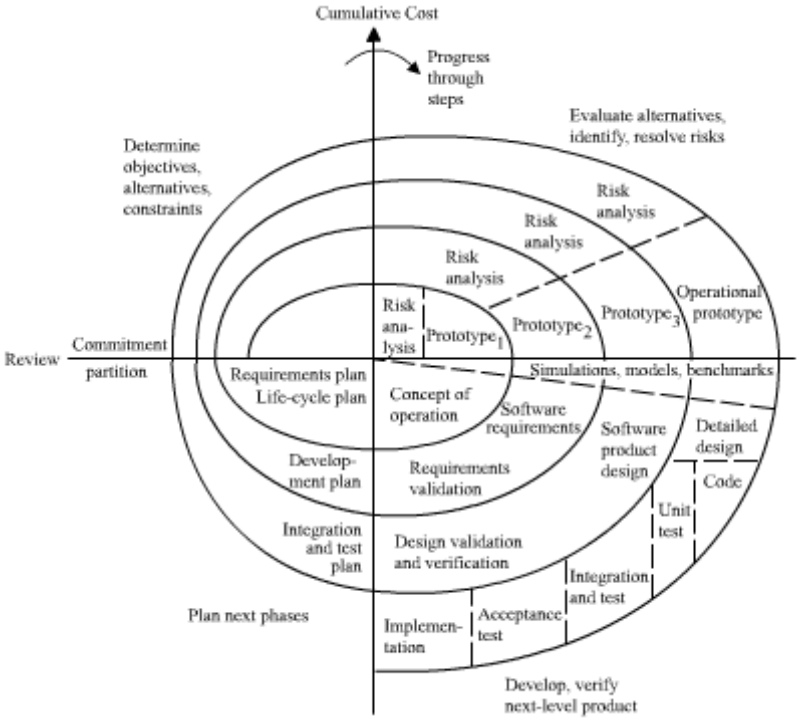


Figure 9 - Spiral model (from [85], based on [84])

3.2.3 Rapid Application Development (RAD)

The Rapid Application Development (RAD) methodology was developed to respond to the need to deliver software very quickly. RAD was first developed during the mid-1970s at the New York Telephone Company.

Nonetheless, it only became notorious in the early 1990s, when James Martin published his approach [86]. Martin's methodology is iterative development and based on the construction of prototypes. These prototypes are used for both requirement elicitation and validation.

The general idea was to take advantage of the more powerful development software: if you could build it fast enough before the requirements changed, you would be more successful. And if you did get it wrong, you would simply build it again.

3.2.4 Rational Unified Process (RUP)

The Rational Unified Process (RUP) is a widely adopted, iterative and incremental software process model. It was created by the Rational Software Corporation and is now actively promoted and supported by the Rational Software Division of IBM [87].

The RUP has determined a project life-cycle consisting of four phases: inception, elaboration, construction and transition. Each phase has one key objective and milestone at the end. When visualizing the RUP phases over time (in what is referred to as the RUP hump chart, seen in Figure 10) the overlap of the various activities that occur during the life cycle phases is clearly visible. This overlap is recognized as necessary.

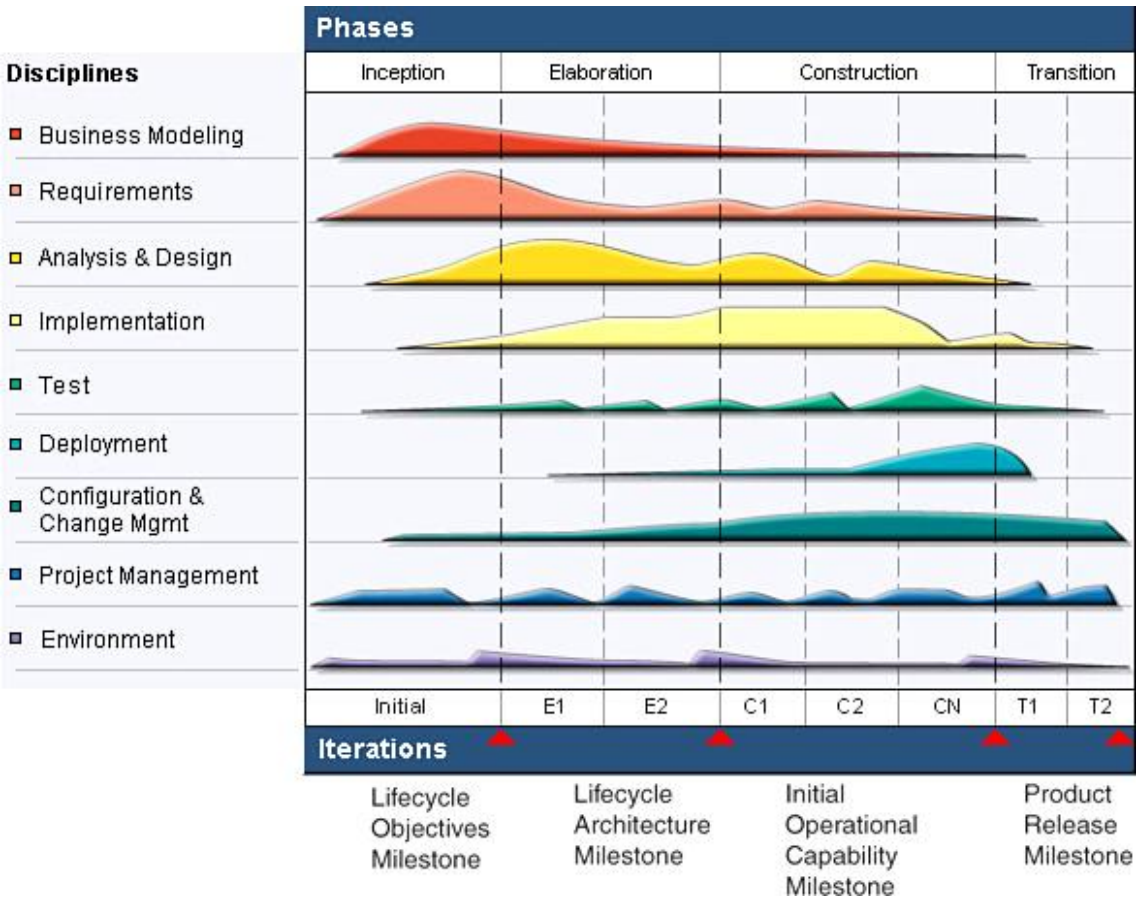


Figure 10 - RUP hump chart [88]

Considering specifically the requirements discipline, we can see in Figure 11 the RUP workflow for the requirements, in which various activities are performed.

The Analyze the Problem activity is one of the most important. It consists of finding who the stakeholders are, what the scope of the system is, and what the different constraints are.

One of the other key artifacts to be produced is the Use-Case Model, originally proposed by Ivar Jacobson [89]. The Use-Case Model tries to capture the system's intended functions and its environment. It also serves almost as a contract between the customer and the developers. It is used as an essential input to activities in Analysis and Design, Implementation, and Test.

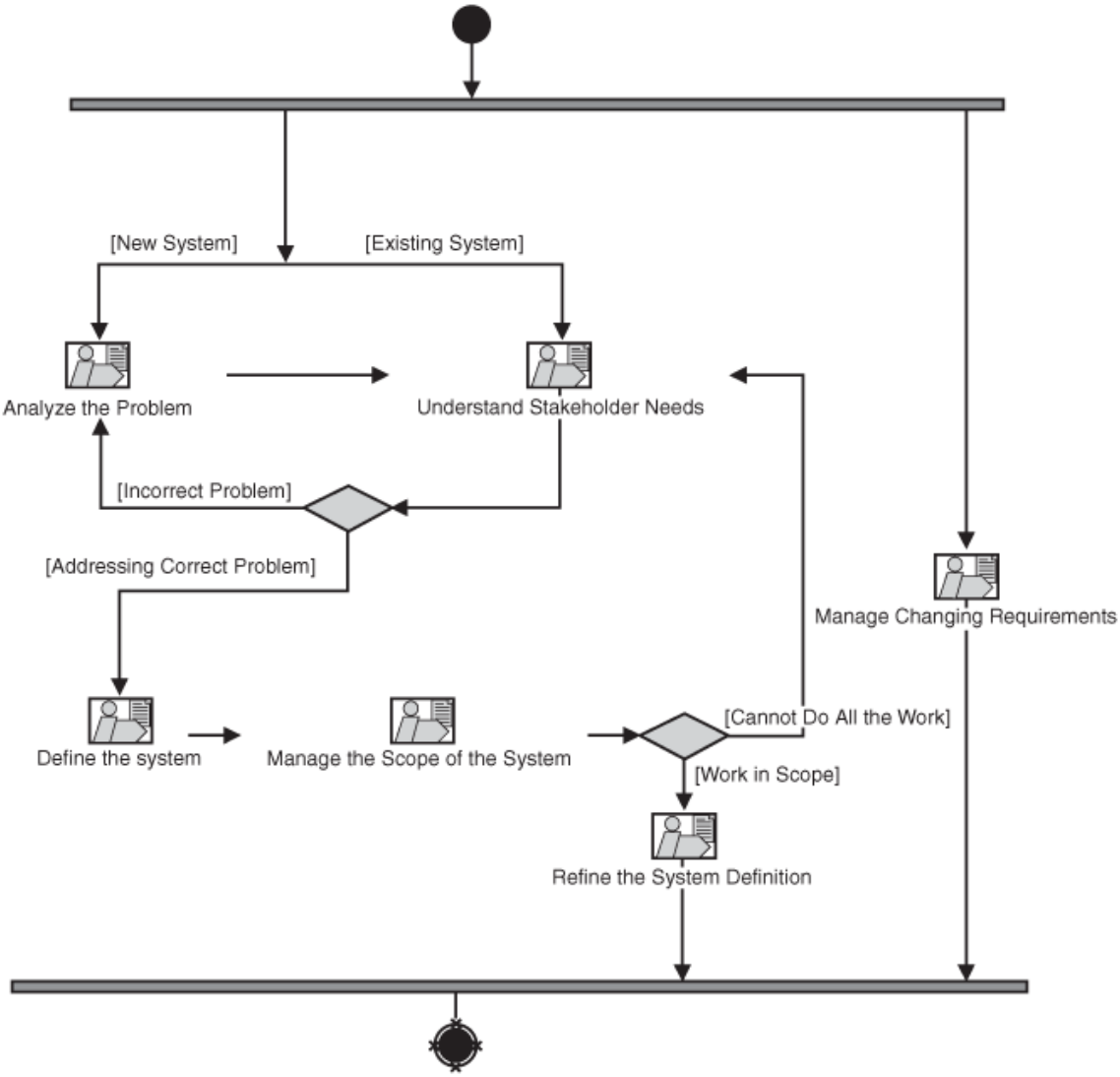


Figure 11 - RUP requirements workflow [88]

There currently are some lighter-weight versions of RUP, such as OpenUP (Open Unified Process). OpenUP was actually created by IBM itself [90] in 2006 and is a subset of RUP tailored for the delivery of Agile projects. OpenUP was released as an OpenSource method through the Eclipse Project Framework [91].

3.2.5 Agile

The term Agile software development was coined in 2001 in the Agile Manifesto [92]. This manifesto presents twelve principles. Several methodologies apply the Agile Manifesto principles, for example, Adaptive Software Development (ASD), Extreme Programming (XP) and Scrum.

In terms of requirements, the Agile approach is fundamentally different, no matter the specific method.

This stems from the first two principles of the Agile Manifesto:

- Manifesto principle #1: Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.
- Manifesto principle #2: Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.

Highsmith and Cockburn, two of the manifesto authors specifically point out that [93] "Traditional approaches assumed that if we just tried hard enough, we could anticipate the complete set of requirements early and reduce cost by eliminating change. Today, eliminating change early means being unresponsive to business conditions—in other words, business failure." In the Agile software community the attempt to anticipate the complete set of requirements early and follow that with the complete design is mockingly known as BDUF - Big Design Up Front.

Nonetheless, whether BDUF is a mistake or not is a controversial topic. Boehm [94] argues that it depends on the project, stating that (referring to the first two principles above) "developers can misapply them, with disastrous results. Plan-driven methods work best when developers can determine the requirements in advance - including via prototyping - and when the requirements remain relatively stable, with change rates on the order of one percent per month. In the increasingly frequent situations in which the requirements change at a much higher rate than this, the traditional emphasis on having complete, consistent, precise, testable, and traceable requirements will encounter difficult to insurmountable requirements-update problems. Yet this emphasis is vital for stable, safety-critical embedded software.".

Some authors [95] note that in practice, agile methods do try to anticipate the requirements, but that the difference is on the level of detail that is expected: usually the effort of gathering the details is postponed until the requirement needs to be fulfilled in the next iteration. For this reason, they refer to agile requirement elicitation as "lazy requirements elicitation".

Two other principles are important to understand the agile view on requirements:

- Manifesto principle #4: Business people and developers must work together daily throughout the project.
- Manifesto principle #6: The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.

Having stakeholders involved on software development projects always increases the chances of project success. Because of the two principles above, on agile projects, there is an attempt that customers and other stakeholders are engaged continuously throughout the project and face-to-face meetings are favoured. It is also common to have a product owner who represents the customers work on-site with the development team. Some authors [95], however, believe that this is an "idealized picture of stakeholder involvement" and that it is almost an utopia to find a product owner that "can answer all developer questions correctly, (…) is empowered to make binding decisions and able to make the right decisions".

Yet another difference in the Agile approach to requirements is the use of frequent review meetings and acceptance tests for requirements validation. These review meetings are often based on the delivery of an increment, that is, a working software version that adds something to the previous one.

3.2.6 GORE

Goal-Oriented Requirements Engineering (GORE) was a term that evolved from what was originally called goal driven (or goal directed, goal based) requirement engineering research in 1990s [96], [97] [98], [99], [100]. It includes several methods, such as KAOS [96], [101], i* [102], GBRAM [103] and NFR [104] (the latter concerning non-functional requirements, as the name implies).

It was developed in further research since 2000 [105], [106], [107], [108]. The idea behind GORE is that requirements can be justified if they are linked them to goals. A goal is a high-level, long-term objective owned by, at least, one of the stakeholders.

The authors of the GORE methods combine goals, requirements, and concerns in a tree-like structure. Each goal is refined by a list of sub-goals and requirements. The leaves of the tree present system requirements. It is important to note that goal identification is not done exclusively from either a top-down or a bottom-up approach. In most cases the two approaches are combined. Refining goals is done by a decomposition approaches, the most popular being the "milestone approach".

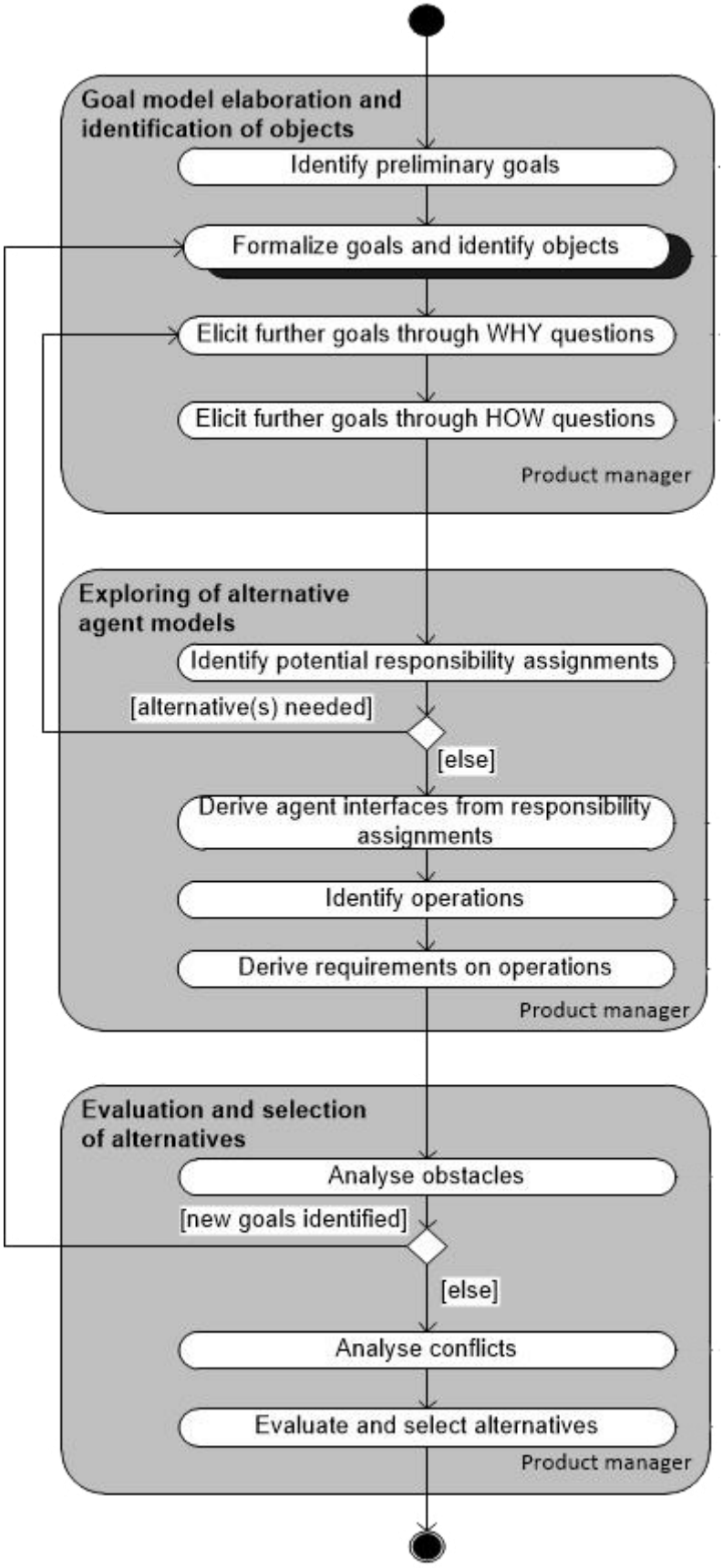


Figure 12 - KAOS process workflow, adapted from [109]

3.3 Requirements engineering process models

It is important to note again that we will consider a process model as the set of framework activities and tasks necessary to get the job done, including inputs and outputs in every task [25].

In research the majority of requirements engineering process models show it as an ordered set of activities, with most process models incorporating some iteration and feedback, as is the case with model shown in Figure 13, in which the considered activities are elicitation, analysis, specification and validation. We will go into further detail of each of these activities includes in section 3.3.2.

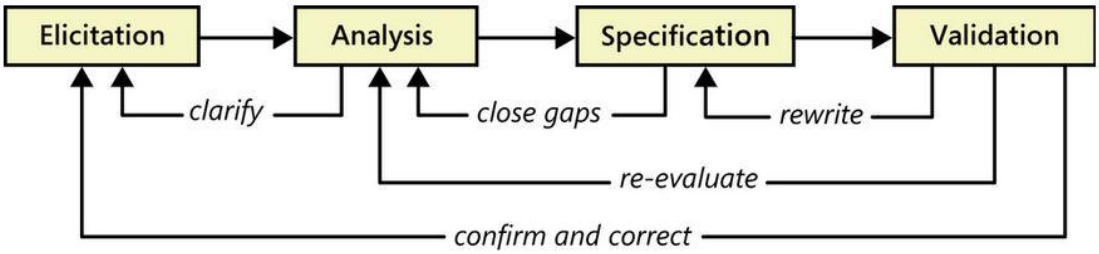


Figure 13 - Requirements engineering process model [110]

Others incorporate some parallel activities, as is the case with the model proposed in 1995 in [111] (Figure 14) and the model from the "Essentials of Software Engineering" textbook (third edition in 2013, Figure 15).

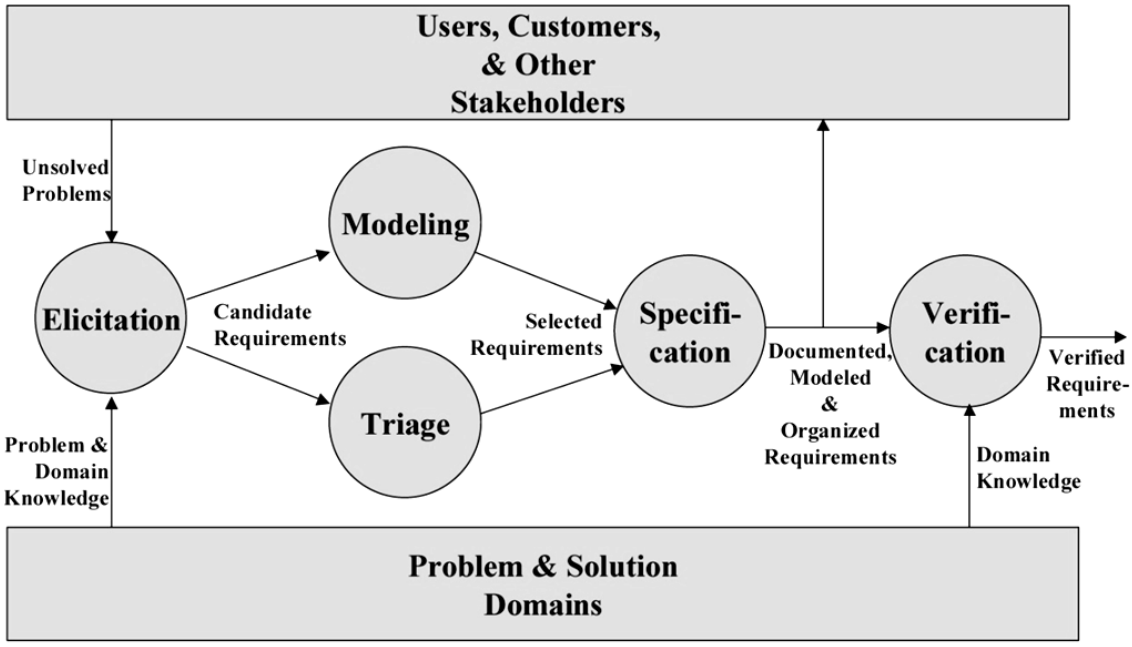


Figure 14 - Requirements engineering process model [111] (figure taken from [112])

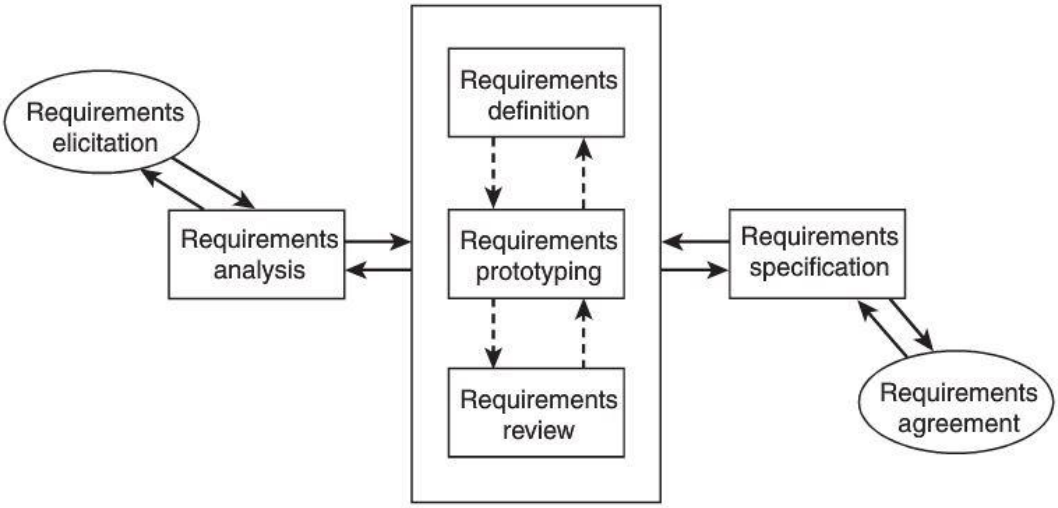


Figure 15 - Requirements engineering process model [78]

3.3.1 Volere requirements process

The Volere requirements process was developed by the consulting firm Atlantic Systems Guild and its name is derived from the Italian verb *volere* (to want, wish) [113]. The process is iterative and is organised according to the following points:

- Motivation
- Restrictions and specifications for the project
- Functional requirements
- Non-functional requirements
- Project information (such as risks and costs)

All the information about the requirements is held in a single document that is developed based on a requirements template. Quality assurance is included as an intermediate step (refered to as “Quality gateway”).

The Volere requirements process is graphically presented in Figure 16.

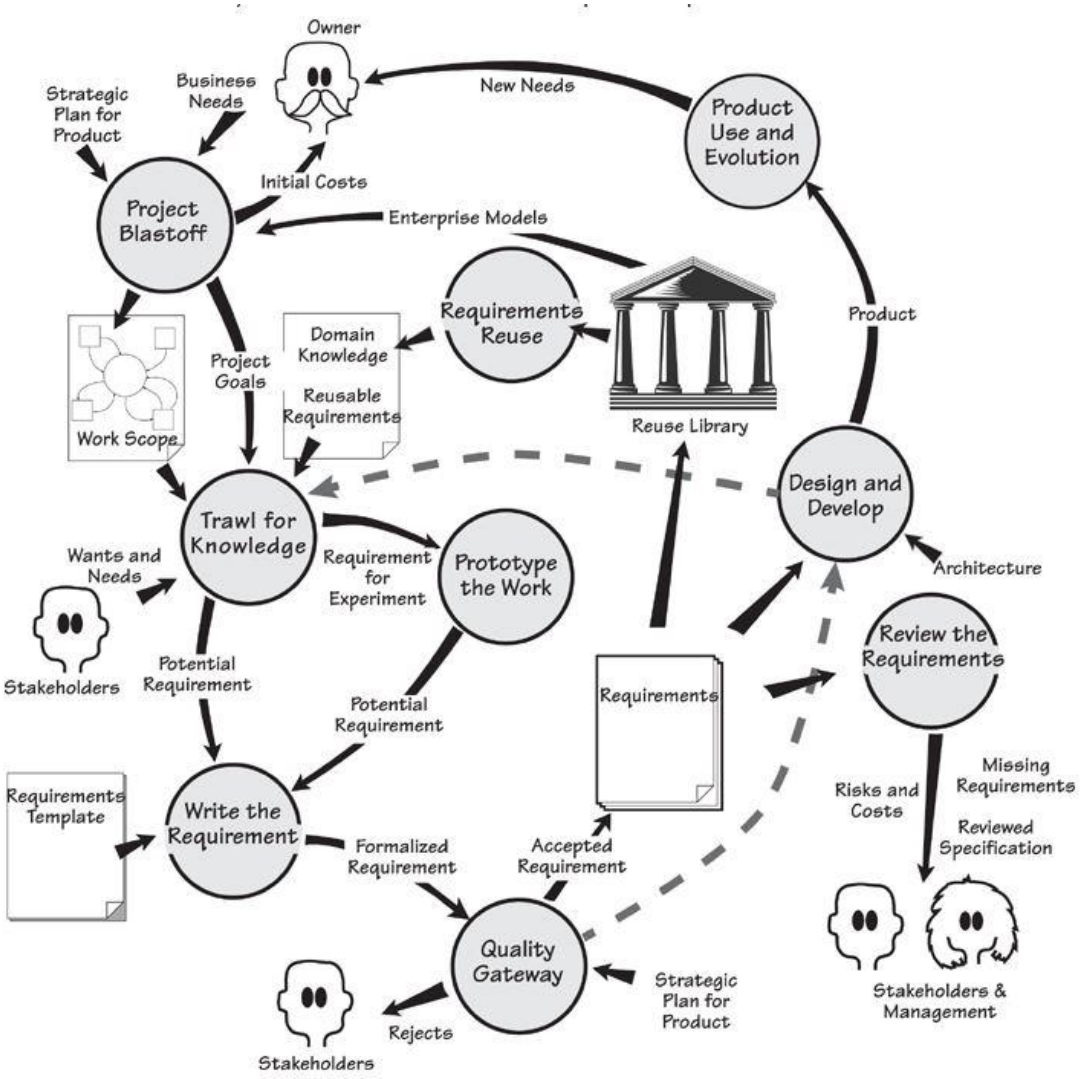


Figure 16 - Volere requirements process [113]

In these process, the requirements are not seen as something that can be gathered, but rather as something that evolves, having a life cycle that will bring them from embryos to adults (Figure 17).

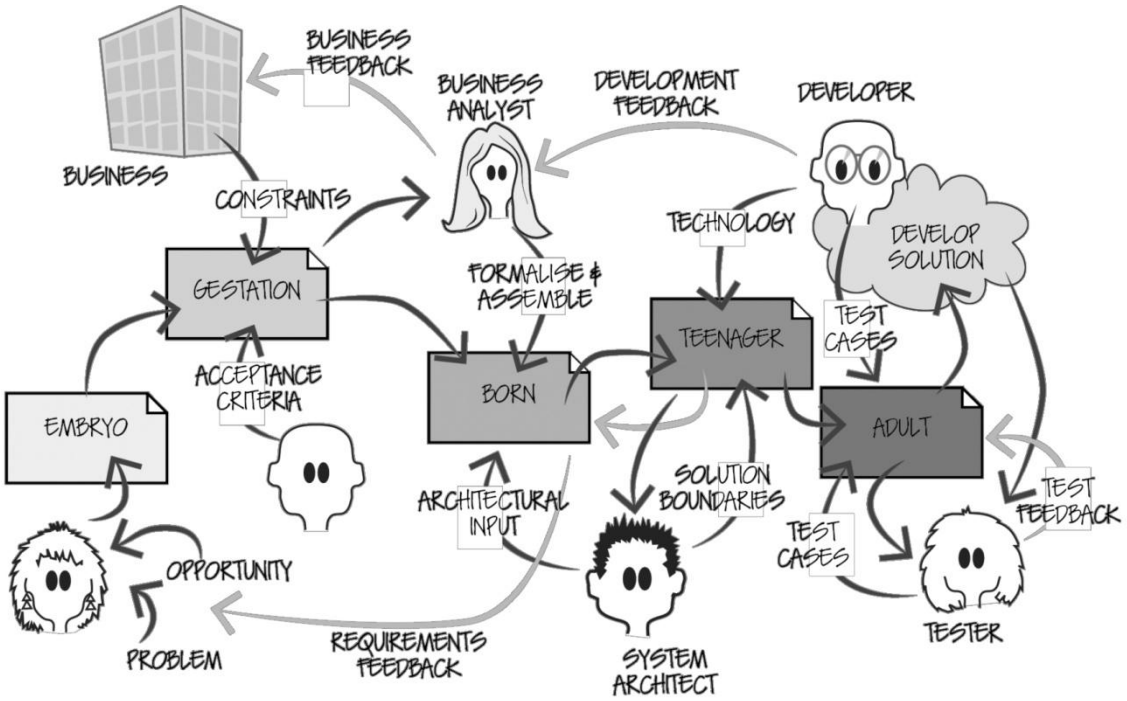


Figure 17 - Evolution of a requirement during the Volere requirements process [114]

3.3.2 R-CMM process

The Requirements Capability Maturity Model (R-CMM) [115] provides guidelines for practitioners to understand their own RE process and go through an RE process improvement. The R-CMM is an adaptation for the RE area of the Capability Maturity Model for Software (CMM) developed by the Software Engineering Institute of the Carnegie Mellon University [116].

The R-CMM identifies five phases of requirements. Each phase in the R-CMM defines a set of processes. This model recognizes a total of 68 processes, but we will only list some of them in the corresponding phases:

1. Elicitation

   The system requirements are discovered by consulting with the stakeholders and by analysing system documents, domain knowledge and market studies. The following processes are part of this phase:

   - Assess system feasibility
   - Identify and consult system stakeholders
   - Record requirements sources
   - Define the systems operation environment
   - Look for domain constraints

- Record requirements rationale
- Collect requirements from multiple viewpoints
- Prototype poorly understood requirements
- Use scenarios to elicit requirements
- Define operational processes. Reuse requirement

2. Analysis and Negotiation

   A detailed analysis of the requirements is done and the multiple stakeholders negotiate to decide which requirements are accepted. The following processes are part of this phase:

   - Define system boundaries or scope
   - Prioritise requirements
   - Assess requirements risks

3. Documentation

   The agreed requirements are documented at what is judged to be an appropriate level of detail.

4. Validation

   Requirements are validated, that is, checked for completeness and, most of all, for consistency.

5. Management

   All the previous requirements engineering phases and activities are planned and controlled by the management phase.

It is important to go into more detail regarding the assessment of requirement risks. According to [110], “a risk is a condition that could cause some loss or otherwise threaten the success of a project”. In other words, risks can be seen as potential problems. A risk could have a negative impact on:

- Project’s schedule
- Project’s cost
- Software’s quality
- Project team’s effectiveness

Requirements risk assessment is the process of identifying and evaluating risks that are related to each specific requirement. This is an important step so that we can:

- Decide if the requirement is included in the system’s scope
- Decide the requirement priority
- Later manage the risk (which includes avoidance and, if necessary, control)

Requirement risks can result from many factors, such as:

- Dependencies - If a requirement depends on other requirements or specific resources (human or not), it has an increased risk
- Technically difficult features - It is important to identify early on the requirements that might take longer than anticipated to implement because of their inherent technical difficulty. These requirements might require prototyping to evaluate their feasibility or estimate implementation time.
- Unfamiliar technologies - It is key that the learning curve is taken into account for any new technologies that are needed to satisfy certain requirements. The same goes for new methods, languages, tools, or hardware.

4 Requirements engineering for KD

"There is no unique picture of reality."

Stephen Hawking

4.1 Why is requirements engineering for KD different?

The different process models discussed in historical overview of section 2.3 consider requirements specification as one of the early activities of the project. In this sense, there is a parallel to what happens in the software engineering process models discussed in section 3.2. In both we find the need to discover or elicit the needs of the stakeholders as soon as it is possible.

Requirements engineering for KD is, however, quite different mainly because there normally is no software being produced and little or no code being written. The result of a KD project is knowledge and that must be delivered in a usable form to the stakeholders. Available software tools that include the necessary techniques and algorithms are used. There has been significant research in the area of selecting Commercial Off-The-Shelf (COTS) software products [117], [118] but in the KD area it is not only a matter of selecting the right product or tool but also the data mining technique, the algorithm and the best parameters.

Another difference still is that the final users might be unaware of what potential outcomes a KD project might bring them. Most organizations have not previously worked with KD projects and therefore they do not get a chance to mature through experience. KD projects are like a bright light entering a dark attic: You do not really know what you will find… until you find it. Some authors describe the final users' typical feedback to be "Ah! Now that I see what the possibilities are, I can tell you what I really want to see. But until I know what the possibilities are I cannot describe to you what I want." [18].

For this reason, if early delivery of a smaller version or a prototype is viable, it might be a great way to educate users about what to expect and to make elicitation of requirements is more effective.

Another difference, noted in [119] and represented in Figure 18, is that in this type of project, you can use either:

- A **demand driven** approach, aimed to determine the information requirements of the users (similar to the traditional requirements) or
- A **supply driven** approach, in which you start with the analysis of the data sources available.

In the figure it is clear that everything originates from the requirements on a demand driven approach, while on a supply driven approach everything originates from the data.

Some authors even go to the extreme of defending that only a supply driven approach is viable, especially if the KD project is a data warehouse [18].

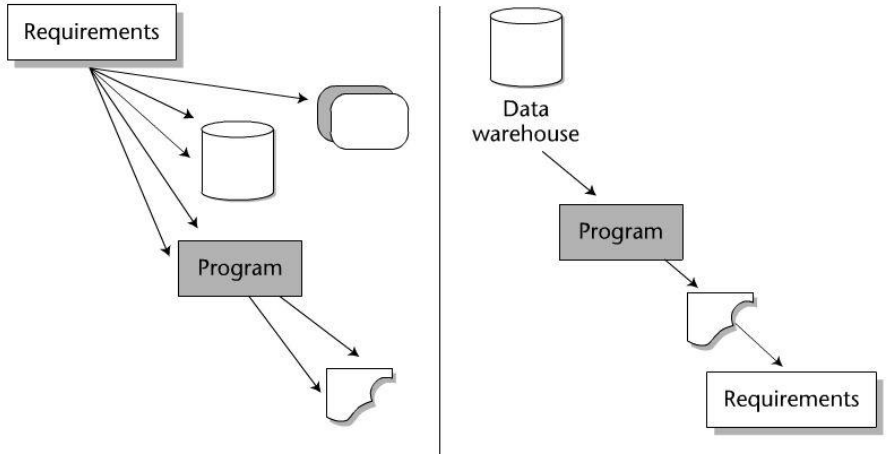


Figure 18 - Demand driven approach (left) vs. supply driven approach (right) [18]

Yet another difference is that KD projects might have a minimal user interface. This might give the developers a false sense that the only need to elicit the requirements for the minimal user interface and they can figure out the rest for themselves, as mentioned in [120].

To sum up, there is no requirements engineering solution for all problems. One of the most-quoted papers in software engineering, Frederick Brooks' 1986 essay "No Silver Bullet" [121], argues that there is no way to make software development easy. There is no silver bullet for requirements engineering either. Instead of trying to have a one-size-fits-all solution, focusing on how to make the most of the specificities of KD projects seems much more promising.

4.2 Need for a requirements engineering for KD process model

A specific process model for requirements engineering for KD is needed because of the differences between RE in normal software engineering projects and KD projects presented in section 4.1.

Some authors have already presented process model proposals for RE in KD projects, arguing for them with some of those differences.

For instance, following from the fact that in many cases, only pre-available tools and algorithms will be used, some authors [122] describe requirements elicitation as the sequence of steps seen in Figure 19.

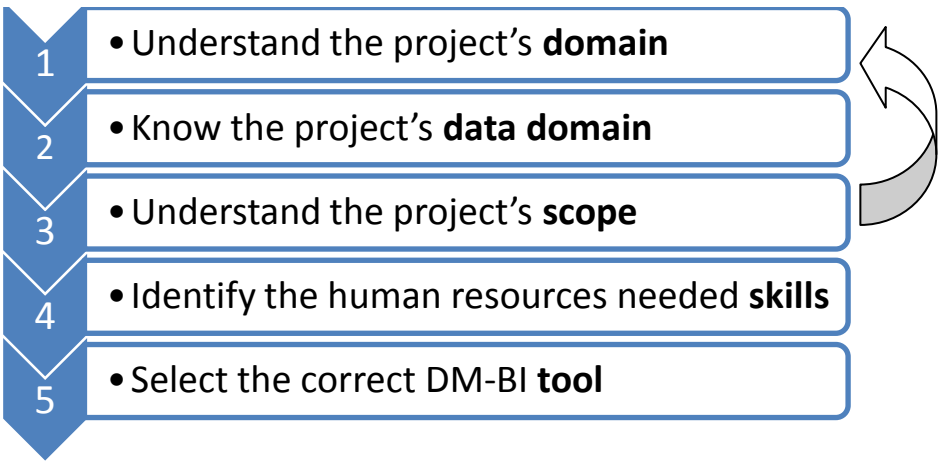


Figure 19 - Process of requirements elicitation adapted from [122]

Following from the fact that you can use a demand driven approach or a supply driven approach, in [119] the authors propose an hybrid approach that is shown in Figure 20.

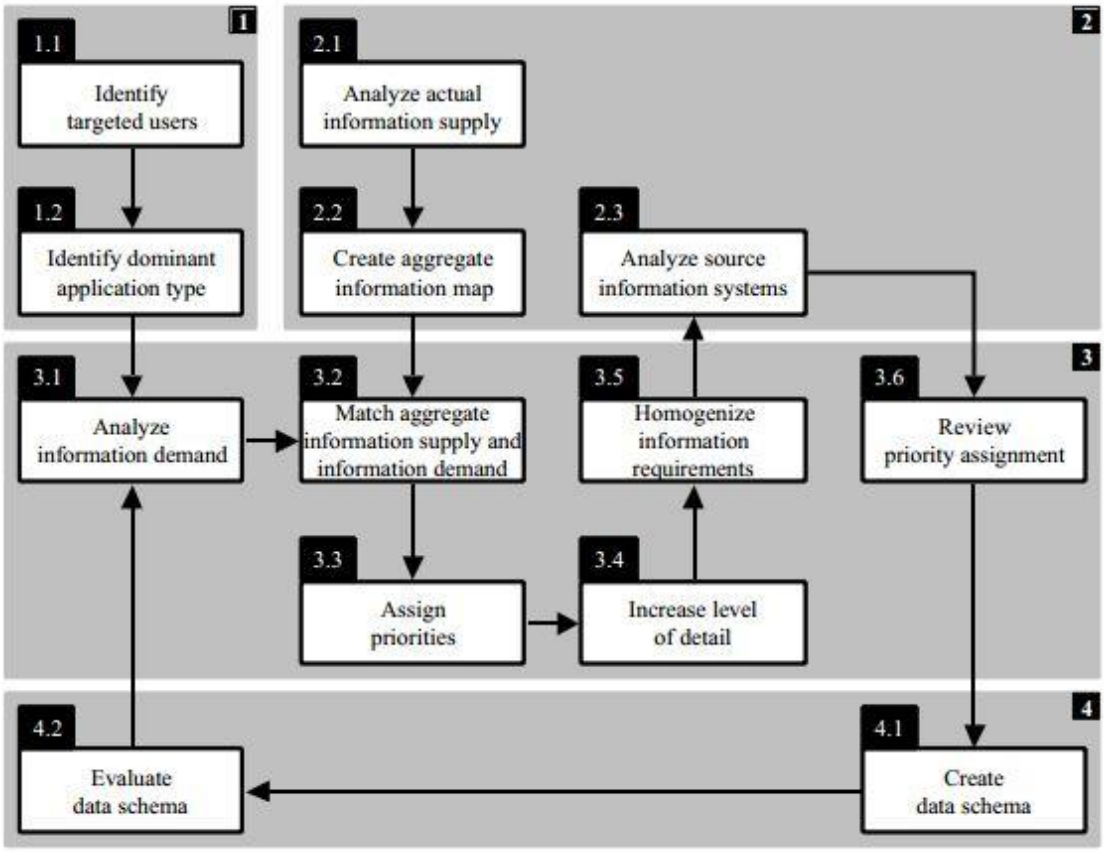


Figure 20 - Model for a hybrid KD requirements methodology [119]

Other authors stress that a process model for RE in KD projects is needed to reduce the failure rate of this type of projects. This is the motivation for a recent work from 2012 [68]. Here the focus is just on the initial activities of RE, which the authors defend to be determining the feasibility of the KD project and estimating the effort involved. For the effort estimation, eight cost drivers are proposed:

- Information Mining objective type
- Level of collaboration of the organization
- Quantity and type of the available data repositories
- Quantity of tuples in main table
- Quantity of tuples in auxiliary tables
- Knowledge about data sources
- Knowledge and experience of the mining team
- Functionality and usability of tools

These drivers are combined in a Cocomo-like linear equation.

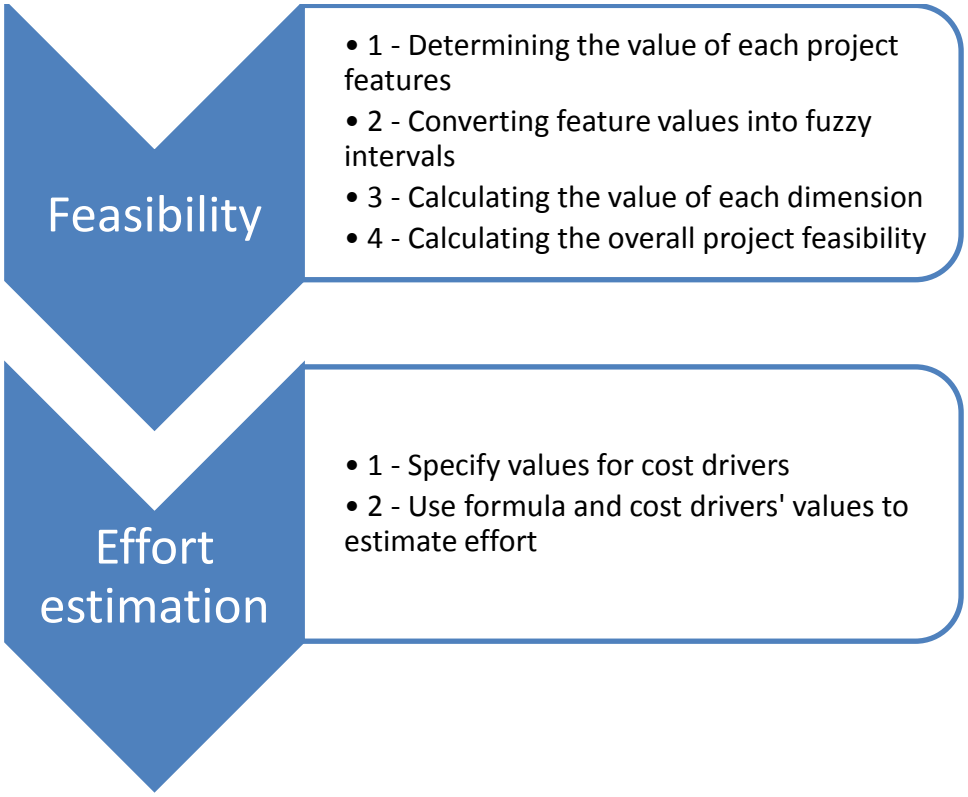


Figure 21 - Initial activities of a KD project [68]

In [123], an approach to RE is presented with data warehouse systems in mind (named DWARF, a Data WArehouse Requirements deFinition technique). The authors describe their approach as the following sequence of steps:

1. Requirements Management Planning
2. Requirements Specification
    a. Requirements Elicitation
        i. Interviews
        ii. Workshops
        iii. Prototyping
        iv. Scenarios
        v. NFR Framework for the non-Functional Requirements
    b. Requirements Analysis & Negotiation
    c. Requirements Documentation
    d. Requirements Conformance
3. Requirements Validation
4. Requirements Management Control

These steps are then organized in the process model shown in Figure 22.

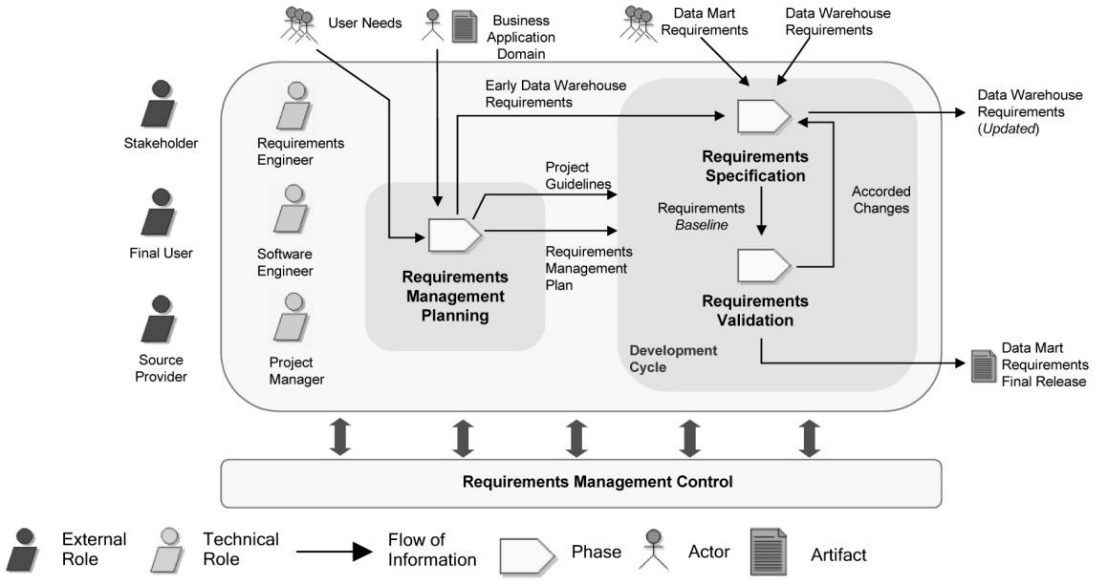


Figure 22 - DWARF process model for RE for data warehouse systems [123]

5 Proposed Knowledge Discovery Process

"Without a repeatable process, the only repeatable results you are likely to produce are errors."

Randall Macala

5.1 Roles and human resources

Some authors [124] argue that many different types of human resources are necessary to approach and implement a data mining project successfully, especially in large projects:

- Business Analyst
- Data Analyst
- Data Engineer
- Domain Expert
- Data Miner
- Knowledge Engineer
- Strategic Manager
- Project Manager

We will consider these as roles that can be performed by one or more actual persons. In many data mining projects, for instance, scientific data mining projects or projects for small to medium companies these roles are all performed by one or two persons (and some roles, like the business analyst might be non-existent altogether in specific types of projects).

5.2 Process description

In this section we will describe the Knowledge Discovery Process (KDP) as we see it after analysing the existing process models. We will go into more detail with what regards Requirements Engineering within the KDP.

We will specifically consider business KDPs, but we believe that this description would also be accurate for other types of organizations, namely governmental or non-profit.

The knowledge discovery process begins when the business analyst realizes that there is a business problem or opportunity in which Knowledge Discovery and Data Mining might be

helpful. More commonly, the business analyst starts with a question and needs certain information relevant to the decision he must make.

He starts by trying to learn as much as possible about the business and the application domain. He will identify the stakeholders. He will try to understand what issues are important for the stakeholders. The five core issues are [45]:

- Product (goods or services, tangible or intangible)
- Place
- Price
- Time
- Quantity

The business analyst will classify the knowledge discovery process as:

- **Demand driven**
  The process is aimed to fulfil the information requirements of the users
- **Data driven**
  The process is aimed to discover the best use to the specific existing data
- **Exploratory**
  The process is designed to find how KD and DM in general can offer value within that specific business

He will try to discover any relevant prior knowledge, namely the currently existing solutions for the problem, and identify the goal for the project.

If it is an exploratory process, the business analyst will identify several possible goals and review his stakeholders' identification for each one (including the core issues that each one might be concerned with).

Since starting the project might have costs, the business analyst might have to ask for approval for the data mining project to the business manager. The business manager might ask the project manager for a cost and resources estimation so that he can decide on the approval. The project manager will create the cost, time and resources estimates or a project plan, if necessary. The project manager will hand these to the business manager. The business manager will decide to go ahead or not, that is, he will decide on the feasibility of the KD project. If the decision is to go ahead, the project manager might have to get the resources (human or otherwise) that are necessary and that were not available in the beginning.

If it is a demand driven project, the business analyst will then begin eliciting specific requirements. If it is a data driven project, the business analyst will then proceed by asking the data analyst to perform the data analysis. A hybrid approach is also possible, in which both will happen in parallel.

For the requirements elicitation, the business analyst will choose the elicitation techniques (list in Appendix I), which might be one or more. He will execute them and document the resulting requirements from each technique at what is judged to be an appropriate level of detail. These requirements will be mostly information demand requirements, that is, requirements that describe why and how the stakeholders need specific information.

The business analyst will also elicit non-functional requirements, and for that he will be particularly concerned with the delivery mechanism (how will the results be physically made available to the end user? What tools will the user employ to view it?), the format (will the user view the results in reports, dashboards, or other formats?) and the degree of interaction needed (to what extent must the user be able to manipulate the results following delivery?).

A detailed analysis of the requirements will be done by the business analyst. The business analyst and the multiple stakeholders will negotiate to:

- Decide which requirements are accepted (which, in fact, is the same as deciding the system boundaries or scope)
- Do a triage and prioritization of the requirements
- Assess requirements risks

The business analyst will validate, that is, check for completeness and for consistency the resulting requirements. The triage and prioritization should be done after the validation, as the validation process might result in adding, changing or removing some requirements.

The business analyst will also need data, so he will ask the data analyst. Again, note that in a demand driven project this request will normally happen after the requirement elicitation, but in a supply driven project the data gathering that we will describe next will happen before the requirement elicitation.

The data analyst will look for the raw data to use for the project. The data might come from databases, internal or external, or from other sources. It might also need still to be collected for this specific purpose. The data analyst will need to select the data and decide if and when the data might need to be combined. If the data analyst considers the data to be too large for an initial analysis, he might consider using a sample of the data.

The data analyst will also try to understand the data. To begin with, if the data was already available at the beginning of the project, the data analyst should find the business motivation to collect and store the data in the first place, as it might provide some insights. From the data understanding he might suggest a possible hypotheses or objective to the business analyst. He might also identify constrains that arise from the data, so he will inform the business analyst of the detected constrains.

Since the raw data might be incomplete, noisy or inconsistent, the data engineer will perform data cleaning, pre-processing and transformation. This might include filling missing values, normalization, discretization, reduction, projection or other techniques. The data cleaning, pre-processing and transformation is guided by the data itself and also by what data mining techniques are going to be used on the data.

The data miner selects the tool to be used (for the same project, more than one tool might be used). For selecting the tool he will start by identifying possible tools and decide on how he will compare them, specifying the evaluation criteria that are important and how the evaluation will be performed (for instance, he might decide to run a specific algorithm using all the tools and a sample of the data). He will then proceed with the evaluation and choose the tool (or tools).

The data miner also selects the data mining technique (e.g., summarization, classification, regression, clustering) and the specific algorithms. For the same project, more than one tool might be used, as well as more than one data mining technique and one algorithm.

Some authors believe the choice of data mining technique can be simplified to four decisions, as seen in Figure 23 [61].

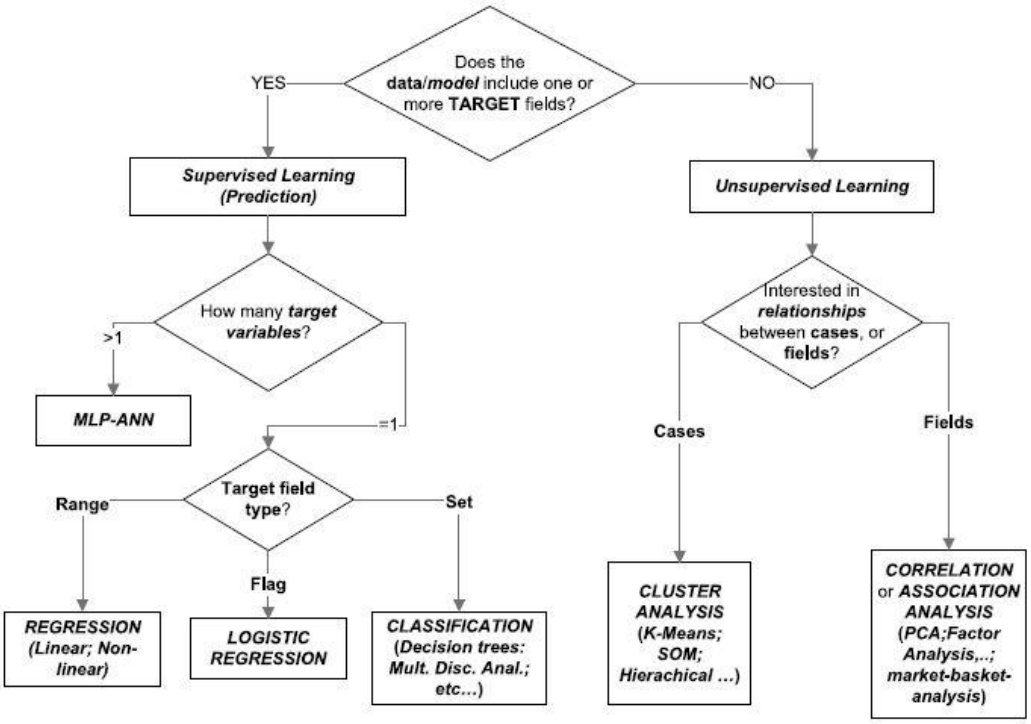


Figure 23 - A simple flowchart for data mining technique selection [61]

The data miner will entail the prepared data to the tool and be responsible for the generation of the model. This means he will have, for instance, to decide on the appropriate parameters.

After the actual data mining has occurred and the KD results are available, both the domain expert and the strategic manager will analyse the results.

The domain expert analyses the data mining result, in the sense that he evaluates how the results fit his domain knowledge, possibly resulting in the need for refining what was done previously through:

- creating new questions or hypothesis for the business analyst
- pointing the need for new or more data for the data analyst
- indicating the need to use a different data mining technique or algorithm or simply to adjust parameters to the data miner

The strategic manager interprets and evaluates the data mining result, in the sense that he evaluates how these results are relevant to or have an impact on the current or future business situation.

The knowledge engineer will use the analysis results from the domain expert and the strategic manager and make sure the discovered knowledge is used. He will specify how the knowledge discovery result should be deployed, for instance he can decide that an annual report should be produced for the senior management. The knowledge discovery result will then be deployed to the end users as planned.

5.3 DEMO specification

We used the Design and Engineering Methodology for Organizations (DEMO) [125] to specify the proposed generic KDP described in the previous section. For additional information regarding DEMO, please check Appendix III.

5.3.1 Analysis of the process description

We began by doing a Performa-Informa-Forma (PIF) analysis and a Coordination-Actors-Production (CAP) analysis.

The goal of these analyses is to gain insight to what concepts and activities are important in the KD process. Namely, in terms of activities, the Performa items are the truly relevant ones and will later be the transactions of the DEMO specification of SysPRE (sections 5.3.2 and 5.3.3).

The Performa-Informa-Forma analysis that follows was done by coloring the appropriate parts of the descriptions: red for Performa items, green for Informa items, and blue for Forma items. The Coordination-Actors-Production analysis was done simultaneously by enclosing a piece of text indicating an actor role between the brackets "[" and "]" and in yellow. We also chose to mark the transaction's id (for instance T01) next to Performa items. These transactions are presented in sections 5.3.2 and 5.3.3.

The knowledge discovery process begins {T01} when the [business analyst] realizes that there is a business problem or opportunity {T02} in which Knowledge Discovery and Data Mining might be helpful. More commonly, the [business analyst] starts with a question and needs certain information relevant to the decision he must make.

He or she starts by trying to learn {T03} as much as possible about the business and the application domain. He will identify the [stakeholders] {T04}. He will try to understand what issues are important for the [stakeholders] {T05}. The five core issues are [45]:

- Product (goods or services, tangible or intangible)
- Place
- Price
- Time
- Quantity

The [business analyst] will classify the knowledge discovery process as {T06}:

- **Demand driven**
  The process is aimed to fulfil the information requirements of the users
- **Data driven**
  The process is aimed to discover the best use to the specific existing data
- **Exploratory**
  The process is designed to find how KD and DM in general can offer value within that specific business

He will try to discover any relevant prior knowledge, namely the currently existing solutions for the problem, and identify the goal for the project {T05}.

If it is an exploratory process, the [business analyst] will identify several possible goals {T05} and review his stakeholders' identification {T04} for each one (including the core issues that each one might be concerned with {T05}).

Since starting the project might have costs, the [business analyst] might have to ask for approval {T14} for the data mining project to the [business manager]. The [business manager] might ask {T13} the [project manager] for a cost and resources estimation so that he can decide on the approval {T14}. The [project manager] will create the cost, time and resources estimates or a project plan {T13}, if necessary. The [project manager] will hand these to the [business manager]. The [business manager] will decide to go ahead or not {T14}, that is, he will decide on the feasibility of the KD project. If the decision is to go ahead, the [project manager] might have to get the resources (human or otherwise) that are necessary and that were not available in the beginning.

If it is a demand driven project, the [business analyst] will then begin eliciting specific requirements {T07}. If it is a data driven project, the [business analyst] will then proceed by

asking the [data analyst] to perform the data analysis. A hybrid approach is also possible, in which both will happen in parallel.

For the requirements elicitation {T07}, the [business analyst] will choose the elicitation techniques {T08} (list in Appendix I), which might be one or more. He will execute them and document the resulting requirements from each technique at what is judged to be an appropriate level of detail. These requirements will be mostly information demand requirements, that is, requirements that describe why and how the [stakeholders] need specific information.

The [business analyst] will also elicit non-functional requirements {T07}, and for that he will be particularly concerned with the delivery mechanism (how will the results be physically made available to the [end user]? What tools will the [user] employ to view it?), the format (will the [user] view the results in reports, dashboards, or other formats?) and the degree of interaction needed (to what extent must the [user] be able to manipulate the results following delivery?).

A detailed analysis of the requirements will be done by the [business analyst]. The [business analyst] and the multiple [stakeholders] will negotiate to:

- Decide which requirements are accepted {T09} (which, in fact, is the same as deciding the system boundaries or scope)
- Do a triage and prioritization of the requirements {T10}
- Assess requirements risks {T11}

The [business analyst] will validate {T12}, that is, check for completeness and for consistency the resulting requirements. The triage and prioritization {T10} should be done after the validation {T12}, as the validation {T12} process might result in adding, changing or removing some requirements.

The [business analyst] will also need data, so he will ask the [data analyst]. Again, note that in a demand driven project this request will normally happen after the requirement elicitation {T07}, but in a supply driven project the data gathering that we will describe next will happen before the requirement elicitation {T07}.

The [data analyst] will look for the raw data {T15} to use for the project. The data might come from databases, internal or external, or from other sources. It might also need still to be collected for this specific purpose. The [data analyst] will need to select the data {T16} and decide if and when the data might need to be combined {T17}. If the [data analyst] considers the data to be too large for an initial analysis, he might consider using a sample {T17} of the data.

The [data analyst] will also try to understand the data. To begin with, if the data was already available at the beginning of the project, the [data analyst] should find the business motivation to collect and store the data in the first place, as it might provide some insights. From the data understanding he might suggest a possible hypotheses or objective {T18} to the [business analyst]. He might also identify constrains {T19} that arise from the data, so he will inform the [business analyst] of the detected constrains.

Since the raw data might be incomplete, noisy or inconsistent, the [data engineer] will perform data cleaning, pre-processing and transformation {T17}. This might include filling missing values, normalization, discretization, reduction, projection or other techniques. The data cleaning, pre-processing and transformation is guided by the data itself and also by what data mining techniques are going to be used on the data.

The [data miner] selects the tool {T20} to be used (for the same project, more than one tool might be used). For selecting the tool he will start by identifying possible tools {T28} and decide on how he will compare them {T20}, specifying the evaluation criteria that are important and how the evaluation will be performed (for instance, he might decide to run a specific algorithm using all the tools and a sample of the data). He will then proceed with the evaluation and choose the tool {T20} (or tools).

The [data miner] also selects the data mining technique {T21} (e.g., summarization, classification, regression, clustering) and the specific algorithms {T22}. For the same project, more than one tool might be used, as well as more than one data mining technique and one algorithm.

Some authors believe the choice of data mining technique can be simplified to four decisions {T21}, as seen in Figure 23 [61].

The [data miner] will entail the prepared data to the tool and be responsible for the generation of the model {T24}. This means he will have, for instance, to decide on the appropriate parameters {T23}.

After the actual data mining has occurred and the KD results are available, both the [domain expert] and the [strategic manager] will analyse the results {T25}.

The [domain expert] analyses {T25} the data mining result, in the sense that he evaluates how the results fit his domain knowledge {T25}, possibly resulting in the need for refining what was done previously through:

- creating new questions or hypothesis {T18} for the [business analyst]
- pointing the need for new or more data {T15} for the [data analyst]
- indicating the need to use a different function {T21} or algorithm {T22} or simply to adjust parameters {T23} to the [data miner]

The [strategic manager] interprets and evaluates {T25} the data mining result, in the sense that he evaluates how these results are relevant to or have an impact {T25} on the current or future business situation.

The [knowledge engineer] will use the analysis results from the [domain expert] and the [strategic manager] and make sure the discovered knowledge is used. He will specify {T26} how the knowledge discovery result should be deployed, for instance he can decide that an annual report should be produced for the senior management. The knowledge discovery result will then be deployed to the [end users] as planned.

5.3.2 Transaction Result Table (TRT)

From the Performa-Informa-Forma analysis and Coordination-Actors-Production analysis that was done in section 5.3.1, we came up with the Transaction Result Table (TRT), shown in Table 2.

The table shows the transactions (that correspond to the main tasks of the process) and the result types corresponding to each transaction. In the result types, we can see (between square brackets) the main concept that is being created or whose state is being changed.

The last transactions (T28 to T31) refer to the specification of an elicitation technique for requirements or, regarding the data mining stage, the specification of a tool, data mining technique, algorithm or data mining parameter that was previously unknown to the system. This is necessary as the knowledge discovery and data mining area is very dynamic and it is very likely that new tools, data mining techniques, algorithms or data mining parameters need to be considered.

T27 is the transaction that manages all this. The elicitation techniques, tools, data mining techniques, algorithms and data mining parameters are referred to as artefacts in the context of T27 (KD area artefact management).

Transaction		Result type	
Id	Name	Id	Description
T01	Knowledge Discovery	R01	[knowledge discovery process] was realized
T02	Problem/Opportunity identification	R02	[problem/ opportunity] was identified
T03	Problem/Opportunity analysis	R03	[problem/ opportunity] was analysed
T04	Stakeholder identification	R04	[stakeholder] was identified
T05	Goal / core issue identification	R05	[goal / core issue] was identified
T06	Process classification	R06	[knowledge discovery process] was classified
T07	Requirement elicitation	R07	[requirement] was elicited
T08	Choice of elicitation technique	R08	[elicitation technique] was chosen
T09	Decision of scope	R09	decision on whether the [requirement] is in scope was made
T10	Requirement prioritization	R10	priority of [requirement] was defined
T11	Assessment of requirement risks	R11	risks of [requirement] were assessed
T12	Requirement validation	R12	[requirement] was validated
T13	Cost and resources estimation	R13	[cost and resources] were estimated
T14	Go-no-go Decision	R14	go-no-go decision of [knowledge discovery process] was made
T15	Data source identification	R15	[data source] was identified
T16	Data selection	R16	[data] was selected
T17	Data preparation	R17	[data] was prepared
T18	Hypothesis creation	R18	[hypothesis] was created
T19	Data constrain identification	R19	[data constrain] was identified
T20	Choice of tool	R20	tool was chosen for [result]
T21	Choice of data mining technique	R21	data mining technique was chosen for [result]
T22	Choice of algorithm	R22	algorithm was chosen for [result]
T23	Choice of data mining parameter	R23	data mining parameter was chosen for [result]
T24	Result obtention	R24	[result] was obtained
T25	Result analysis	R25	[result] was analysed
T26	Deployment specification	R26	[deployment] was specified
T27	KD area artefact management	R27	[KD area artefact] was managed
T28	Elicitation technique specification	R28	[elicitation technique] was specified
T29	Tool specification	R29	[tool] was specified
T30	Data mining technique specification	R30	[data mining technique] was specified
T31	Algorithm specification	R31	[algorithm] was specified
T32	Data mining parameter specification	T32	[data mining parameter] was specified

Table 2 - Transaction Result Table (TRT)

5.3.3 Actor Transaction Diagram (ATD)

From the Transaction Result Table that was presented in section 5.3.2, we came up with Actor Transaction Diagram (ATD), shown in Figure 24 and Figure 25. For additional information regarding DEMO and the ATD, please check Appendix III.

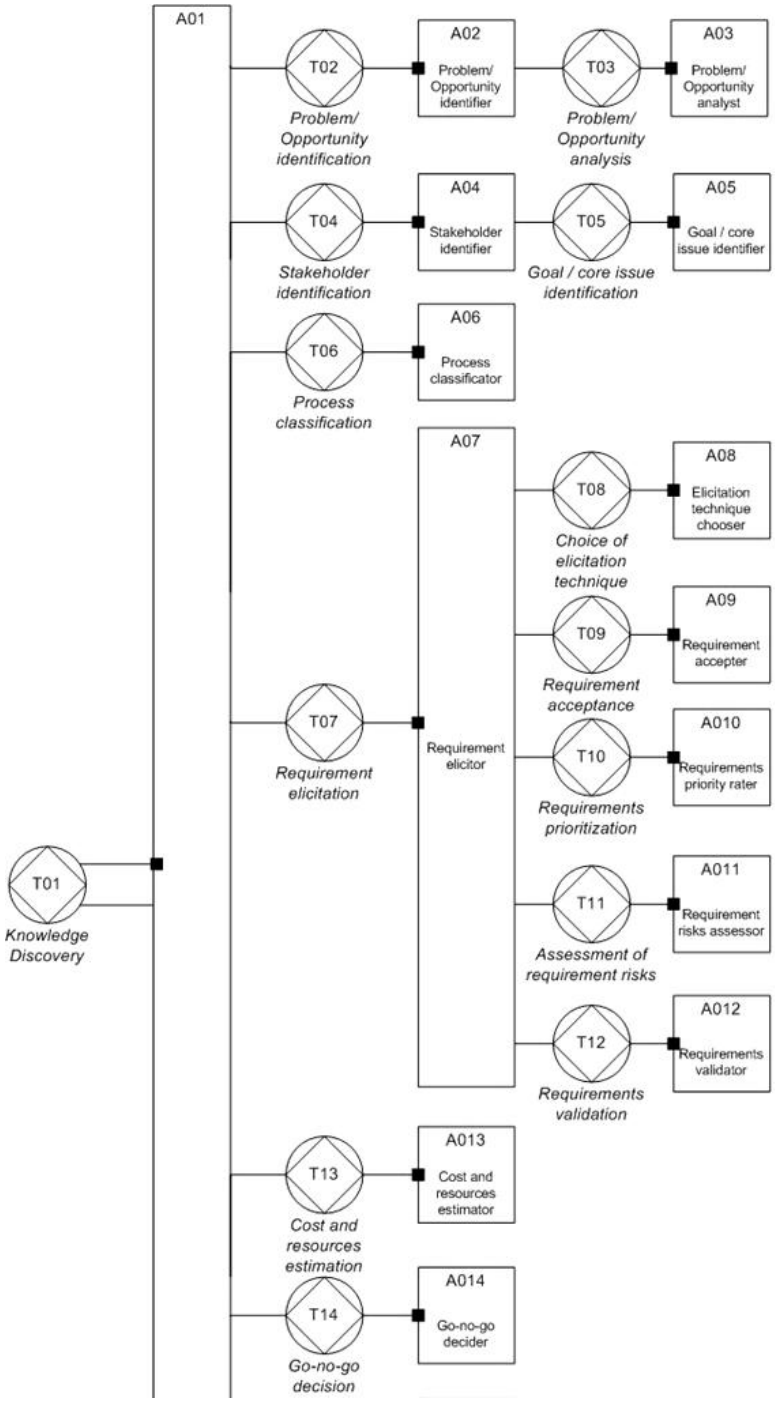


Figure 24 - Actor Transaction Diagram (ATD) - first part

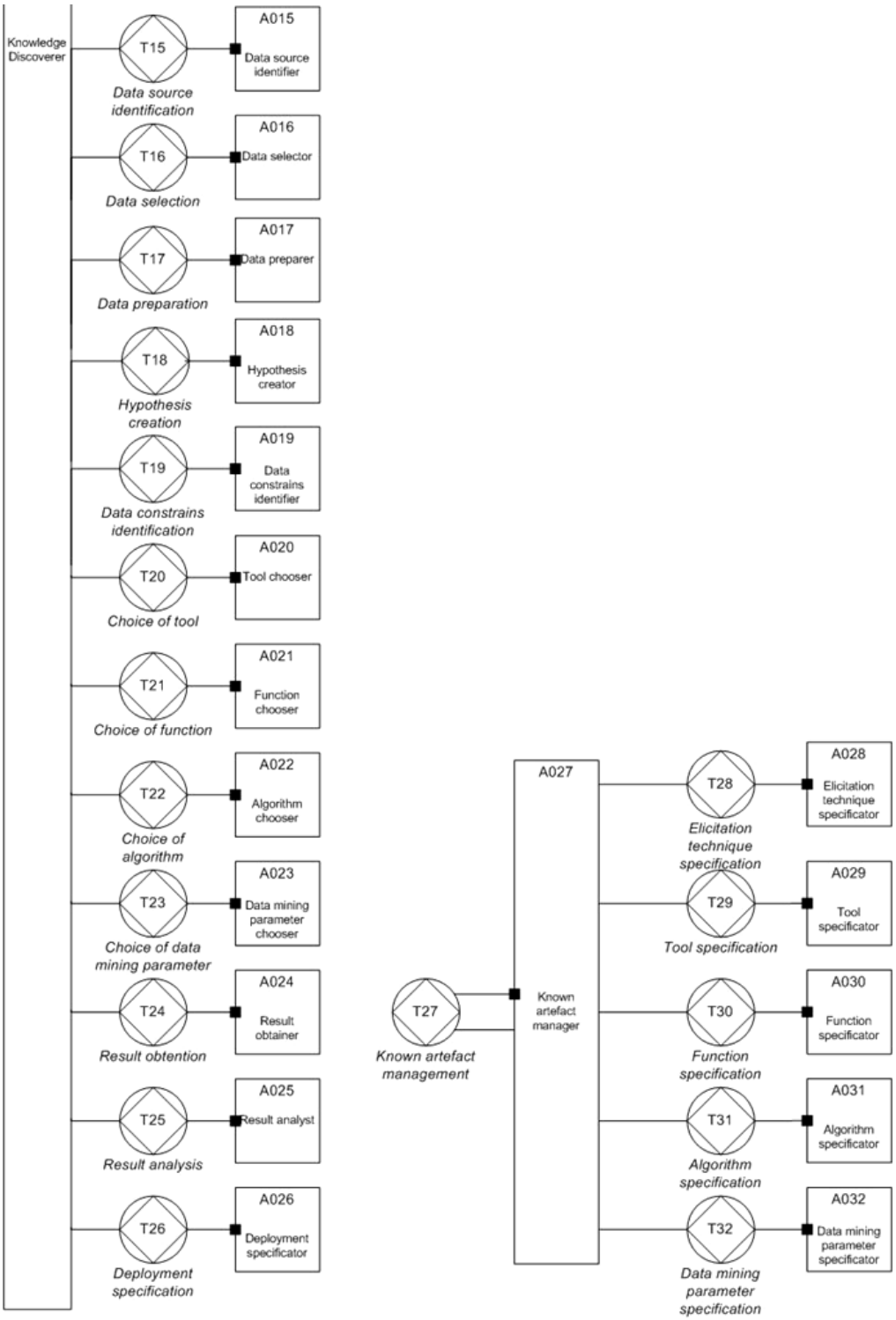


Figure 25 - Actor Transaction Diagram (ATD) - second part

The main transaction is T01 (Knowledge Discovery). The actor role A01 (Knowledge Discoverer) is both the initiator and the executor of T01. A01 is also the initiator of several transactions, that are, in a way, part of T01:

- T02 (Problem/Opportunity identification)
- T04 (Stakeholder identification)
- T05 (Goal / core issue identification)
- T06 (Process classification)
- T07 (Requirement elicitation)
- T13 (Cost and resources estimation)
- T14 (Go-no-go Decision)
- T15 (Data source identification)
- T16 (Data selection)
- T17 (Data preparation)
- T18 (Hypothesis creation)
- T19 (Data constrain identification)
- T20 (Choice of tool)
- T21 (Choice of data mining technique)
- T22 (Choice of algorithm)
- T23 (Choice of data mining parameter)
- T24 (Result obtention)
- T25 (Result analysis)
- T26 (Deployment specification)

The actor A02 (Problem / Opportunity identifier) is the executor of T02 (Problem / Opportunity identification) and the initiator of T03 (Problem / Opportunity analysis).

The actor A04 (Stakeholder identifier) is the executor of T04 (Stakeholder identification) and the initiator of T05 (Goal / core issue identification).

The actor A07 (Requirement elicitor) is the executor of T07 (Requirement elicitation) and the initiator of several transactions, that are, in a way, part of T07:

- T08 (Choice of elicitation technique)
- T09 (Decision of scope)
- T10 (Requirement prioritization)
- T11 (Assessment of requirement risks)
- T12 (Requirement validation)

There is a management transaction T27 (KD area artefact management). The actor role A27 (KD area artefact manager) is both the initiator and the executor of T27. A27 is also the initiator of several transactions that are part of T27:

- T28 (Elicitation technique specification)
- T29 (Tool specification)
- T30 (Data mining technique specification)
- T31 (Algorithm specification)
- T32 (Data mining parameter specification)

As mentioned previously, the elicitation techniques, tools, data mining techniques, algorithms and data mining parameters are referred to as artefacts in the context of T27 (KD area artefact management).

These transactions all refer to the specification of artefacts that were previously unknown to the system. As mentioned previously, this is necessary because it is very likely that new tools, data mining techniques, algorithms or data mining parameters need to be considered.

5.3.4 Object Fact Diagram (OFD)

We also specified DEMO's Object Fact Diagram (OFD), which uses WOSL (World Ontology Specification Language) [125] a notation similar to ORM (Object-Role Modeling) [126]. For additional information regarding DEMO and the OFD, please check Appendix III. The OFD can be found in Figure 26 and Figure 27.

In this diagram we can see the classes that correspond to the main concepts identified in the DEMO transactions of Table 2, as well as other related classes, the fact types that are associated with each class and the cardinalities and dependence laws.

In red comments we can see a possible instantiation of each class, so that the interpretation of the diagram is easier. The instantiation was done using data from the first case study described in chapter 6.

The main class of this OFD is the KNOWLEDGE DISCOVERY PROCESS (KDP), related to the main transaction T01. Each instance of this class will specify a particular KDP. Most of the classes that follow (in all caps text) are self-explanatory, so we will present them as we present the example.

Instances of the class PROBLEM/OPPORTUNITY specify a problem or an opportunity that triggered the KDP. Let's say that a company wants to increase its sales to existing customers. The company we are considering, from the case study presented in section 6.1, sells memberships, so basically they'll want to increase the percentage of customers that renew their memberships. This is the problem/opportunity.

One STAKEHOLDER is the Board of Directors. This particular stakeholder had a GOAL/CORE ISSUE: they want to increase the annual revenue. Using one or more ELICITATION TECHNIQUES, a REQUIREMENT to satisfy the above GOAL/CORE ISSUE was elicited: Predict how many customers will renew. One possible ELICITATION TECHNIQUE is a structured interview, but many others were possible (see Appendix I).

Normally several STAKEHOLDERS will be identified (T04), each with one or more GOAL/CORE ISSUE from which several REQUIREMENTS will stem and be elicited (T05).

From the accepted REQUIREMENTS, one will then proceed to create an HYPOTHESIS that can be tested in a KDP. For this case, one of the tested HYPOTHESIS was if the number of logins can be used to predict if a customer will renew. The link between HYPOTHESIS and REQUIREMENTS is important for traceability.

In the end, the RESULT of the KDP will either confirm this hypothesis or not. For the KDP, there needs to be an estimation of COST AND RESOURCES, so that a Go-no-go decision (T14) can take place.

If the KDP proceeds, instances of classes corresponding to the DATA SOURCE (from which DATA will be selected and prepared), the data mining TOOL (in this case, Tableau 8.1), the type of DATA MINING TECHNIQUE (in this case, classification) and the ALGORITHM (in this case, AdaBoost) will be used to obtain a particular RESULT. The ALGORITHM might require a DATA MINING PARAMETER (or more) to be set. In this case we could change the value for a_t weight, but did not.

The KD AREA ARTEFACT is a generalization that includes ELICITATION TECHNIQUE, TOOL, DATA MINING TECHNIQUE, ALGORITHM and DATA MINING PARAMETER. The management of these artefacts (T27) involves specifying an artefact that was previously unknown to the system whenever needed (T28, T29, T30, T31, T32). The can then be chosen for use (T08, T20, T21, T22, T23) using ELICITATION TECHNIQUE CHOICE CRITERIA, TOOL CHOICE CRITERIA, DATA MINING TECHNIQUE CHOICE CRITERIA, ALGORITHM CHOICE CRITERIA or DATA MINING PARAMETER CHOICE CRITERIA respectively. It is important that the choice criteria are all documented, which is why all these classes appear.

From the DATA might result some kind of DATA CONSTRAINT. In this case, it was very noticeable that the customer age was not available. The identified DATA CONSTRAINTS affected the KDP.

As mentioned, the execution of a particular algorithm with particular parameters and applied to a particular data, in the context of a KDP will produce a particular RESULT - for example, a classification model or a set of association rules. For the case study at hand, we found that the members who login more than once per month are more likely to renew.

The RESULT will be target of an analysis (T25). From such analysis the conclusion might be that new hypothesis needs to be formulated and/or new data, tools, data mining techniques or specific algorithms applied so that refined or alternative results are found. If none of this is necessary, the DEPLOYMENT of a RESULT can also be specified. For example, in this case it was decided that an annual report with the obtained result was to be produced.

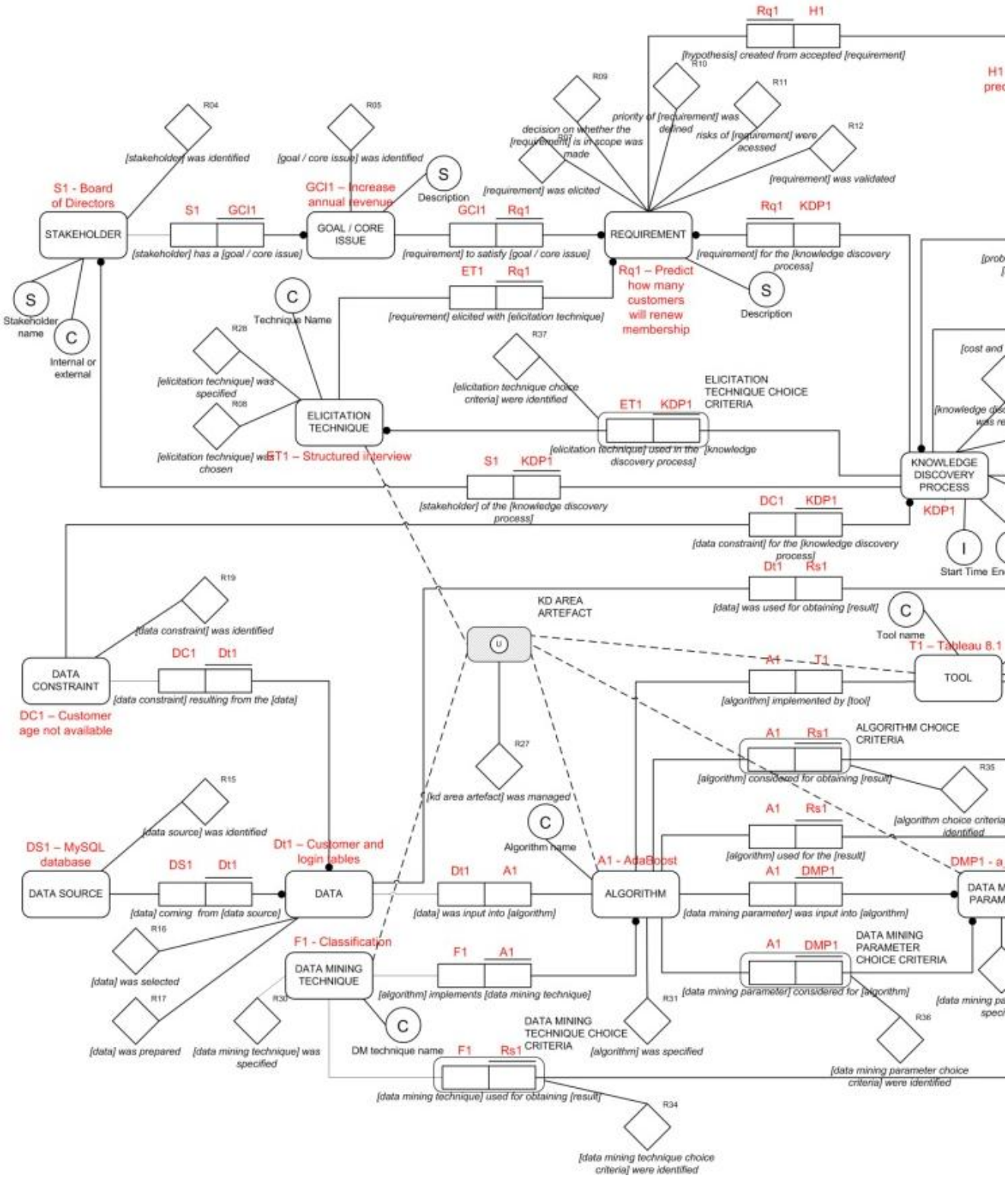


Figure 26 - Object Fact Diagram (OFD) – first part

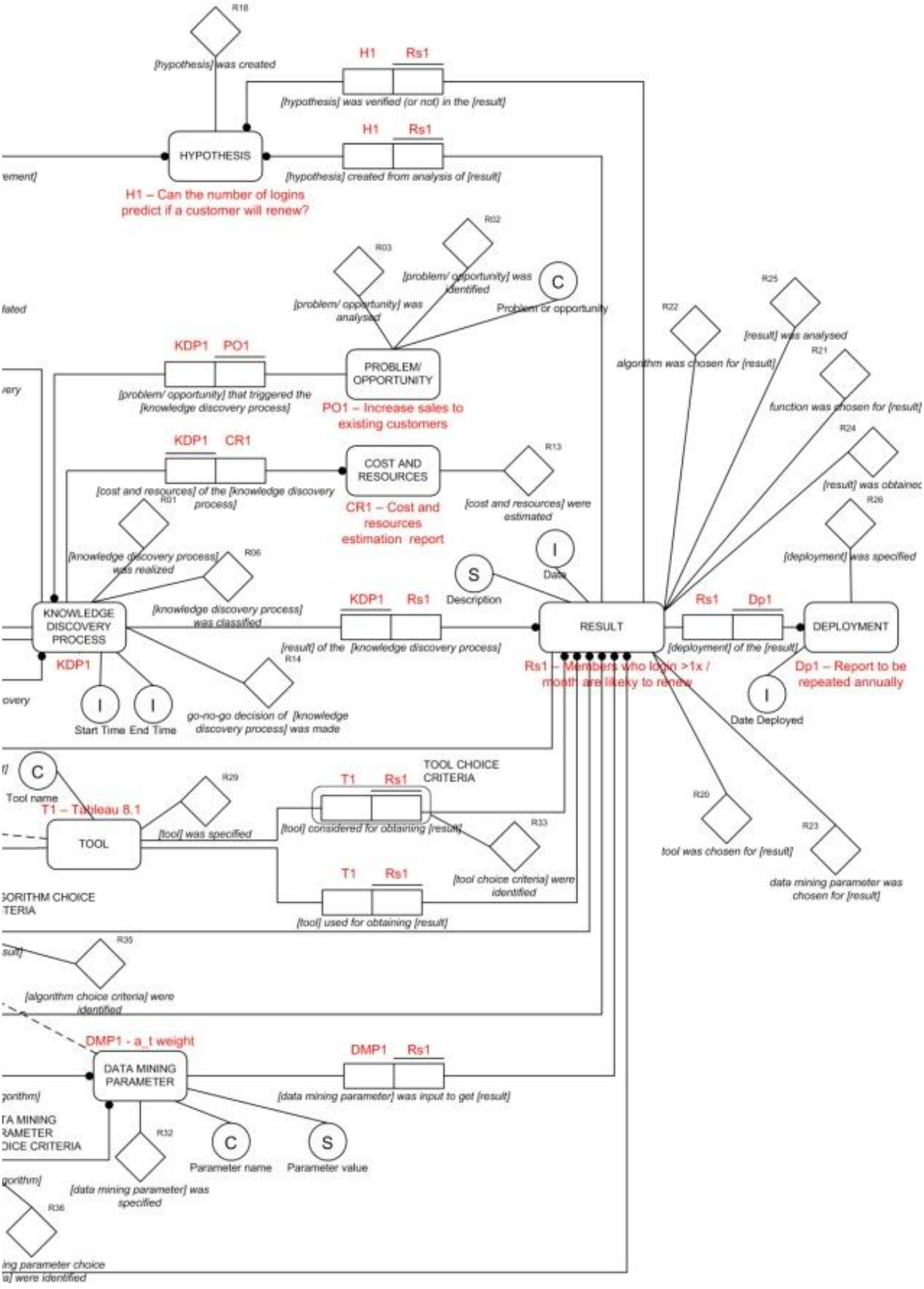


Figure 27 - Object Fact Diagram (OFD) – second part

5.3.5 Process Structure Diagram (PSD)

We also specified the Process Structure Diagram (PSD). For more information regarding DEMO and the PSD, please check Appendix III. Due to the diagram dimensions, it is presented divided into four parts (Figures 27 to 30).

The process starts when **T01 - Knowledge Discovery** is requested (T01 is the transaction that can be seen on top of Figure 28). When the executor promises to execute it, one or more **T02 - Problem/Opportunity identification** are requested. When each of them is finished and accepted by the initiator, a **T03 - Problem/Opportunity analysis** is requested as well as a **T06 - Process Classification** (in which the executor will classify the knowledge discovery process as Demand driven, Data driven or Exploratory, as described in the Process description, page 57).

When the executor promises to perform **T03 - Problem/Opportunity analysis**, one or more **T04 - Stakeholder identification** are requested. Acceptance by the initiator of **T06 - Process Classification** might also request **T04 - Stakeholder identification** or **T05 - Goal/core issue identification** for an already known stakeholder.

Upon acceptance by the initiator of **T04 - Stakeholder identification,** one or more **T05 - Goal/core issue identification** for the identified stakeholder are requested. **T03 - Problem/Opportunity analysis** execution cannot be considered finished until **T05 - Goal/core issue identification** is accepted by the initiator (this is represented by the dashed arrow, a wait-connection).

When **T03 - Problem/Opportunity analysis** is accepted by the initiator a **T13 - Cost and resources estimation** is requested. It is possible that more than one cost and resources estimation is requested, depending on the problem/opportunities identified and analysed.

Upon acceptance of **T13 - Cost and resources estimation** by the initiator, a **T14 - Go-no-go decision** is requested.

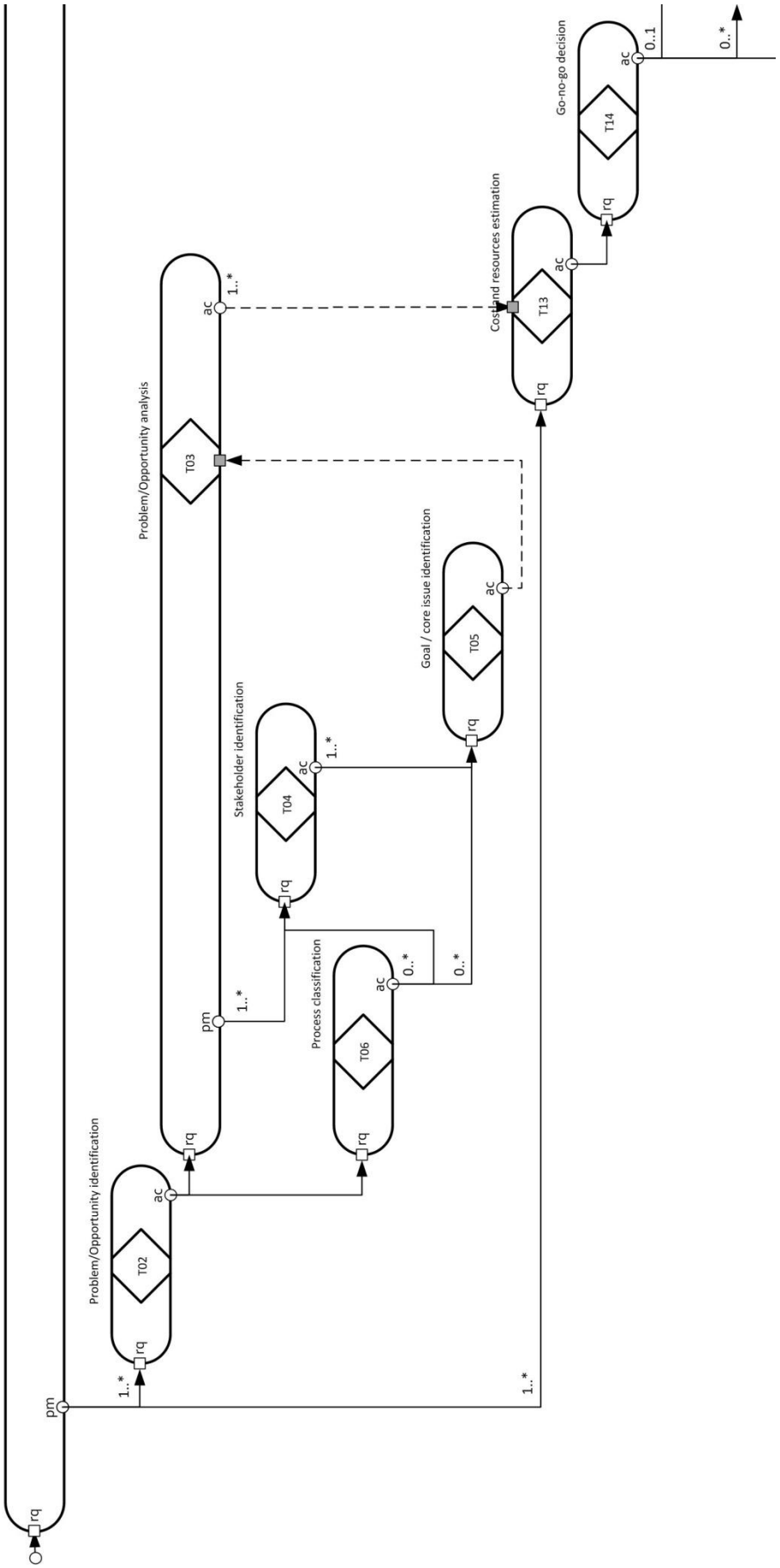


Figure 28 - Process Structure Diagram (PSD) - first part

With the accept of **T14 - Go-no-go decision**, the decision can either be to proceed or not. If the decision is to proceed, one or more of the following transactions are requested:

- **T07 - Requirement elicitation**
- **T15 - Data source identification**
- **T20 - Choice of tool**
- **T21 - Choice of data mining technique**

However, the accept can also correspond to a decision of not proceeding. In that case, the execution of **T01 – Knowledge Discovery Process** will finish (this is represented by the arrow that links T14-accept to T01-execution).

When the executor of **T07 - Requirement elicitation** promises to perform it, one or more **T08 - Choice of elicitation technique** are requested and a **T12 - Requirements validation** is also requested. Naturally, the execution of **T07 - Requirement elicitation** cannot finish execution before **T08 - Choice of elicitation technique** finishes at the acceptance by its initiator.

Acceptance of **T07 - Requirement elicitation** will lead to a request of **T11 - Assessment of requirement risks**. Acceptance of **T11 - Assessment of requirement risks** will, in turn, lead to a request of **T09 – Decision of scope**.

Having decided which requirements are accepted (which, in fact, is the same as deciding the system boundaries or scope and, in terms of transactions, is the same as having reached the accepted state in **T09 - Decision of scope**), we can now say that we have the full set of requirements. Therefore, **T12 - Requirements validation**, which will validate the set for consistency and completeness can now execute.

Upon acceptance of **T12 - Requirements validation**, a triage and prioritization of the requirements is needed, so a request of **T10 - Requirements prioritization** is done.

Acceptance of **T10 - Requirements prioritization** means that the whole requirements cycle is completed and from that one or more hypothesis for the KDP will be created. Therefore, a request for **T18 - Hypothesis creation** is issued.

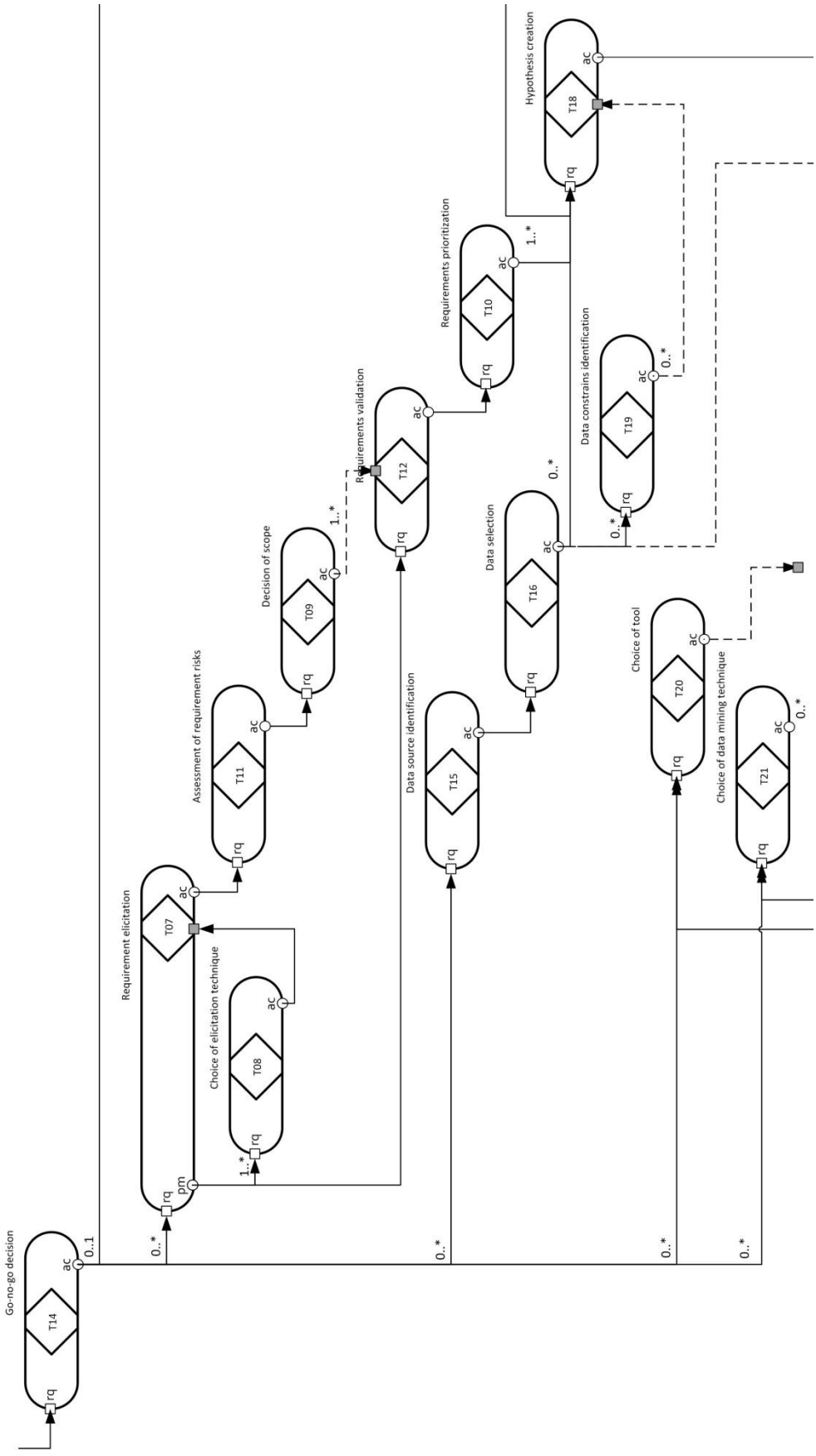


Figure 29 - Process Structure Diagram (PSD) - second part

Let's now go back to **T15 - Data source identification** which was also requested when **T14 - Go-no-go decision** was accepted. When it is accepted, **T16 - Data selection** is requested. This means that when a data source is identified, the actual data that will be used for the KDP must be selected from it.

When execution of **T16 - Data selection** is concluded and accepted, **T19 - Data constrains identification** may be requested. This means that once the data is selected, it is possible to identify aspects of that data that might be problematic in the sense that they might constrain what can be done in the KDP. This identification, however, is not mandatory, hence the cardinality 0..* in that link. **T19 - Data constrains identification** is important for the hypothesis creation, as there are hypothesis that might be unable to test at all due to the data constrains. That is why there is a wait-connection between the accept of **T19 - Data constrains identification** and the execution of **T18 - Hypothesis creation**.

When **T16 - Data selection** is accepted a request for **T18 - Hypothesis creation** might also be issued. This is because the data itself can be the source of an hypothesis to be tested.

It is also important to note that only when **T16 - Data selection** is accepted, can **T17 - Data preparation** start execution (as seen in Figure 30).

Let's now go back to **T20 - Choice of tool** and **T21 - Choice of data mining technique** which were also requested when **T14 - Go-no-go decision** was accepted. Acceptance of **T21 - Choice of data mining technique** means that the data mining technique to be used for the KDP, for instance, clustering or classification, has been chosen. Therefore, upon acceptance of **T21 - Choice of data mining technique**, any number of requests for **T22 - Choice of algorithm** are issued. This means that one or more algorithms might be chosen for the same KDP or the algorithm choice might actually be omitted because of the tool used. **T22 - Choice of algorithm** execution waits for **T20 - Choice of tool** acceptance, as the chosen algorithm must be implemented by the chosen tool.

Upon acceptance of **T22 - Choice of algorithm**, both **T23 - Choice of data mining parameter** and **T17 - Data preparation** are requested. Any number of data mining parameters might have their values set, which is represented by the 0..* cardinality on the link between T22-accept and T23-request. As noted before, **T17 - Data preparation** waits for **T16 - Data selection** acceptance to proceed.

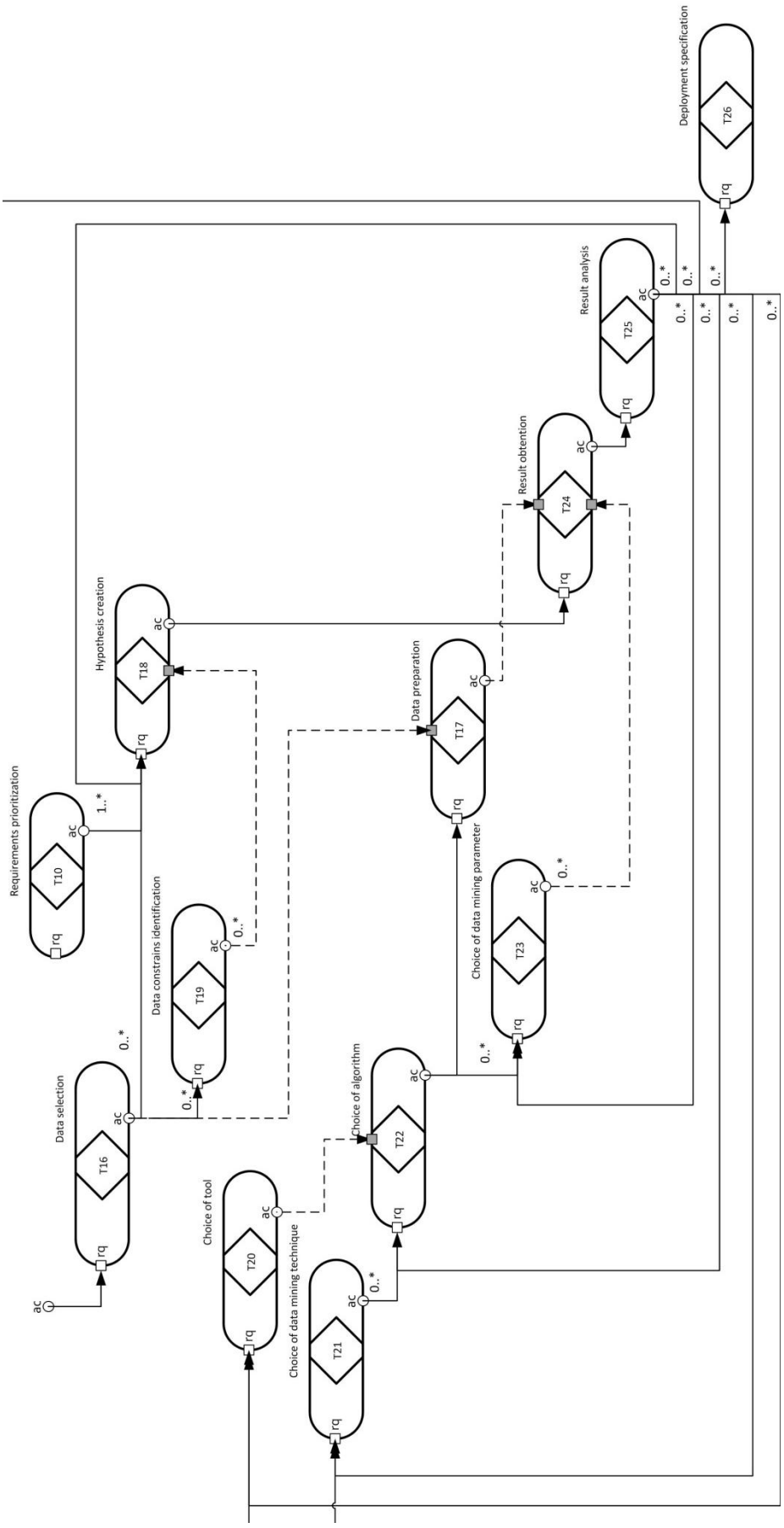


Figure 30- Process Structure Diagram (PSD) - third part

T24 - Result obtention is a key transaction, as it is in the execution phase of this transaction that the actual results of the KDP are obtained. This transaction is requested upon acceptance of **T18 - Hypothesis creation.** Its execution, however, needs to wait for **T17 - Data preparation** acceptance (which, in fact, means that data has been selected and prepared) and **T23 - Choice of data mining parameter** (which, in fact means, the tool, technique and/or algorithm and parameters have been chosen). In the case of parameters, there might not be any, so this is represented by the 0..* cardinality.

Acceptance of **T24 - Result obtention** causes **T25 - Result analysis** to be requested. The result analysis is rather complex, as the results are evaluated to see their relevance and impact on the current or future business situation, but also to see if they fit the domain knowledge. From this analysis, might stem the need to refine what was done previously. That is why the acceptance of **T25 - Result analysis** might result in a request for:

- **T18 - Hypothesis creation**, if the results suggest a new hypothesis should be tested
- **T20 - Choice of tool**, if the tool seemed to limit the result in any way
- **T21 - Choice of data mining technique**, for instance, if the results are non-conclusive using classification, maybe better results could be obtained with clustering
- **T22 - Choice of algorithm**, if a new algorithm is to be tried
- **T23 - Choice of data mining parameter**, if it is the parameters for the algorithm that need adjustment

If refining is not needed, then execution of **T01 - Knowledge discovery** can conclude and a request for **T26 - Deployment specification** is issued. This means that it will be decided how the knowledge discovery result should be deployed, for instance, the specification might determine that an annual report should be produced for the senior management.

Finally, there is also a management transaction **T27 - KD area artefact management** (Figure 31). A promise of it by its executor, might request:

- **T28 - Elicitation technique specification**
- **T29 - Tool specification**
- **T30 - Data mining technique specification**
- **T31 - Algorithm specification**
- **T32 - Data mining parameter specification**

These transactions all refer to the specification of artefacts that were previously unknown to the system. As mentioned previously, this is necessary because it is very likely that new tools, data mining techniques, algorithms or data mining parameters need to be considered.

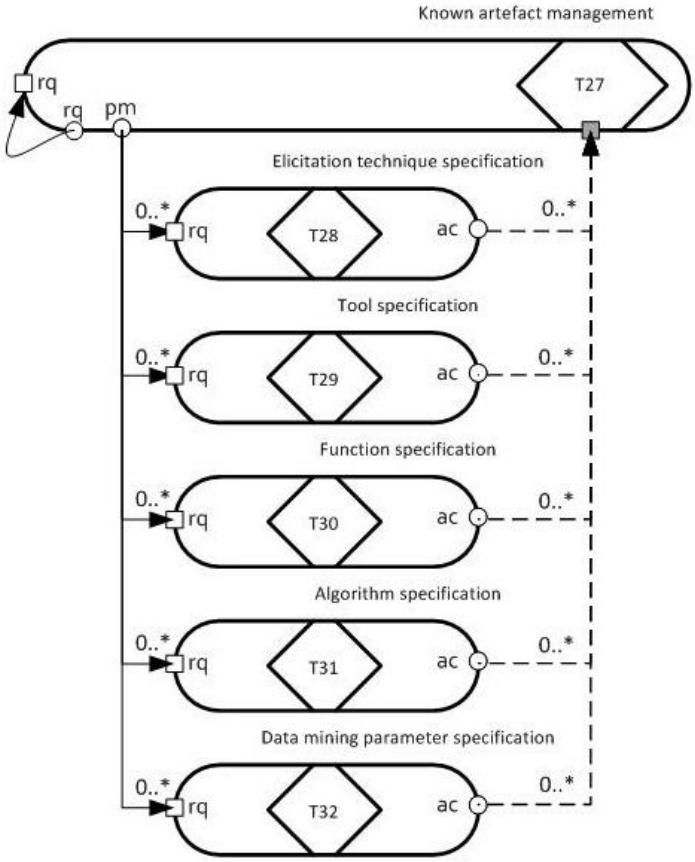


Figure 31- Process Structure Diagram (PSD) - fourth part

6 Case study validation

"Our greatest weakness lies in giving up. The most certain way to succeed is always to try just one more time. "

Thomas A. Edison

Having developed SysPRE into a DEMO ontology, we wanted to validate it. We first used as case study from the author's professional experience and then we used a published case study for validation.

6.1 Intervac

6.1.1 Description

The first case study is based on Intervac, an international organization that promotes home exchange for holidays.

Intervac was started in Europe in the early 1950's by a group of teachers with plenty of vacation time looking for economic means to travel internationally. Currently Intervac offers its services around the world and facilitates home exchanges for 30,000 members from all walks of life. Intervac has local organizers or agents in 45 countries and the umbrella organisation - Intervac International - is a registered non-profit organisation in Sweden [127].

For the case study a snapshot of their MySQL database from August 2013 was used with permission (a non-disclosure agreement was signed in June 2013 allowing the database to be used for this effect).

Statistical data taken from the database shows:

- 50.000 users
- 1,48 million messages exchanged
- 22.000 exchange agreements

The project this case study is based on was entirely exploratory, as defined in section 5.2: it was directed to find how KD and DM can offer value within that specific business.

The general goal of Intervac is to increase the number of members and to achieve this, it is necessary to:

- Have new members sign in
- Have existing members renew

The existing database was analyzed using regular queries and statistical analysis. After that the tools RapidMiner and Tableau 8.1 were used.

Figure 32 – Report from Tableau 8.1 regarding number of exchanges per member

6.1.2 Insights

From this case study we learned that:

- The approach to a KD or DM project can be quite different depending on the project classification. For this reason we created a specific transaction for it **T06 - Process classification**. This way, the process can be classified as demand-driven, data-driven or exploratory and then it will be clear on whether the focus should be on the stakeholders requirements, the data or on exploring potential problems to solved or opportunities to be taken.
- Data constrains are very relevant. In this specific case, data that could potentially bring interesting results was not registered in the database (such as the members' ages). This affects what hypothesis can be tested in a KDP. This was reflected both in the OFD and the PSD. Finding a data constrain, interestingly enough, can

be, in itself, of value to the organization, as it may motivate them to gather missing information, for instance. This was the case for Intervac, and now the registration form for new members is been reformulated to include the birth date and other informations.

- Considering a KDP generally linear, as most process models do, does not reflect reality. KDP involves a lot of trial-and-error and new hypothesis can stem from requirements, but also from the result analysis, the selected data or the data constrains.

6.2 Centrelink

6.2.1 Description

The case study was originally published in the book "Data Mining for Business Applications" [28] and was based on Centrelink, a statutory agency established to provide the Australian Government services under the umbrella of the Department of Human Services.

Centrelink is one of the largest data users in Australia: in 2005 it was distributing approximately $63 billion annually in social security payments to 6,4 million customers, making 9,98 million individual entitlement payments and recording 5,2 billion electronic customer transactions each year [127].

These statistics result not only from Australia's very large population, but also from a significant volume of customer data. The data used for this case study, as presented in [28], included demographic data, transactional data and time series data.

For reasons that we will present further on, customers on benefit payments or allowances sometimes get overpaid and these overpayments collectively lead to a considerable debt to Centrelink. Statistical data for the period 1 July 2004 to 30 June 2005 shows that [127]:

- Centrelink carried out 3,8 million entitlement reviews, which resulted in 525.247 payments being cancelled or reduced;
- Approximately $43 million a week was saved and debts totalling $390,6 million were raised as a result of this review activity
- There were 3.446 convictions for welfare fraud involving $41,2 million in debts.

Naturally, the Australian Government wants to reduce these customer debts. The basic reason these debts exist is that qualification for payment is assessed against a customer’s

personal circumstances and if all criteria are met, payment will continue until a change of circumstances disqualifies the customer from obtaining further benefit.

Customer debt may occur when changes of customer circumstances are not properly advised or processed to Centrelink. For example, if you are a caregiver entitled to an allowance, should the person you are caring for pass away without you informing Centrelink of the event, Centrelink may continue to pay the allowance until the event is notified or discovered through a random review process.

Furthermore, Centrelink's statistics show that 14 per cent of all entitlement reviews resulted in a customer debt. The remaining 86 per cent of reviews simply confirmed a correct situation. Identifying and reviewing those customers who display a high probability of having or acquiring a debt could, therefore:

- Save much effort
- Reduce customers debts to Centrelink

Resulting from this, predicting debtors based on demographic characteristics was the main requirement. To meet this requirement, decision tree and association rules algorithms were chosen, using, as data mining tool for both, the Teradata Warehouse Miner. The data used to tackle this problem was extracted from Centrelink's database for the period 1/7/2004 to 30/6/2005.

Using this case study's information the SysPRE OFD diagram was instanciated. The result is shown in Appendix IV, where the instantiation values are shown in green comments.

6.2.2 Insights

From this case study we learned that:

- Some tools, such as Teradata Warehouse Miner, completely omit the choice of algorithm. The data miner can choose the data mining technique and the parameters, but the algorithm is not available for choice.
- It is normal to use several data mining techniques (in this case, decision trees and association rules) for the same KDP. **T025 – Result analysis** can result in a new choice of data mining technique, tool, algorithm or parameter.
- For the trial-and-error process to be most successful, it is important to know all the details about what was previously tried, why and what the results were. For traceability sake, knowing what hypothesis were tested and if they resulted from a stakeholder requirement (indicating which one) or a data/result analysis is crucial.

7 Discussion

"If you can't describe what you are doing as a process, you don't know what you're doing. "

W. Edwards Deming

The proposed SysPRE formal ontology allows for a complete description of the KDP as a process and that alone has its merits. This was already partially discussed in section 2.2). In this chapter we discuss the merits of SysPRE by comparing it against the models that were analysed in the state-of-the-art review in section 2.3 (page 19 and onwards).

SysPRE has three main advantages over those models. The first advantage is that it is not a linear model. The models that were reviewed all resemble the waterfall model in the sense that they model the KDP as a sequence of steps, assuming the previous step is completed and then you move on to the next step.

In reality, there is a strong need for non-linearity:

- Steps can proceed in parallel
- Choices can be made between several possible next steps
- You might need to go back to a previous step

As an example of the first two points, we consider that you can identify your data sources first and then proceed to requirement elicitation, or you can start by eliciting the requirements from the stakeholders, or you can do both in parallel. It will depend on the project at hand, on whether you previously classified it as demand-driven, data-driven or exploratory.

As an example for the final point, the most striking example is the result analysis that is done once you have the results. This very frequently will result in the need to review or adjust choices that you made previously (in terms of tool, data mining technique, algorithm, parameters or data). We believe it is as likely to get all these right the first time, as it is to write a piece of software code with no bugs the first time round.

The second advantage is the ontological view taken for SysPRE. This ontological view means the focus is on the decisions that the persons involved need to take and what communication steps need to occur. As result of this ontological view, a go-no-go decision was considered, which none of the previous models did. All activities that lead up to this decision were also considered, and those were also either disregarded or bundled up in one single step.

The major final advantage is the focus on the requirements step. The studied models either ignored the requirements step altogether or treated the requirements as one single step that needs to be done before proceeding to the actual data mining. The most recent model studied focused on the business understanding step. In SysPRE we also take into consideration business concerns (such as the cost and resources estimation) but focus with a higher degree of detail into the requirements. The requirements step is broken into 6 transactions, covering elicitation, risk assessment, acceptance and prioritization. The relation between these requirements' transactions and the other transactions is also considered, and in a way that we believe to be closer to what happens in reality.

Having the SysPRE formal ontology can also be helpful for the communication between the organization and other stakeholders because, despite being formal, they are understandable and do sum up a lot of information in a graphical way. We therefore believe they can:

- Help bridge the gap that normally exists between the organization that is carrying out the KD process and other stakeholders.
- Help each technical role involved keep an eye on the big picture while working on whatever task they are working on at that specific moment.
- Make enterprises (and specifically decision makers within the enterprise) become aware of their own KD process and RE process in the KD projects.
- Assist enterprises that want to improve their own KD process and RE process in the KD projects.

We see SysPRE as the foundation for building a support tool for those involved in KD projects. We believe such tool could enhance all the above stated goals and also enable keeping a record of iterations and refinements of a particular process in a highly structured way. This is a crucial aspect for KD projects, as success often depends on a trial-and-error cycle that could be much improved by recording what has been done in this way.

8 Conclusion

"From a small seed a mighty trunk may grow."
Aeschylus

Knowledge Discovery and Data Mining is an area that has grown a lot during the past few years.

Since 1989 the need for a KDP model has been felt and that has resulted in a multitude of models that have been proposed from 1996 onwards. However, these models tend to oversimplify reality and present a simple sequence of steps and the details for these steps are presented in an informal way, subject to different interpretations by different people.

In order to systematize all these proposals, we first did a thorough literature review to collect as much information as possible of the recommended steps and needed info for the KDP. We then focused our attention in the Requirements Engineering area and again did a literature review there. We then looked at how these two areas work together, focusing first on why requirements engineering is different when done in a KD project and then on the state-of-the-art in terms of existing process models for requirements engineering in KD projects.

We then picked all the information found about the KDP in general models and aggregated them in a proposal of a more thorough and generic KDP than the ones we had found in the review. Then, by applying DEMO, we managed to identify in a much more comprehensive way the tasks of the KDP, as well as a set of the main concepts or classes of information that should be kept during a KDP.

This DEMO based ontology gave us several interesting insights. Thanks to the specified classes, for a particular problem/opportunity we can keep a record of detailed and important information of a respective KDP. Namely we can keep a consistent and integrated record of important business side information like the stakeholders, requirements, hypothesis and costs; and also of the technical side like tools, sources and algorithms used. The class RESULT is pivotal in the sense that each instance will include not only the patterns obtained using the data mining technique, but also an analysis of the results which may lead to the formulation of new hypothesis and requirements on the business side.

The research question that we investigated during this Master thesis was **"How can systematization be brought into Knowledge Discovery projects, in general, and into their**

Requirement Engineering phase, in particular, aiming at improvements in their success rate?". We believe that having SysPRE, an ontology that represents both the KD work in general and the RE for KD work specifically can help technical roles not lose track of the big picture while working on the task at hand. Also, since it is understandable not only by the technical roles involved, but also by other stakeholders, SysPRE can foster a more effective dialogue between them.

We believe this ontology can encourage knowledge reuse of the KD process or RE KD process itself in a consistent and integrated fashion because it enables keeping a record of iterations and refinements of a particular process in a highly structured way. This way, we hope to make enterprises become aware of their own KD process and RE process in the KD projects, but also to improve such processes in reality, namely in terms of the success rate. In other words, we hope this can help the lessons learned from the past be reused to improve the present.

8.1 Contributions

The main contribution of this thesis is to provide a systematization that can be applied to KD projects in general and to the requirements engineering process in such processes in particular.

In a more detailed way we believe the contributions of this thesis are:

- Having a short, plain text description of a generic KD process with emphasis on RE that was proposed after doing a thorough literature review can be useful for novices in the area, both in the research and in the industry communities. As we mentioned before, as the KD area grows, there is a growing number of non-specialists entering the area, so we believe this can be a good starting point.
- Having the SysPRE formal ontology can be helpful within the organization using them because it can:
    - Enable keeping a record of iterations and refinements of a particular process in a highly structured way.
    - Make enterprises (and specifically decision makers within the enterprise) become aware of their own KD process and RE process in the KD projects.
    - Assist enterprises that want to improve their own KD process and RE process in the KD projects.
    - Help each technical role involved keep an eye on the big picture while working on whatever task they are working on at that specific moment.

- Having the SysPRE formal ontology can also be helpful for the communication between the organization and other stakeholders because, despite being formal, they are understandable and do sum up a lot of information in a graphical way. We therefore believe they can help bridge the gap that normally exists between the organization that is carrying out the KD process and other stakeholders.

8.2 Future Work

Our SysPRE ontology and specifications need further validation in practice. For this, as future work, we would consider:

1. Using more case studies from literature to more extensively populated SysPRE with instances and eventually gain more insights
2. Using it from the beginning in one practical real-world case study
3. Building a software that would implement the complete process model would allow for a large scale validation in practice

Another possible next step in terms of future work is to expand the ontology to express other areas of KD process as we have for the RE area.

Once the implementation step is done, it would be possible to verify if the success rates really are improved by using SysPRE.

We strongly believe that implementation would also allow the true value of SysPRE to be apparent outside the academic world, and adoption by businesses and organizations could be widespread. There is a “vacant spot” left by the dominant process model, CRISP-DM, that is no longer updated by its proponents, and there are real advantages for using a process model as complete as SysPRE when it is implemented.

9 References

[1] The Standish Group, "1994 CHAOS Report," 1994.

[2] R. L. Glass, "IT Failure Rates-70% or 10-15%?," *Softw. IEEE*, vol. 22, no. 3, pp. 112–110, 2005.

[3] M. Jørgensen and K. Moløkken-Østvold, "How large are software cost overruns? A review of the 1994 CHAOS report," *Inf. Softw. Technol.*, vol. 48, no. 4, pp. 297–301, Apr. 2006.

[4] R. L. Glass, "The Standish report: does it really describe a software crisis?," *Commun. ACM*, vol. 49, no. 8, pp. 15–16, 2006.

[5] J. Eveleens and C. Verhoef, "The Rise and Fall of the Chaos Report Figures," *IEEE Software*, 2010.

[6] K. Pohl, *Requirements Engineering: Fundamentals, Principles, and Techniques*. Springer Berlin Heidelberg, 2010.

[7] K. El Emam and A. G. Koru, "A replicated survey of IT software project failures," *Softw. IEEE*, vol. 25, no. 5, pp. 84–90, 2008.

[8] C. Atkins, "An Investigation of the Impact of Requirements Engineering Skills on Project Success," East Tennessee State University, 2013.

[9] A. Paiva, J. Varajão, and C. Dominguez, "Principais aspectos na avaliação do sucesso de projectos de desenvolvimento de software. Há alguma relação com o que é considerado noutras indústrias?," *Interciencia*, vol. 36, no. 3, pp. 200–204, 2011.

[10] J. Wateridge, "How can IS/IT projects be measured for success?," *Int. J. Proj. Manag.*, vol. 16, no. 1, pp. 59–63, Feb. 1998.

[11] U. M. Fayyad, G. Piatetsky-Shapiro, and P. Smyth, "Knowledge Discovery and Data Mining: Towards a Unifying Framework.," in *KDD*, 1996, vol. 96, pp. 82–88.

[12] W. W. Royce, "Managing the development of large software systems," in *proceedings of IEEE WESCON*, 1970, vol. 26.

[13] "Statistics - YouTube." [Online]. Available: https://www.youtube.com/yt/press/statistics.html. [Accessed: 29-Oct-2014].

[14] S. Radicati, Ed., "Email Statistics Report 2013-2017 Executive Summary." Apr-2013.

[15] J. Manyika, M. Chui, B. Brown, and J. Bughin, "Big data: The next frontier for innovation, competition, and productivity | McKinsey & Company," May-2011. [Online]. Available: http://www.mckinsey.com/insights/business_technology/big_data_the_next_frontier_for_innovation. [Accessed: 20-May-2014].

[16] M. Traverso, "Presto: Interacting with petabytes of data at Facebook," *Research at Facebook*, Nov-2013. [Online]. Available: https://research.facebook.com/blog/1489667567986457/presto-interacting-with-petabytes-of-data-at-facebook/. [Accessed: 30-Oct-2014].

[17] P. Pytel, P. Britos, and R. García-Martínez, "A Proposal of Effort Estimation Method for Information Mining Projects Oriented to SMEs," in *Enterprise Information Systems of the Future*, Springer, 2013, pp. 58–74.

[18] W. H. Inmon, *Building the Data Warehouse*. John Wiley & Sons, 2005.

[19] G. Piatetsky-Shapiro, "Knowledge discovery in databases: 10 years after," *ACM SIGKDD Explor. Newsl.*, vol. 1, no. 2, pp. 59–61, 2000.

[20] U. Fayyad, G. Piatetsky-Shapiro, and P. Smyth, "From data mining to knowledge discovery in databases," *AI Mag.*, vol. 17, no. 3, p. 37, 1996.

[21] U. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy, *Advances in Knowledge Discovery and Data Mining*. The MIT Press, 1996.
[22] L. A. Kurgan and P. Musilek, "A survey of Knowledge Discovery and Data Mining process models," *Knowl. Eng. Rev.*, vol. 21, no. 01, p. 1, Mar. 2006.
[23] A. Bernstein, F. Provost, and S. Hill, "Toward intelligent assistance for a data mining process: an ontology-based approach for cost-sensitive classification," *IEEE Trans. Knowl. Data Eng.*, vol. 17, no. 4, pp. 503–518, Apr. 2005.
[24] G. Piatetsky-Shapiro, "Knowledge discovery in real databases: A report on the IJCAI-89 Workshop," *AI Mag.*, vol. 11, no. 4, p. 68, 1990.
[25] R. Pressman and B. Maxim, *Software Engineering: A Practitioner's Approach*, 8 edition. New York, NY: McGraw-Hill Science/Engineering/Math, 2014.
[26] K. J. Cios, *Data mining a knowledge discovery approach*. New York; New York: Springer, 2007.
[27] L. Cao, "Domain-driven, actionable knowledge discovery," *IEEE Intell. Syst.*, pp. 78–79, 2007.
[28] L. Cao, P. S. Yu, C. Zhang, and H. Zhang, Eds., *Data Mining for Business Applications*. Boston, MA: Springer US, 2009.
[29] M. Ganesh, E. H. Han, V. Kumar, S. Shekhar, and J. Srivastava, "Visual data mining: Framework and algorithm development," *Dep. Comput. Inf. Sci. Univ. Minn. MN USA*, 1996.
[30] P. Adriaans and D. Zantinge, *Data mining*. Addison-Wesley, 1996.
[31] R. J. Brachman and T. Anand, "Advances in Knowledge Discovery and Data Mining," U. M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy, Eds. Menlo Park, CA, USA: American Association for Artificial Intelligence, 1996, pp. 37–57.
[32] M. J. Berry and G. Linoff, *Data mining techniques: for marketing, sales, and customer support*. John Wiley & Sons, Inc., 1997.
[33] K. Collier, B. Carey, E. Grusy, C. Marjaniemi, and D. Sautter, "A Perspective on data Mining," *Cent. Data Insight North. Ariz. Univ. USA*, pp. 2–4, 1998.
[34] P. Cabena, Hadjnian, Stadler, Verhees, and Zanasi, *Discovering Data Mining: From Concept to Implementation*. Upper Saddle River, N.J: Prentice Hall, 1997.
[35] S. W. Lee and L. Kerschberg, "A methodology and life cycle model for data mining and knowledge discovery in precision agriculture," in *Systems, Man, and Cybernetics, 1998. 1998 IEEE International Conference on*, 1998, vol. 3, pp. 2882–2887.
[36] A. Feelders, H. Daniels, and M. Holsheimer, "Methodological and practical aspects of data mining," *Inf. Manage.*, pp. 271–281, 1998.
[37] A. G. Buchner, M. D. Mulvenna, S. S. Anand, and J. G. Hughes, "An internet-enabled knowledge discovery process," in *Proceedings of the 9th international database conference, Hong Kong*, 1999, vol. 1999, pp. 13–27.
[38] S. S. Anand and A. G. Büchner, *Decision Support Using Data Mining*. Financial Times Management, 1998.
[39] R. Wirth and J. Hipp, "CRISP-DM: Towards a standard process model for data mining," in *Proceedings of the 4th International Conference on the Practical Applications of Knowledge Discovery and Data Mining*, 2000, pp. 29–39.
[40] KDNuggets, "Poll: Data Mining Methodology," Aug-2007. [Online]. Available: http://www.kdnuggets.com/polls/2007/data_mining_methodology.htm. [Accessed: 10-Jun-2014].
[41] P. Chapman, J. Clinton, R. Kerber, T. Khabaza, T. Reinartz, C. Shearer, and R. Wirth, "CRISP-DM 1.0 Step-by-step data mining guide.pdf," *SPSS*, 2000.

[42] SAS Institute, "SEMMA," 2005. [Online]. Available: http://www.sas.com/offices/europe/uk/technologies/analytics/datamining/miner/semma.html. [Accessed: 10-Jun-2014].
[43] J. M. Moine, "Metodologías para el descubrimiento de conocimiento en bases de datos: un estudio comparativo," Facultad de Informática, 2013.
[44] A. I. R. L. Azevedo and M. F. Santos, "KDD, SEMMA and CRISP-DM: a parallel overview," presented at the IADIS European Conf. Data Mining, 2008, pp. 182–185.
[45] D. Pyle, *Business Modeling and Data Mining*. Morgan Kaufmann, 2003.
[46] G. Mariscal, Ó. Marbán, and C. Fernández, "A survey of data mining and knowledge discovery process models and methodologies," *Knowl. Eng. Rev.*, vol. 25, no. 02, pp. 137–166, Jun. 2010.
[47] L. A. Kurgan, K. J. Cios, R. Tadeusiewicz, M. Ogiela, and L. S. Goodenday, "Knowledge discovery approach to automated cardiac SPECT diagnosis," *Artif. Intell. Med.*, vol. 23, no. 2, pp. 149–169, 2001.
[48] J. Han and M. Kamber, *Data Mining: Concepts and Techniques*. Morgan Kaufmann, 2001.
[49] H. Edelstein, "Building profitable customer relationships with data mining," in *Customer Relationship Management*, Vieweg+Teubner Verlag, 2001, pp. 339–351.
[50] S. Moyle and A. Jorge, "RAMSYS-A methodology for supporting rapid remote collaborative data mining projects," in *ECML/PKDD01 Workshop: Integrating Aspects of Data Mining, Decision Support and Meta-learning (IDDM-2001)*, 2001.
[51] H. Blockeel and S. Moyle, "Centralized model evaluation for collaborative data mining," in *Conference on Data Mining and Warehouses (SiKDD 2002)*, 2002.
[52] J. Solarte, "A Proposed Data Mining Methodology and its Application to Industrial Engineering," *Masters Theses*, Aug. 2002.
[53] W. Klösgen and J. M. Zytkow, Eds., *Handbook of Data Mining and Knowledge Discovery*. New York, NY, USA: Oxford University Press, Inc., 2002.
[54] D. J. Haglin, R. J. Roiger, J. Hakkila, and T. W. Giblin, "A tool for public analysis of scientific data.," *Data Sci. J.*, vol. 4, no. 30, pp. 39–53, 2005.
[55] K. J. Cios and L. A. Kurgan, "Trends in data mining and knowledge discovery," in *Advanced techniques in knowledge discovery and data mining*, Springer, 2005, pp. 1–26.
[56] Ó. Marbán, G. Mariscal, E. Menasalvas, and J. Segovia, "An Engineering Approach to Data Mining Projects," in *Proceedings of the 8th International Conference on Intelligent Data Engineering and Automated Learning*, Berlin, Heidelberg, 2007, pp. 578–588.
[57] O. Marbán, J. Segovia, E. Menasalvas, and C. Fernández-Baizán, "Toward data mining engineering: A software engineering approach," *Inf. Syst.*, vol. 34, no. 1, pp. 87–107, Mar. 2009.
[58] IEEE Computer Society, "IEEE Standard for Developing Software Life Cycle Processes," *IEEE Std 1074-1995*, 1996.
[59] International Organization for Standardization, "ISO/IEC Standard 12207:1995. Software Life Cycle Processes," 1995.
[60] P. Gottgtroy, "Ontology driven knowledge discovery process: a proposal to integrate ontology engineering and KDD," 2007.
[61] K. Rennolls and A. Al-Shawabkeh, "Formal structures for data mining, knowledge discovery and communication in a knowledge management environment," *Intell. Data Anal.*, vol. 12, no. 2, pp. 147–163, 2008.

[62] M. Hofmann and B. Tierney, “An Enhanced Generic Data Mining Life Cycle,” *ITB J.*, p. 50, 2009.
[63] W. E. Deming, *Out of the Crisis*. MIT Press, 2000.
[64] M. Alnoukari, Z. Alzoabi, and S. Hanna, “Applying adaptive software development (ASD) agile modeling on predictive data mining applications: ASD-DM Methodology,” in *Information Technology, 2008. ITSim 2008. International Symposium on*, 2008, vol. 2, pp. 1–6.
[65] M. Alnoukari and A. El Sheikh, “Knowledge Discovery Process Models,” 2012.
[66] Kweku-Muata Osei-Bryson, “A context-aware data mining process model based framework for supporting evaluation of data mining results,” *Expert Syst. Appl.*, vol. 39, no. 1, pp. 1156–1164, Jan. 2012.
[67] M. J. Harry, *The vision of six sigma: Tools and methods for breakthrough*. Sigma Pub. Co., 1994.
[68] IEEE Computer Society, “IEEE Standard Glossary of Software Engineering Terminology,” *IEEE Std 61012-1990*, pp. 1–84, Dec. 1990.
[69] J. O. Grady, *System Requirements Analysis*. Elsevier, 2013.
[70] W. Swartout and R. Balzer, “On the Inevitable Intertwining of Specification and Implementation,” *Commun ACM*, vol. 25, no. 7, pp. 438–440, Jul. 1982.
[71] R. B. Grady, *Practical Software Metrics for Project Management and Process Improvement*. Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 1992.
[72] I. Sommerville and P. Sawyer, *Requirements Engineering: A Good Practice Guide*, 1st ed. New York, NY, USA: John Wiley & Sons, Inc., 1997.
[73] P. A. Laplante, *Requirements Engineering for Software and Systems, Second Edition*. CRC Press, 2013.
[74] B. Nuseibeh and S. Easterbrook, “Requirements engineering: a roadmap,” in *Proceedings of the Conference on the Future of Software Engineering*, 2000, pp. 35–46.
[75] D. Carrizo, O. Dieste, and N. Juristo, “Systematizing requirements elicitation technique selection,” *Inf. Softw. Technol.*, Jan. 2014.
[76] I. Sommerville and G. Kotonya, *Requirements Engineering: Processes and Techniques*. New York, NY, USA: John Wiley & Sons, Inc., 1998.
[77] F. F. Tsui, *Essentials of Software Engineering*. Jones & Bartlett Publishers, 2013.
[78] T. DeMarco, “Bells, Whistles, Power, and the Requirements Process,” *IEEE Softw.*, vol. 30, no. 4, pp. 104–104, 2013.
[79] A. H. Dogru, *Modern Software Engineering Concepts and Practices: Advanced Approaches*. IGI Global, 2010.
[80] Y. Wang, *Software Engineering Foundations: A Software Science Perspective*. CRC Press, 2007.
[81] R. Y. Lee, *Software Engineering: A Hands-On Approach*. Springer Science & Business Media, 2013.
[82] A. Taylor, “IT projects: sink or swim,” *Comput. Bull.*, vol. 42, no. 1, pp. 24–26, Jan. 2000.
[83] B. Boehm, “A Spiral Model of Software Development and Enhancement,” *SIGSOFT Softw Eng Notes*, vol. 11, no. 4, pp. 14–24, Aug. 1986.
[84] O. Balci, W. Gilley, and A. Robin, “Online Interactive Modules for Teaching Computer Science - Virginia Tech: The Spiral Model.” [Online]. Available: http://courses.cs.vt.edu/~csonline/SE/Lessons/Spiral/index.html. [Accessed: 11-Sep-2014].
[85] J. Martin, *Rapid Application Development*. Mac Millan, 1991.

[86] "IBM Rational software and systems delivery," 26-Aug-2014. [Online]. Available: http://www-01.ibm.com/software/rational/. [Accessed: 03-Sep-2014].
[87] A. K. Shuja and J. Krebs, *IBM Rational Unified Process Reference and Certification Guide: Solution Designer (RUP)*. Pearson Education, 2007.
[88] I. Jacobson, *Object-oriented software engineering: a use case driven approach*. ACM Press, 1992.
[89] "Skillful and maneuverable: OpenUP and the Eclipse Way," 15-Jul-2008. [Online]. Available: http://www.ibm.com/developerworks/rational/library/edge/08/jul08/vanVelzen/. [Accessed: 11-Sep-2014].
[90] "Eclipse Process Framework Project (EPF)." [Online]. Available: http://www.eclipse.org/epf/. [Accessed: 04-Sep-2014].
[91] Beck, Beedle, and Bennekum, "Agile Manifesto," 2001. [Online]. Available: http://www.agilemanifesto.org/. [Accessed: 19-Sep-2014].
[92] Highsmith and Cockburn, "Agile Software Development: The Business of Innovation," *Computer - IEEE Computer Society*, 2001.
[93] B. Boehm, "Get ready for agile methods, with care," *Computer*, vol. 35, no. 1, pp. 64–69, 2002.
[94] F. Paetsch, A. Eberlein, and F. Maurer, "Requirements engineering and agile software development," in *2012 IEEE 21st International Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises*, 2003, pp. 308–308.
[95] A. Dardenne, A. Van Lamsweerde, and S. Fickas, "Goal-directed requirements acquisition," *Sci. Comput. Program.*, vol. 20, no. 1, pp. 3–50, 1993.
[96] A. I. Antón, "Goal-Based Requirements Analysis," *Proc. OflCRE*, vol. 96, p. 136, 1996.
[97] P. Bertrand, R. Darimont, E. Delor, P. Massonet, and A. van Lamsweerde, "GRAIL/KAOS: An Environment for Goal-Driven Requirements Engineering," 1997.
[98] A. Van Lamsweerde, R. Darimont, and E. Letier, "Managing conflicts in goal-driven requirements engineering," *Softw. Eng. IEEE Trans. On*, vol. 24, no. 11, pp. 908–926, 1998.
[99] A. Van Lamsweerde, "Requirements engineering in the year 00: A research perspective," in *Proceedings of the 22nd international conference on Software engineering*, 2000, pp. 5–19.
[100] A. Van Lamsweerde and E. Letier, "From object orientation to goal orientation: A paradigm shift for requirements engineering," in *Radical Innovations of Software and Systems Engineering in the Future*, Springer, 2004, pp. 325–340.
[101] E. S. Yu, "Towards modelling and reasoning support for early-phase requirements engineering," in *Requirements Engineering, 1997., Proceedings of the Third IEEE International Symposium on*, 1997, pp. 226–235.
[102] A. I. Anton, "Goal identification and refinement in the specification of software-based information systems," Dissertation, 1997.
[103] J. Mylopoulos, L. Chung, and B. Nixon, "Representing and using nonfunctional requirements: A process-oriented approach," *Softw. Eng. IEEE Trans. On*, vol. 18, no. 6, pp. 483–497, 1992.
[104] A. Van Lamsweerde, "Goal-oriented requirements engineering: A guided tour," in *Requirements Engineering, 2001. Proceedings. Fifth IEEE International Symposium on*, 2001, pp. 249–262.
[105] E. Kavakli, "Goal-oriented requirements engineering: A unifying framework," *Requir. Eng.*, vol. 6, no. 4, pp. 237–251, 2002.

[106] G. Regev and A. Wegmann, “Revisiting Goal-Oriented Requirements Engineering with a Regulation View,” in *Business Modeling and Software Design*, Springer, 2012, pp. 56–69.
[107] G. Regev and A. Wegmann, “Where do goals come from: the underlying principles of goal-oriented requirements engineering,” in *Requirements Engineering, 2005. Proceedings. 13th IEEE International Conference on*, 2005, pp. 353–362.
[108] S. Pachidi, “Goal-Oriented Requirements Engineering with KAOS,” Utrecht University, 2010.
[109] K. E. Wiegers, *Software Requirements*, 2nd ed. Redmond, WA, USA: Microsoft Press, 2003.
[110] P. Loucopoulos and V. Karakostas, *System Requirements Engineering*. New York, NY, USA: McGraw-Hill, Inc., 1995.
[111] A. M. Hickey and A. M. Davis, “Requirements elicitation and elicitation technique selection: model for two knowledge-intensive software development processes,” in *System Sciences, 2003. Proceedings of the 36th Annual Hawaii International Conference on*, 2003, p. 10–pp.
[112] S. Robertson and J. Robertson, *Mastering the Requirements Process: Getting Requirements Right*. Addison-Wesley, 2012.
[113] S. Robertson and J. Robertson, “Requirements food chain,” The Atlantic Systems Guild, 2014.
[114] S. Beecham, T. Hall, and A. Rainer, “Defining a Requirements Process Improvement Model,” *Dep. Comput. Sciense Fac. Eng. Inf. Scienses*, 2003.
[115] M. C. Paulk, B. Curtis, M. B. Chrissis, and C. V. Weber, “Capability maturity model, version 1.1,” *Softw. IEEE*, vol. 10, no. 4, pp. 18–27, 1993.
[116] N. A. Maiden and C. Ncube, “Acquiring COTS software selection requirements,” *Softw. IEEE*, vol. 15, no. 2, pp. 46–56, 1998.
[117] R. Land, L. Blankers, M. Chaudron, and I. Crnković, “COTS selection best practices in literature and in industry,” in *High Confidence Software Reuse in Large Systems*, Springer, 2008, pp. 100–111.
[118] R. Winter and B. Strauch, “A method for demand-driven information requirements analysis in data warehousing projects,” in *System Sciences, 2003. Proceedings of the 36th Annual Hawaii International Conference on*, 2003, p. 9–pp.
[119] A. Stellman and J. Greene, *Applied Software Project Management*. O’Reilly Media, 2006.
[120] F. P. Brooks Jr, “No silver bullet - Essence and accidents of software engineering,” *Milest. Softw. Evol.*, p. 293, 1986.
[121] P. Britos, O. Dieste, and R. García-Martínez, “Requirements elicitation in data mining for business intelligence projects,” in *Advances in Information Systems Research, Education and Practice*, Springer, 2008, pp. 139–150.
[122] P. Britos, P. Pytel, and R. García-Martínez, “Initial activities oriented to reduce failure in information mining projects,” 2012.
[123] F. R. S. Paim and J. F. B. de Castro, “DWARF: an approach for requirements definition and management of data warehouse systems,” in *Requirements Engineering Conference, 2003. Proceedings. 11th IEEE International*, 2003, pp. 75–84.
[124] M. Hofmann and B. Tierney, “The Involvement of Human Resources in Large Scale Data Mining Projects,” in *Proceedings of the 1st International Symposium on Information and Communication Technologies*, Dublin, Ireland, 2003, pp. 103–109.

[125] J. Dietz, *Enterprise Ontology: Theory and Methodology*. Springer Science & Business Media, 2006.
[126] T. Halpin and T. Morgan, *Information Modeling and Relational Databases*. Morgan Kaufmann, 2010.
[127] "Intervac Home Exchange," *Intervac Home Exchange*. [Online]. Available: http://www.intervac-homeexchange.com/. [Accessed: 27-Oct-2014].
[128] Australian Government and Centrelink, Eds., "Centrelink Annual Report 2004-05." 2005.
[129] A. Eberlein and L. Jiang, "SELECTING REQUIREMENTS ENGINEERING TECHNIQUES," in *Encyclopedia of Software Engineering*, Taylor & Francis, 2010.
[130] D. Jitnah, J. Han, and P. Steele, "Software requirements engineering: An overview," *Penins. Sch. Comput. Inf. Technol. Monash Univ.*, 1995.
[131] H. R. Beyer and K. Holtzblatt, "Apprenticing with the customer," *Commun. ACM*, vol. 38, no. 5, pp. 45–52, 1995.
[132] J. Goguen and C. Linde, "Techniques for requirements elicitation," *Requir. Eng.*, pp. 152–164, 1993.
[133] R. R. Hoffman, "The problem of extracting the knowledge of experts from the perspective of experimental psychology," *AI Mag.*, vol. 8, no. 2, p. 53, 1987.
[134] R. R. Hoffman and G. Lintern, "Eliciting and representing the knowledge of experts," *Camb. Handb. Expert. Expert Perform.*, pp. 203–222, 2006.
[135] G. Rugg and P. McGeorge, "The sorting techniques: a tutorial paper on card sorts, picture sorts and item sorts," *Expert Syst.*, vol. 14, no. 2, pp. 80–93, May 1997.
[136] C. Davis, R. Fuller, Tremblay, and Berndt, "Communication challenges in requirements elicitation and the use of the repertory grid technique," *The Journal of Computer Information Systems*, pp. 78–86, 2006.
[137] R. W. Stoddard, M. D. Konrad, and N. R. Mead, "Elicitation of Unstated Needs: KJ+ Method Overview," presented at the 22nd IEEE International Requirements Engineering Conference, Software Engineering Institute, 17-Sep-2014.
[138] J. L. G. Dietz, "DEMO-3 Models and Representations," 2009. [Online]. Available: http://www.demo.nl/publications/. [Accessed: 10-Oct-2012].
[139] B. Langefors, "Information systems theory," *Inf. Syst.*, vol. 2, no. 4, pp. 207–219, 1977.
[140] M. Dumay, J. L. G. Dietz, H. Mulder, G. Goldhuhl, M. Lind, and S. Haraldson, "Evaluation of DEMO and the Language/Action Perspective after 10 years of experience," in *International Working Conference on the Language Action Perspective on Communication Modeling, Kiruna, Lapland, Sweden*, 2005.
[141] J. L. Dietz, "A world ontology specification language," in *On the Move to Meaningful Internet Systems 2005: OTM 2005 Workshops*, 2005, pp. 688–699.
[142] T. Halpin, "ORM/NIAM Object-Role Modeling," in *Handbook on Architectures of Information Systems*, D. P. Bernus, P. D. K. Mertins, and P. D. G. Schmidt, Eds. Springer Berlin Heidelberg, 1998, pp. 81–101.

10 Appendixes

Appendix I - Requirements elicitation techniques

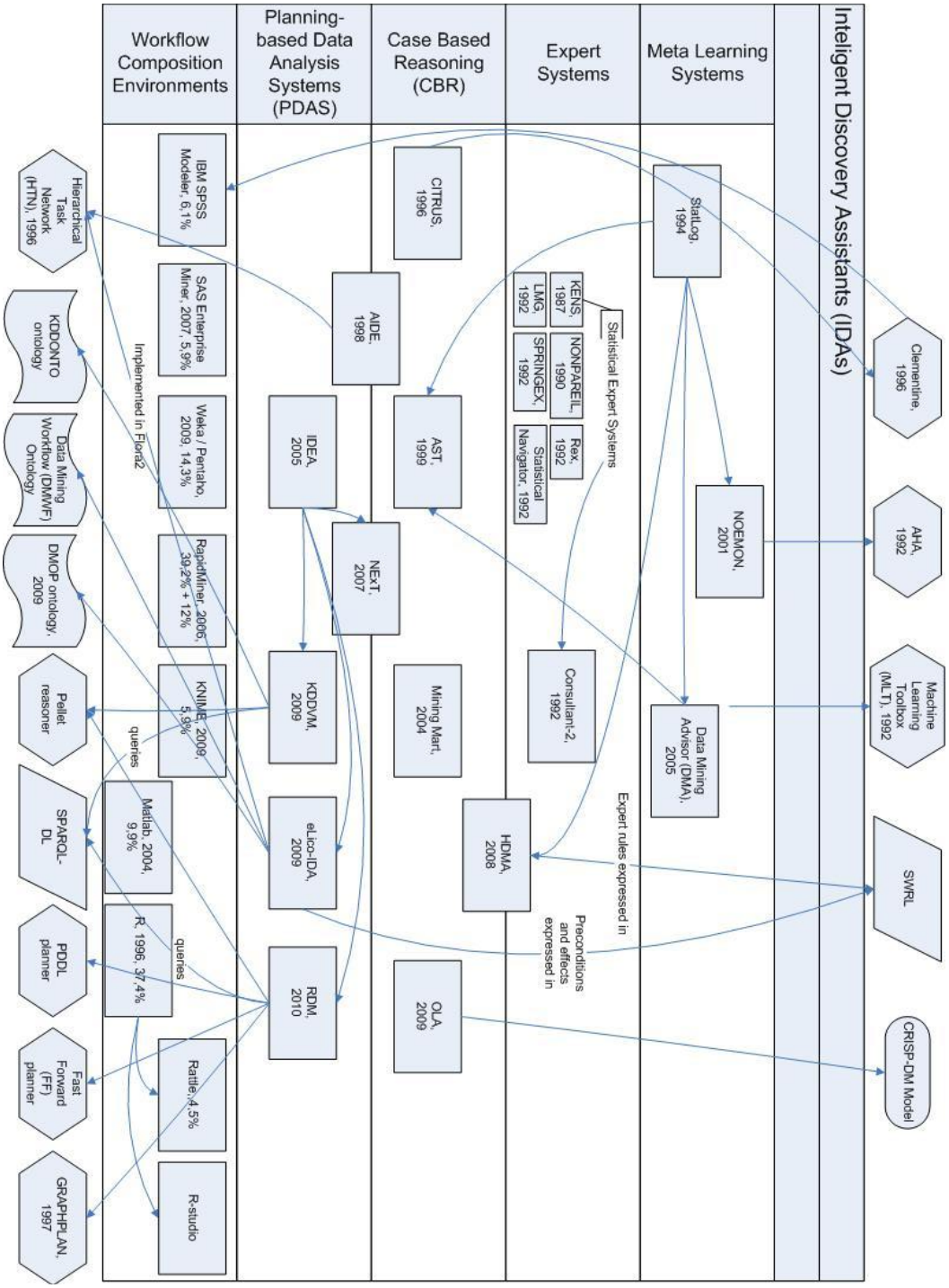
Table 3 presents a list of possible elicitation techniques. [75] has more information on this subject and on some of the techniques. For information on choice criteria for elicitation techniques, [129], [112] and [76] provide valuable information.

Technique	**Description**
Open-ended / unstructured interview	This is an in-person interaction between elicitor and informant where goals are not completely defined and contents are not structured.
Structured interview	This is an in-person interaction between elicitor and informant where goals and contents have been prepared and structured [130]. It is possible to allow for spontaneous unstructured questions during the course of the interview, which, in that case is a semi-structured interview.
Contextual inquiry, designer as apprentice	This is a semi-structured interview method to obtain information about the context of use. While the informants work in their own environments, they are first asked a set of questions and then observed and questioned [131].
Task observation	This involves the observation by the elicitor of people at work.
Protocol analysis	The informants relate aloud what they do when they perform specific tasks [132].
Contrived tasks	This involves the observation by the elicitor of people performing familiar tasks, but with limited information or specific constraints, such as a time limit [133] [134].
Tough cases	This involves the observation by the elicitor of people performing familiar tasks, but one that they consider a "tough case" [133].
Participant observation	The elicitor actively participates in some tasks to find out what skills and knowledge are required to effectively perform them or to get acquainted with the domain.
Analysis of existing documentation	The elicitor analyses documents such as organisational charts, process models or descriptions, standards or user manuals of existing systems.
Card sorting, concept ranking, laddering	The elicitor gives informant(s) a set of cards with domain concepts written on them and asks the informant(s) to sort or organize them in some way [135].

Repertory grid	The informants have to evaluate a set of domain elements based on constructs (element characteristics) [136]. This technique involves three phases: creation, assessment and clarification.
Surveys / questionnaires	A set of questions on paper, e-mail, form or another medium are presented or sent to one or more informants.
Brainstorming	Informants, individually or as a group, come up with ideas, deliberatively and in no particular order. They are not immediately assessed so that they help generate new ideas.
KJ Method, KJ+ method	Informants use this technique to brainstorm and reach group consensus. The KJ method is named after Jiro Kawakita, the Japanese anthropologist who originally conceived it. The KJ+ method was conceived at the Software Engineering Institute of the Carnegie Mellon University specifically for RE [137].
Nominal group technique	The informants come up with ideas that are formally voted on for prioritization.
Focus group	This is an in-person interaction between the elicitor and a group of informants, in the form of semi-structured group interviews encouraging open discussion.
Delphi method	The informants are asked for feedback of responses to a questionnaire with the aim of outputting an outcome representative of the group's opinion.
Prototyping (exploratory or evolutionary)	This involves developing a simplified version of the system and get informants feedback on it to help capture requirements.
RAD/JAD workshop	This involves facilitated meetings between different stakeholders.
Use cases/ scenarios	Informants describe a set of possible action scenarios and events describing part of system behaviour.

Table 3 – Elicitation Techniques

Appendix II - Intelligent Discovery Assistants



Appendix III - DEMO

In this appendix we introduce the aspects of DEMO that are relevant for this thesis. This summary may be needed if the reader is unfamiliar with DEMO. We focused on the more practical aspects of DEMO. Whenever a concept in the current appendix is not sufficiently clear, the full explanation should be consulted in [125] and [138].

The Ψ theory is the base for DEMO and consists of a set of four axioms and a theorem fully presented in [125].

Let's first consider the operation axiom that states that an organization consists of actors (human beings fulfilling an actor role) who perform two kinds of acts:

- Production acts (P-acts) - When the actors perform P-acts, the function of the organization is fulfilled. P-acts may be material (such as manufacturing or carrying) or immaterial (such deciding, evaluating or diagnosing).
- Coordination acts (C-acts) - By performing C-acts actors enter into commitments and comply with them. In doing so, they initiate and coordinate the execution of production acts.

The result of successfully performing a P-act is a production fact or P-fact. The result of successfully performing a C-act is a coordination fact or C-fact.

Let's now consider the transaction axiom. This axiom states that P-acts and C-acts occur in generic recurrent patterns called transactions.

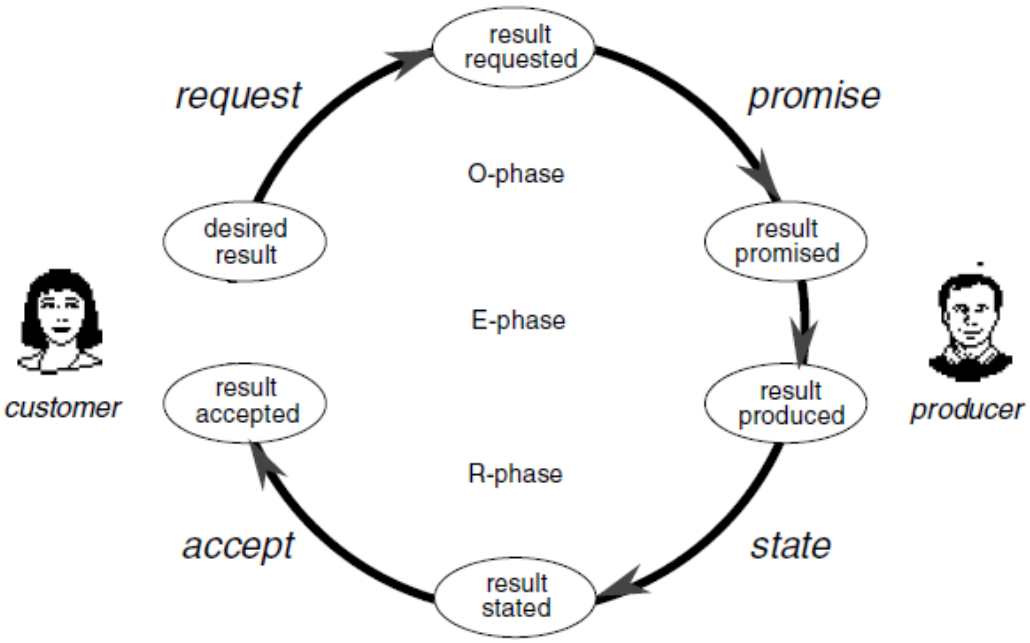


Figure 33 - Transaction - basic pattern [125]

It is carried through by two actors:

- The initiator (shown in Figure 33 as "customer") - The actor who starts the transaction and eventually completes it
- The executor (shown in Figure 33 as "producer") - The other actor, who actually performs the production act.

The transaction starts with a request by the initiator which results (if successful) in a promise by the executor. The executor performs the P-act. Then there is a statement by the executor and (if successful) the transaction ends with an acceptance by the initiator.

Requesting, promising, stating or accepting a P-fact are all C-acts which may be perform tacitly in some cases.

The basic transaction pattern can be completed by adding two dissent patterns and four cancellations patterns as shown in Figure 34.

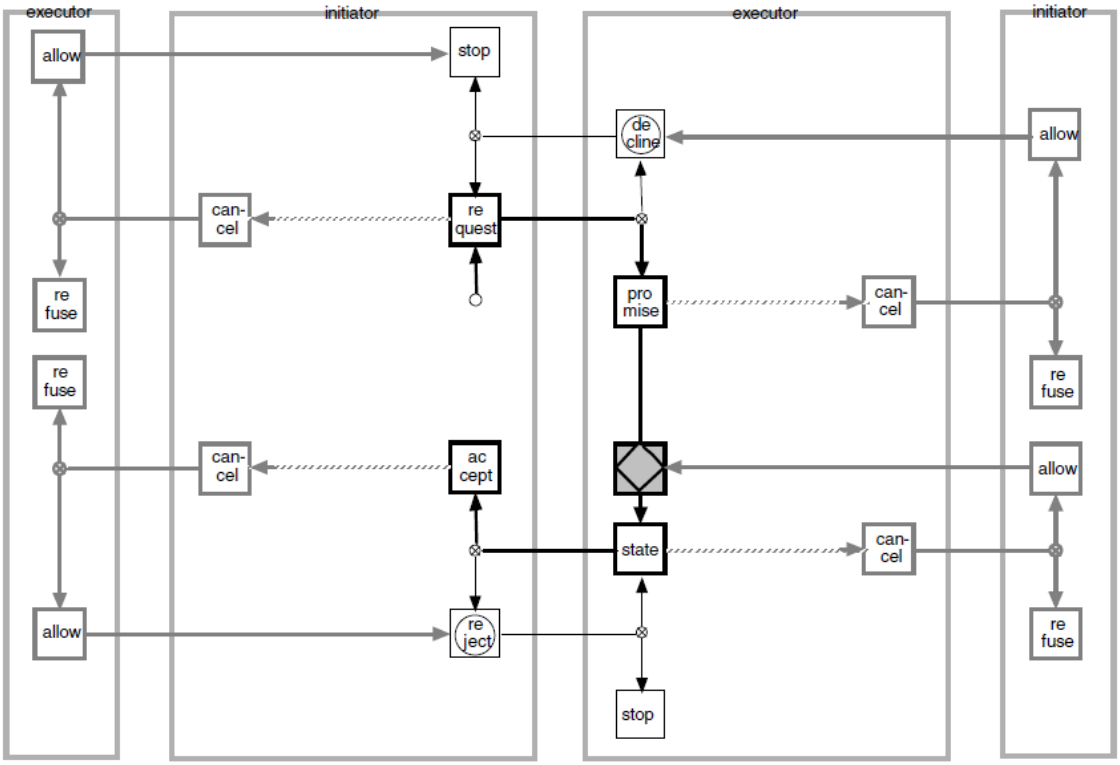


Figure 34- Complete transaction pattern [125]

In Figure 34, white boxes represent C-acts and white circles represent C-facts. Gray boxes represent P-acts and gray diamond shapes represent P-facts. The initial C-act and every terminal C-fact are drawn with a bold line. The responsibility areas of the two actor roles are marked by the large gray frames, denoted "initiator" and "executor".

Next to the basic transaction steps discussed above, it is now possible to have a decline instead of a promise, and a reject instead of an accept. Both of these C-facts are discussion states, where the two actors have to try to come to a (new) agreement. When unsuccessful, the transaction is stopped, either by the initiator or by the executor. Four cancellation patterns, on the left and the right side, complete the transaction pattern, one for every basic step.

Every transaction process is some path through this complete pattern, and every business process in every organization is a connected collection of such transaction processes. This holds also for processes across organizations, like in supply chains and networks. That is why the transaction pattern is universal.

Let's now consider a third axiom, the distinction axiom. This axiom states that there are three distinct human abilities, called forma, informa, and performa.

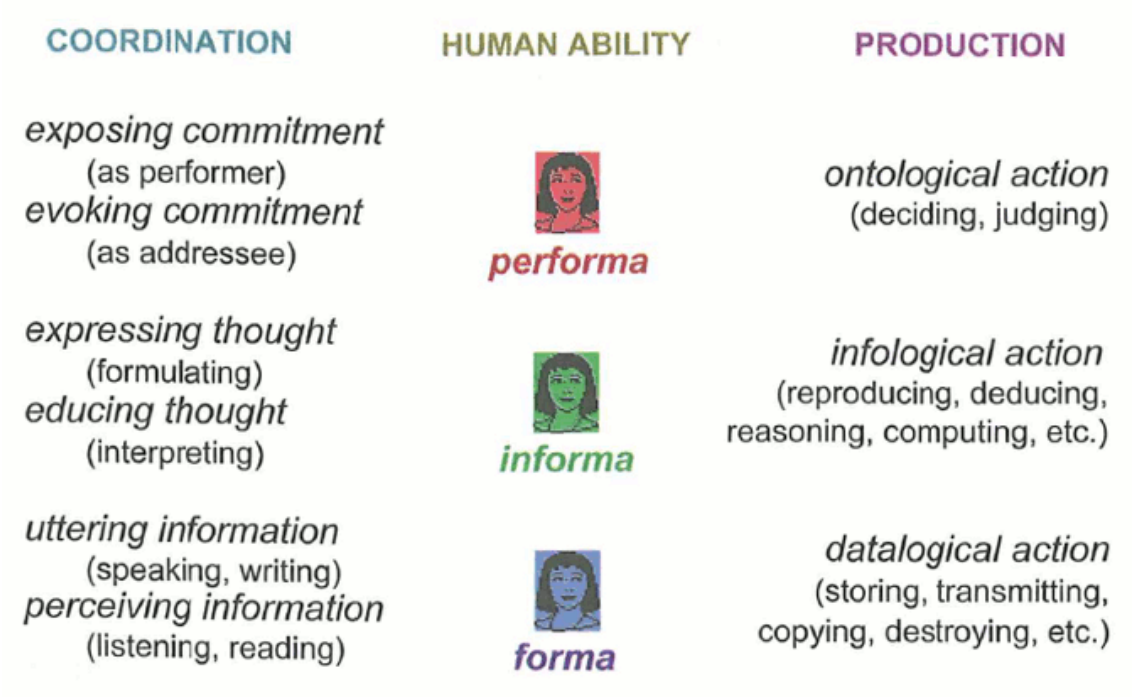


Figure 35 - Three human abilities: forma, informa, and performa [125]

The forma ability concerns the form aspects of communication and information. In other words, forma refers to such things as the uttering and perceiving of sentences in some language, the syntactical analysis of such sentences, coding schemes, transmission of data, and storage and retrieval of data or documents. For the sake of convenience, the forma ability is considered to also comprise the physical substrate in which information is encoded.

Considering the production world, the forma ability concerns the ability to deal with recorded information items, called data or documents. Examples of P-acts are storing, copying, transmitting, and destroying data or documents. Collectively, these acts are called datalogical acts, following the classification done in [139]. The transactions in which datalogical acts occur are called datalogical transactions.

The informa ability involves the content aspects of communication and information. In other words, informa refers to such things as the sharing of thoughts between people, the remembering and recalling of knowledge, and reasoning.

The informa ability on the production side concerns the ability to things like reasoning or computing, as well as reproducing knowledge as you remember it. These acts are collectively called infological acts, and the transactions in which they figure as P-acts are called infological transactions.

The performa ability concerns the bringing about of new, original things, directly or indirectly by communication. We are talking about engaging into commitments, as well as about decisions and judgments. The performa ability is considered to be the essential human ability for doing business of any kind.

This performa ability concerns the ability to establish original new things, like creating material products or making decisions. These acts are at the core of doing business (on the production side). This is why they are called the essential or ontological production acts. Accordingly, the transactions in which they occur as P-acts are called ontological transactions.

To conclude the presentation of the theoretical aspects, we will now briefly consider the organization theorem. This theorem, graphically represented in Figure 36, states that the organization of an enterprise is constituted as the layered integration of three systems: the B-organization (from Business), the I-organization (from Intellect), and the D-organization (from Document).

They differ only in the kind of production: the production in the B-organization is ontological, the production in the I-organization is infological, and the production in the D-organization is datalogical. The B-organization, the I-organization, and the D-organization are called aspect systems of the (total) organization of the enterprise.

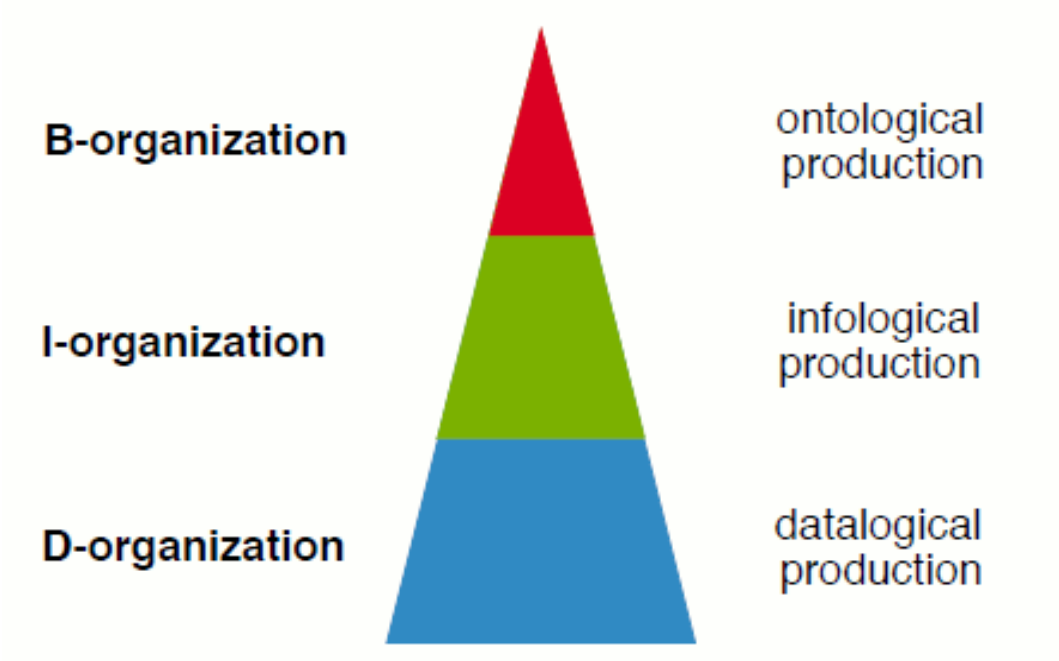


Figure 36 - Representation of the organization theorem [125]

In DEMO, the complete ontological model of an organization is understood as the model of its B-organization. It consists of four aspect models, represented by particular diagrams,

tables and lists: the Construction Model (CM) - constituted by the Interaction Model (IAM) and the Interstriction Model (ISM) - the Process Model (PM), the Action Model (AM) and the State Model (SM). These models constitute the complete ontological model of the B-organization and subsequently represent the ontological model of the corresponding enterprise.

The Construction Model (CM) specifies the construction of the organization system by the identified transaction kinds and the associated actor roles, as well as the information links between the actor roles and the information banks. It is the most concise model and is split into two parts, the active part, the Interaction Model (IAM), and the passive part, the Interstriction Model (ISM). The interaction structure of an organization consists of the transaction kinds in which the identified actor roles participate as initiator or executor, while the interstriction structure shows passive system structure i.e. the information links between actor roles and banks.

The Process Model (PM) contains, for every type in the CM, the specific transaction pattern (basic pattern, standard pattern, cancellation pattern) of the transaction kind. In other words, it contains the detailing of the identified transaction kinds. It also contains the causal and conditional relationships between transactions. The PM specifies the state space and the transition space of the coordination world.

The Action Model (AM) specifies the imperatively formulated business rules that serve as guidelines for the actors in dealing with their agenda2. It contains one or more action rules for every agendum type.

The State Model (SM) specifies the state space and the transition space of the production world with: object classes, fact types, results types and ontological coexistence rules. These can be considered as declarative formulations of business rules.

The logical sequence and proposed method of producing the aspect models starts with the interaction model (IAM). The first result is a list of the identified transaction kinds and the participating actor roles, as well as the identification of the system boundary. The system boundary depends on the object of analysis, but typically represents the border of an organization or a department.

From the transaction list, the IAM can be made straight away. It is composed by the Transaction Result Table (TRT) and an Actor Transaction Diagram (ATD).

The ATD displays both the internal (elementary) actor roles and external (composite) actor roles, related to each other by transactions.

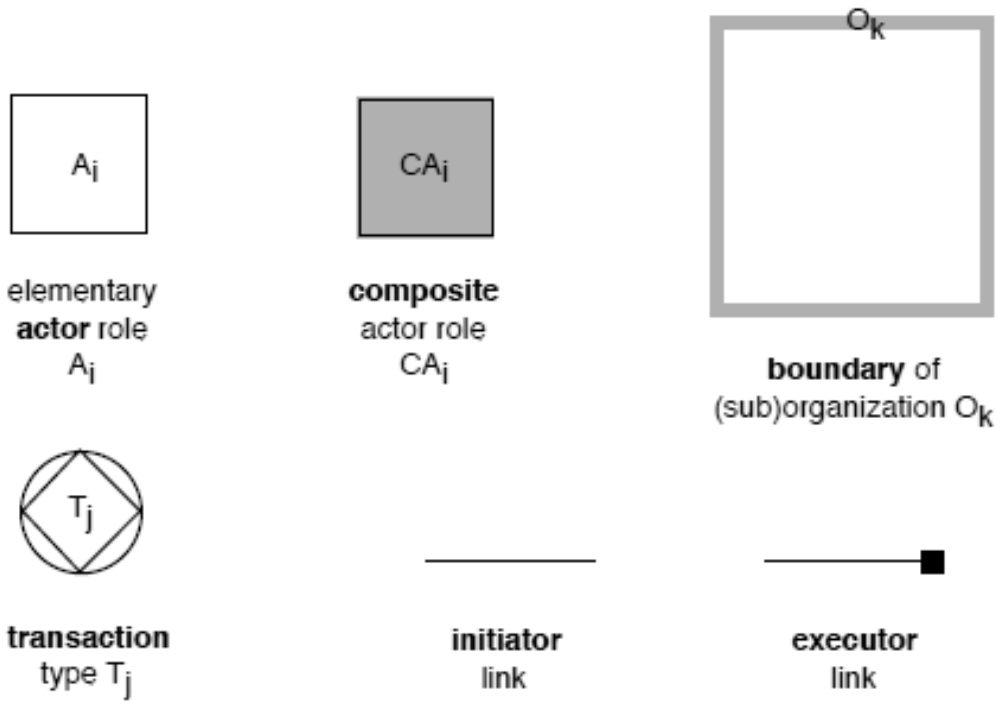


Figure 37 - Actor Transaction Diagram (ATD) legend [125]

Next, the Process Model (PM) is produced. A process model is usually expressed in terms of a Process Structure Diagram (PSD).

The PM is the specification of the state space and the transition space of the C-world; thus, the set of possible or allowed sequences of states in the C-world. In other words, the PSD should specify the process steps that are allowed to be taken for every included transaction type. As a consequence, steps that are not included in the PSD are not allowed.

Since every transition in the C-world consists of the creation of a C-result and since there is a one-to-one relationship between this C-result and the causing C-act, these C-acts are also contained in the PM. A C-result and its causing C-act are collectively called a process step. The PM specifies also for every process step the information used to perform the step. As a convenient addition, the PM duplicates the knowledge from the CM concerning which actor roles perform the C-acts. They are called responsibility areas.

.

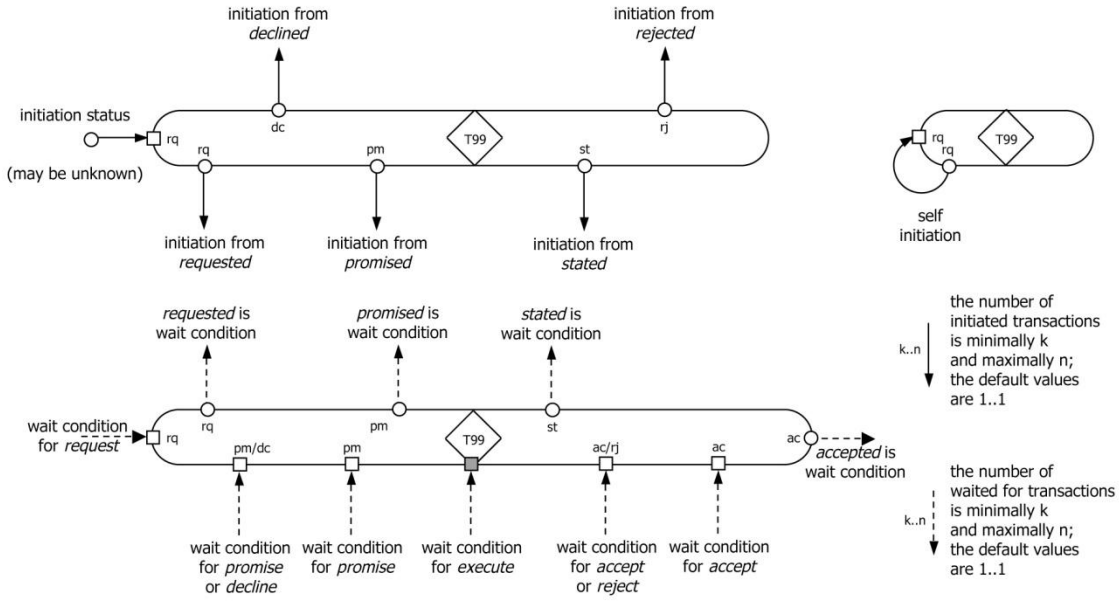


Figure 38 - Process Structure Diagram (PSD) legend [138]

After that the Action Action Model (AM) is produced. The AM is represented by means of Action Rule Specifications (ARS). The action rules are expressed in a pseudo-algorithmic language.

In principle, one can find the categories, object classes, fact types, and result types, as well as all pertaining existential laws and all derivation rules, in the Action Model specifications. In practice, however, one will also refer to the available documentation of the organization at hand.

In fact, in 2005 study [140] the Action Model is not part of any real-world projects the respondents participated in and, in general, the most used model of DEMO methodology was the Construction Model. The Process Model was typically used in the field of business process reengineering, while the Information Model is more applied in the field of Information Systems development.

Next, the State Model (SM) is produced. It consists of specifying the object classes, the fact types, and the result types, as well as the existential laws that hold. The main construct of the SM is the Object Fact Diagram (OFD). An Object Property List (OPL) might also be produced as part of the SM.

The OFD is specified using the World Ontology Specification Language (WOSL) [141], which is, in turn, strongly based in the Object Role Modeling Language (ORM) [142]. The legend for the OFD is shown in two parts, in Figure 39 and Figure 40.

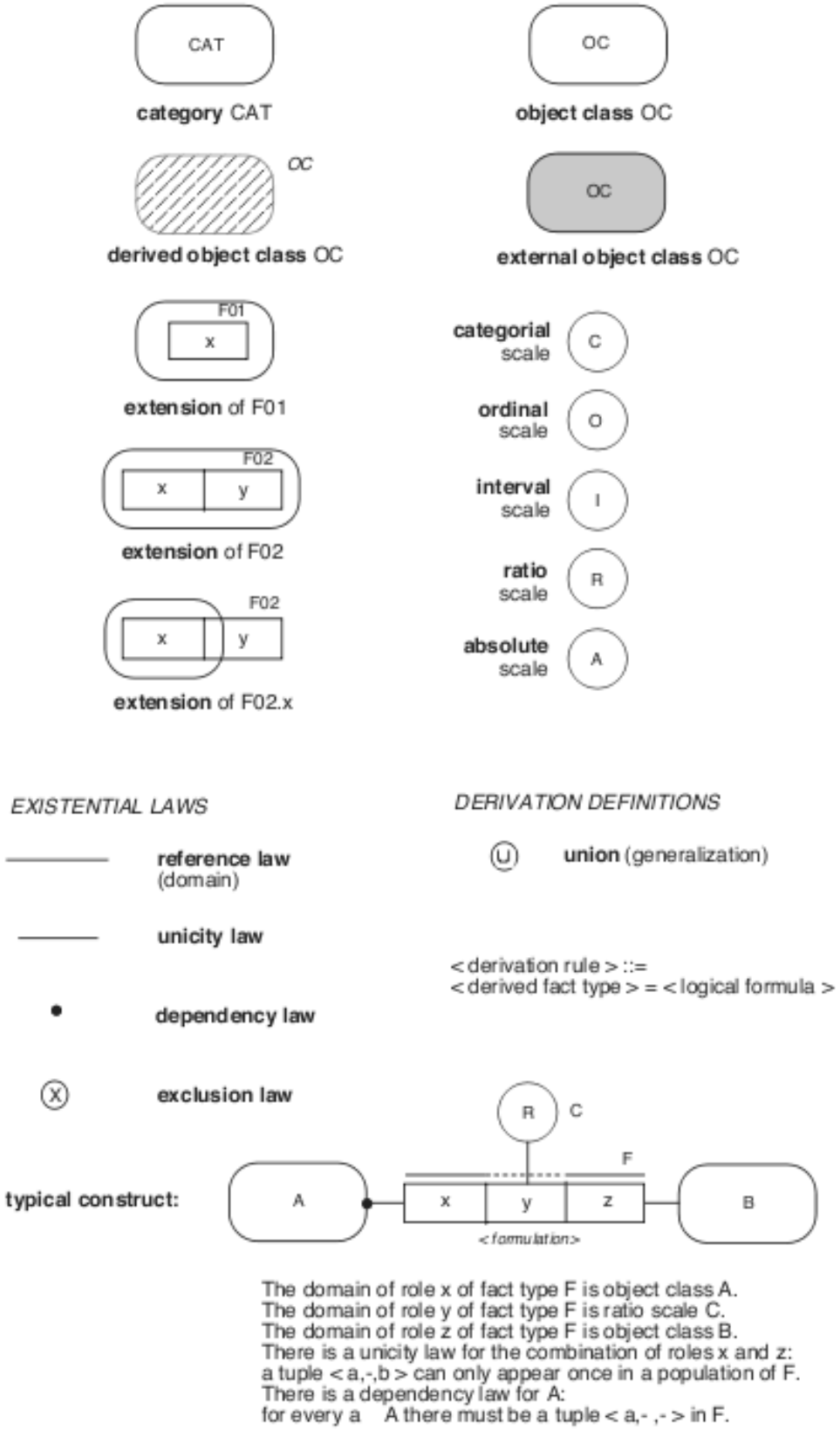


Figure 39 - Object Fact Diagram legend (first part) [138]

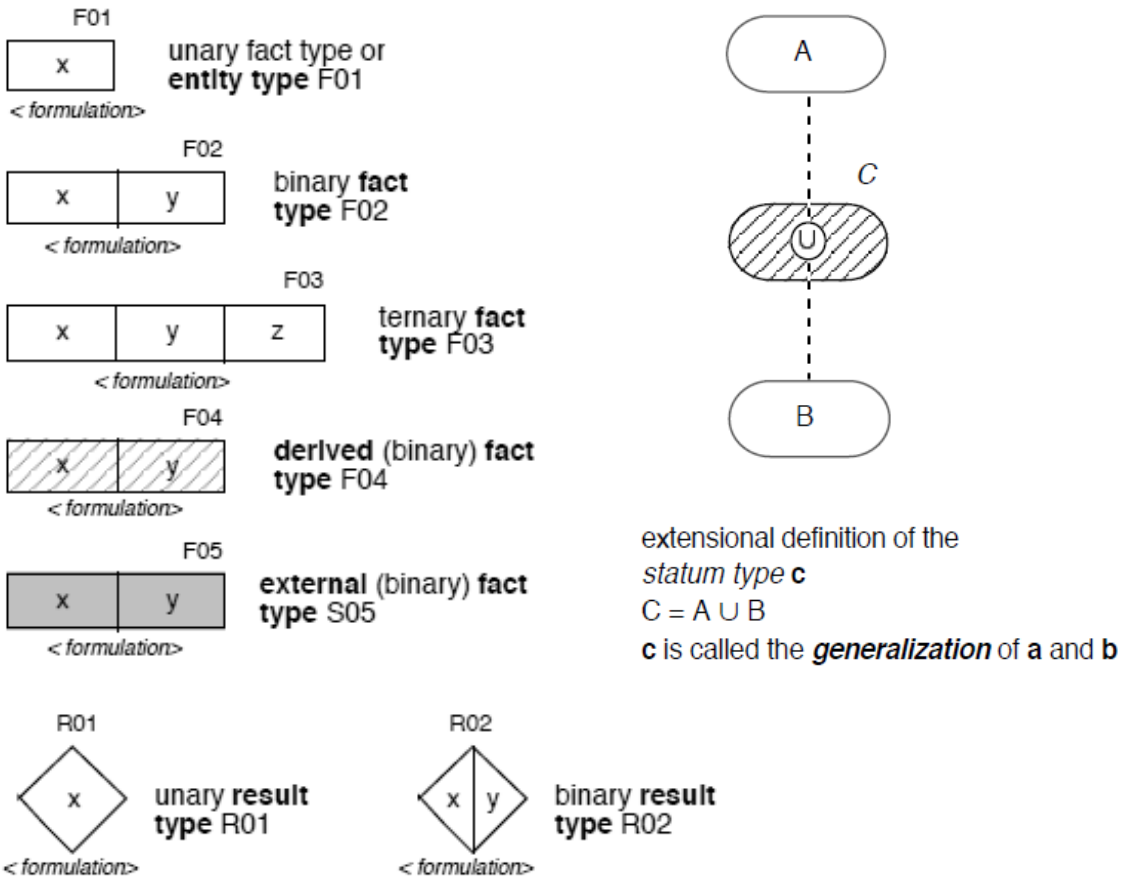


Figure 40 - Object Fact Diagram legend (second part) [138]

Finally, the ISM is produced, consisting of an Actor Bank Diagram (ABD) and a Bank Contents Table (BCT). The Actor Bank Diagram is usually drawn as an extension of the Actor Transaction Diagram and together they constitute the Organization Construction Diagram (OCD).

The general elicitation method to acquire the basis for a correct and complete set of aspect models of an enterprise ontology consists of three analysis and three synthesis steps. The starting point is all available documentation about the enterprise, of whatever kind, and in whatever form.

1. Perfoma-Informa-Forma Analysis - in this step all available pieces of knowledge are divided in three sets, according to the distinction axiom. This is can be difficult, since in natural language descriptions words and sentences may belong to more than one of these sets.

2. Coordination-Actors-Production Analysis - the Performa items are divided into C-acts/results, P-acts/results, and actor roles, according to the operation axiom.
3. Transaction Pattern Synthesis - the transaction pattern is used to cluster the results into transaction kinds. Next, for every transaction kind, the result type is correctly and precisely formulated. The Transaction Result Table can now be produced.
4. Result Structure Analysis - according to the composition axiom, every transaction kind of which an actor in the environment is the initiator may be conceived as delivering and end result to the environment. Generally, the (internal) executor of this transaction kind is initiator of one or more other transaction kinds, and so on. The results of these cascaded transactions can be viewed as components of the end result.
5. Construction Synthesis - for every transaction kind, the initiating actor role(s) and the executing actor role are identified, based on the transaction axiom. This is the first step in producing the Actor Transaction Diagram.
6. Organization Synthesis - a definite choice has to be made as to what part of the construction will be taken as the organization to be studied and what part will become its environment. The Actor Transaction Diagram can now be finalized.

It is very important to note that [125] considers that one may freely iterate through these six steps (and even skip a step) and that one should always keep in mind that a method is an aid, not a dogma.

Appendix IV – Centrelink case instantiation

In this appendix we present the instantiation of SysPRE done for the Centrelink case study described in section 6.2

The main class of the OFD (which is shown in Figure 41 and Figure 42) is the KNOWLEDGE DISCOVERY PROCESS (KDP), related to the main transaction T01.

Let's start by having a look at the PROBLEM/OPPORTUNITY class. Instances of this class specify a problem or an opportunity that triggered the KDP. In Centrelink case it is striking that there is a large debt from customers. This is the problem/opportunity.

One very clear STAKEHOLDER is the Australian Government. This particular stakeholder has a GOAL/CORE ISSUE: they want to reduce the customer debt. From the case study we know that at least one ELICITATION TECHNIQUES was used: the analysis of existing documentation. Others might also have been used, but were not mentioned. Using this ELICITATION TECHNIQUE, a REQUIREMENT to satisfy the above GOAL/CORE ISSUE was elicited: Predict how likely a customer is to have debt. If it is possible to predict how likely it is that a customer has or will have debt, this debt can either be prevented or detected earlier, which will save many complicated procedures for the Centrelink and, in fact, both time and money.

From the accepted REQUIREMENTS, one will then proceed to create an HYPOTHESIS that can be tested in a KDP. For this case, one of the tested HYPOTHESIS was if the marital status, income, home ownership and age can be used to predict debt. In the end, the RESULT of the KDP will either confirm this hypothesis or not. For the KDP, there needs to be an estimation of COST AND RESOURCES, so that a Go-no-go decision (T14) can take place, but we have no details on this from the case study.

As the KDP proceeds, instances of classes corresponding to the DATA SOURCE (the main is Centrelink's transactional database), the data mining TOOL (which, in this case was Terabyte Warehouse Miner), the type of DATA MINING TECHNIQUE (in this case, decision trees were used first, with no success, and then association rules) will be used to obtain a particular RESULT. From the DATA SOURCE, DATA will be selected and prepared. In this particular case the ALGORITHM was not specified (because of the TOOL used), but still a DATA MINING PARAMETER was set (min_support) when the association rules were used.

From the DATA might result some kind of DATA CONSTRAINT, which is this case was the very high number of features (80) that proved to be a problem for the initial combination of TOOL/DATA MINING TECHNIQUE chosen. The identified DATA

CONSTRAINTS naturally affected the KDP, and in this case, association rules ended up being chosen instead of decision trees.

As mentioned before, the execution of a particular algorithm with particular parameters and applied to a particular data, in the context of a KDP will produce a particular RESULT. For the case study at hand, a rule was found that Customers who changed marital status once, have a partner with casual income and do not own a home, have a debt with a lift of 1,69, support of 0,004 and confidence of 0,65.

The RESULT is always target of an analysis (T25). From such analysis when the decision trees DATA MINING TECHNIQUE was used, the conclusion was that the results were totally useless. For this reason, at this point a different DATA MINING TECHNIQUE was chosen and the KDP restarted with association rules as the DATA MINING TECHNIQUE. This time the RESULT analysis was positive, it was possible to proceed to DEPLOYMENT, but, again, we have no information regarding this from the case study.

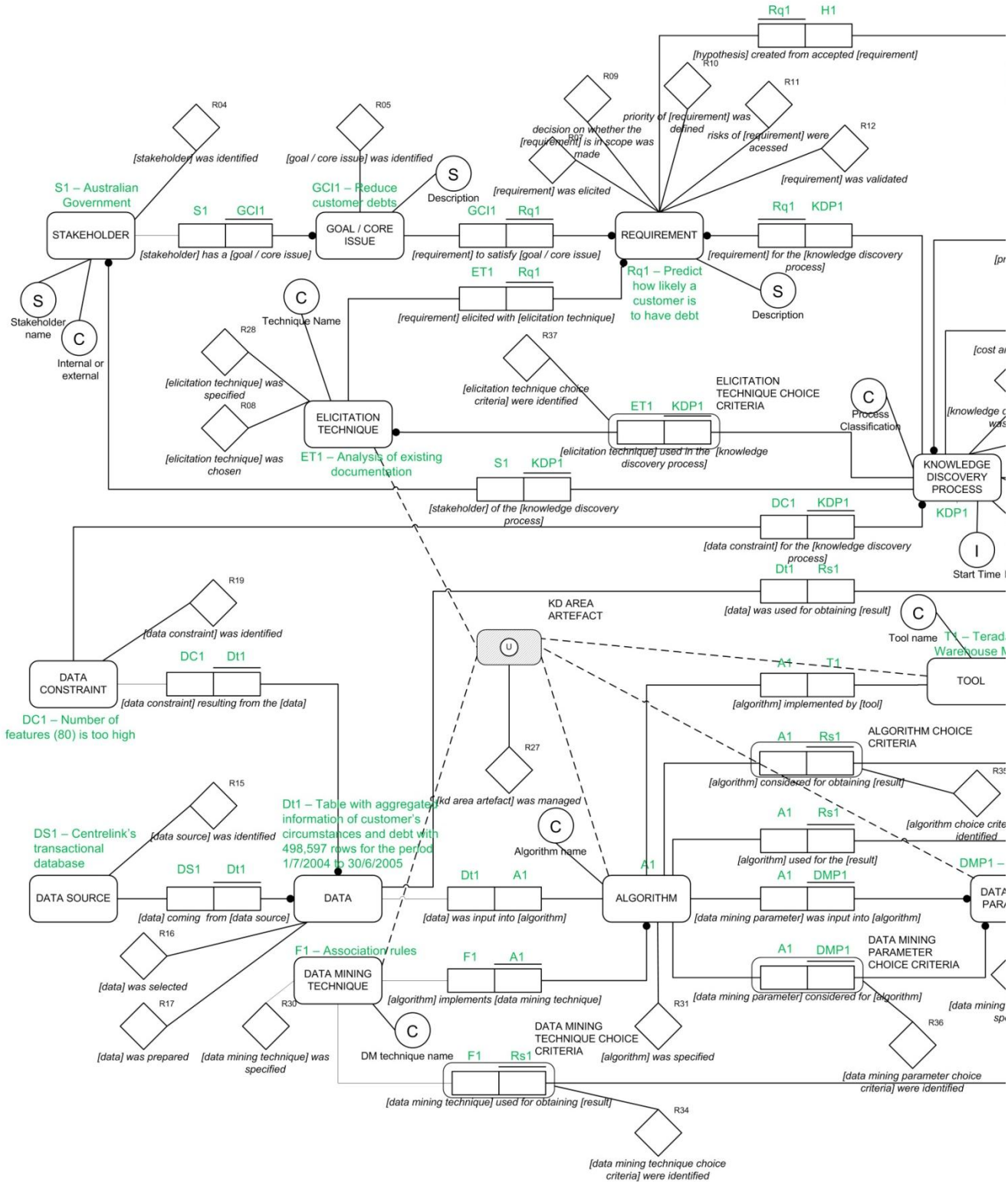


Figure 41 - Object Fact Diagram (OFD) for the Centrelink case study – first part

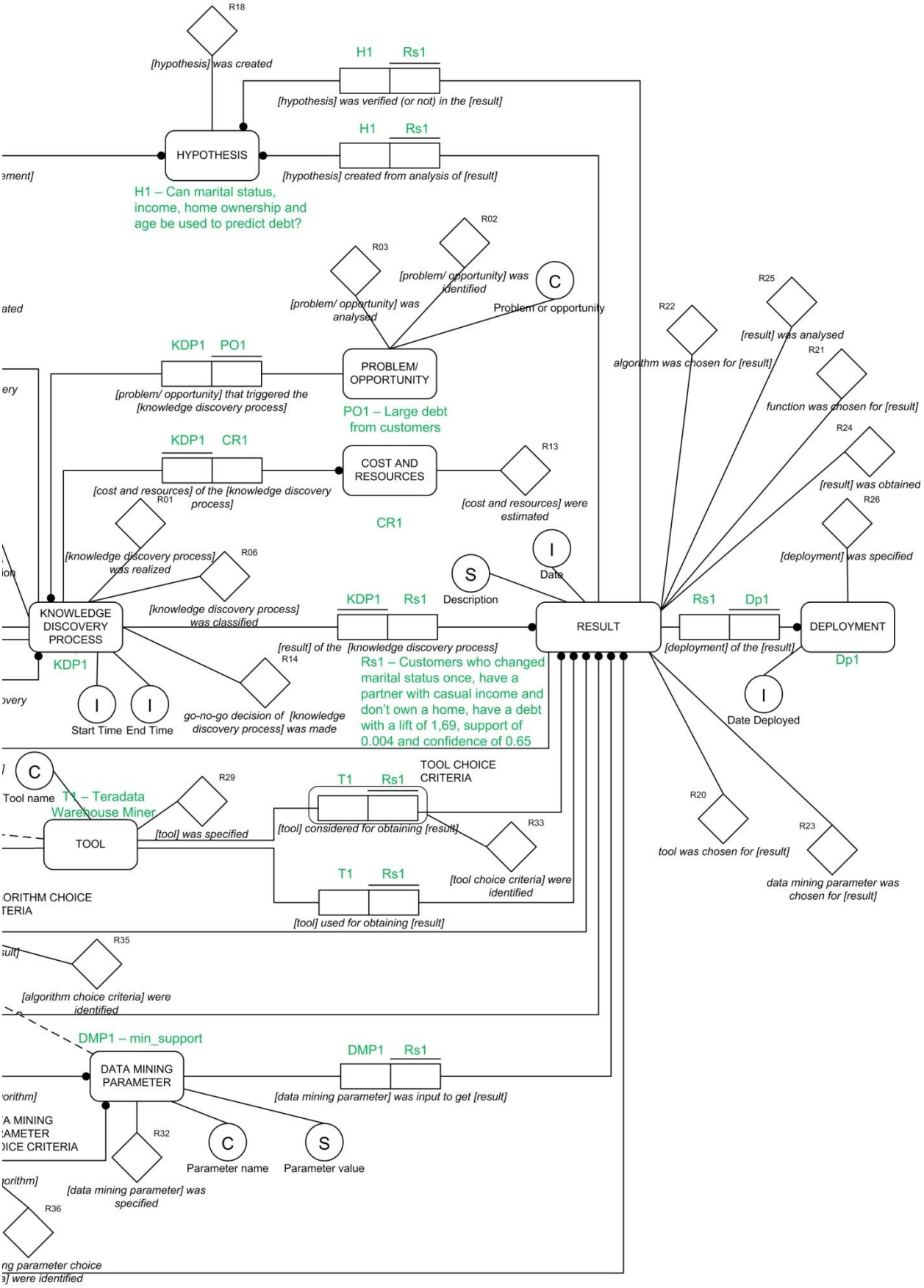


Figure 42 - Object Fact Diagram (OFD) for the Centrelink case study – second part